Robust Distance-Based Clustering with Applications to Spatial Data Mining

Robust Distance-Based Clustering with Applications to Spatial Data Mining

Vladimir Estivill-Castro Michael E. Houle

Revised Submission to Special Issue of Algorithmica:
- Algorithms for Geographical Information –
Topic: Optimization algorithms applied to geographical data, geostatistics, mathematical modeling of geographic problems.

Abstract

KEYWORDS: Clustering, spatial databases, data mining, exploratory data analysis, Delaunay triangulation, combinatorial optimization, p-median problem, medoids, noise.

In this paper, we present a method for clustering geo-referenced data suitable for applications in spatial data mining, based on the medoid method. The medoid method is related to k-MEANS, with the restriction that cluster representatives be chosen from among the data elements. Although the medoid method in general produces clusters of high quality, especially in the presence of noise, it is often criticized for the $\Omega(n^2)$ time that it requires.

Our method incorporates both proximity and density information to achieve high-quality clusters in subquadratic time; it does not require that the user specify the number of clusters in advance. The time bound is achieved by means of a fast approximation to the medoid objective function, using Delaunay triangulations to store proximity information.

1 Introduction

The rising popularity of Geographical Information Systems (GIS) has manifested itself in an increased demand for systems that can analyze and visualize geographical and spatial data [22]. However, the emergence of rapid data collection using remote sensing has resulted in the accumulation of huge volumes of spatially-referenced information in electronic and magnetic media. Geographic data and associated spatial data sets are now being generated faster than they can be meaningfully analyzed [4]. Spatial data mining [26, 28, 29, 41, 51, 52, 53] aims at benefiting from this information explosion. It provides automated and

semi-automated analysis of large volumes of spatial information associated with a GIS, and the discovery of interesting, implicit knowledge in spatial databases.

Central to spatial data mining is clustering [53, 62], which seeks to identify subsets of the data having similar characteristics. Clustering has been identified as one of the fundamental problems in the area of knowledge discovery and data mining [6, 35], and is of particular importance for spatial data sets. Clustering allows for generalization of the spatial component of the data associated with a GIS [53, 55, 62], and as such is complementary to the techniques for generalization used in data mining in relational databases [11]. Clustering of large geo-referenced data sets has many applications [53], and has recently attracted the attention of many researchers [25, 27, 29, 62, 88, 93, 94]. Some geographical data mining systems are based entirely on clustering [63].

Here, we consider the problem of knowledge discovery in spatial databases, those typically associated with GIS, by focusing on the clustering of points according to their spatial dimensions. Most approaches to knowledge discovery with geo-referenced data sets have proposed that the process be triggered by a user query [53, 55, 62, 88], in accordance with the requirements of on-line analytical processing. This essentially means that clustering would be performed on-line as new sectors, projections and selections of the data are produced. Clustering algorithms for spatial data mining must operate on large volumes of data, and for this reason cannot afford to spend too much time on finding clusterings of high quality. Algorithms must be especially fast if the knowledge discovery process is to be an exploratory type of analysis [39, 65], where many potential hypotheses must be constructed and tested.

Exploratory data analysis favors automatic knowledge discovery that allows the "data to speak for itself". Little human participation is involved, minimising the risk that the domain expert will knowingly or unknowingly bias the study to support a preconceived hypothesis [66]. However, exploratory data analysis methods tend to be resource intensive [67]. Nevertheless, the database community seems to favor semi-automatic knowledge discovery, in what has been named on-line analytical mining [40], where the system receives a reduced hypothesis space from the human user in the form of a query, or where inductive bias is to come from the user in selecting parameters from a visualization [3]. Both approaches stand to benefit from faster and more robust spatial clustering algorithms.

In spatial settings, clustering criteria almost invariably make use of some notion of proximity, usually based on the Euclidean metric, as it captures the essence of spatial autocorrelation and spatial association [39]. Bottom-up approaches, in which clusters are formed by agglomeration of items that are 'close' together, are in accordance with the view that in geographical application areas, nearby items have more influence upon each other [91, Chapter 11].

However, top-down approaches to clustering are also of interest [91, Chapter 10]. Instead of composing groups from elements sharing common characteristics, the top-down perspective is one that defines clustering as partitioning a heterogeneous data set into smaller, more homogeneous groups [6, 35]. This approach amounts to identifying regions of elements that are very different from

others in the set. The classical example of this is the k-MEANS heuristic for clustering. k-MEANS attempt to partition the data by assigning each data point to a representative point. The partition optimizes a statistical homogeneity criterion — namely, the total expected squared dissimilarity is minimized.

In this paper we present a *medoid*-based clustering method that incorporates both proximity and density information. Medoid-based clustering [29, 30, 50, 62] is similar to mean-based clustering, but it allows only data points to be chosen as representatives [42]. The use of medoids for clustering results in robustness in the statistical sense [77], as outliers in medoid clustering have less impact on indicators of central tendency than in the optimization problem that is heuristically solved by k-MEANS. This statistical robustness is critically exploited, for example, in one-dimensional (greyscale) data for image analysis in medical domains [36]. There are also fundamental modeling reasons to prefer medoids over means [89].

The first paper to introduce clustering for spatial data mining [62] used the medoid-based approach. Unfortunately, this paper used a random hill-climber strategy which produced poor partitions [30, 59]. Since then, medoid-based methods have been introduced which generally produce clusters of higher quality than k-MEANS and that are robust to outliers, robust to noise and work well with random initialization [30]. (By robust to outliers and noise, we mean the statistical notion of a method working well even in the presence of spurious data. A second form of robustness is in the numerical analysis sense, in which an algorithm is said to be robust if it is not sensitive to perturbations in the inputs or initial conditions. This second type of robustness is particularly important in those situations in which privacy considerations force researchers to slightly modify the data points [78], or when the data is only a close approximation to the ideal values which one wishes to study.)

Nevertheless, there are two main criticisms of medoid-based clustering: if the number n of data points is large, it typically requires $\Theta(n^2)$ time for some hill-climbing iterations, whereas k-MEANS requires only $O(n)$ time per iteration; also, as with k-MEANS, the number of clusters k must be specified in advance. Recently, more efficient and effective hill-climbers have been developed for the medoids approach in the context of spatial data mining [29]. The strategies presented in this paper are further improvements upon these methods.

The medoid-based clustering algorithm that we present requires only expected $O(n \log n)$ time per iteration, and allows for the automatic determination of the number of clusters. Moreover, we achieve this by incorporating proximity information using the Voronoi diagram [69] of the data points and its dual, the Delaunay triangulation. The end result is a partitioning method for clustering that incorporates density and proximity information as in the agglomerative approaches. To our knowledge, this is the first time the bottom-up and top-down philosophies have been combined in this way for clustering. The algorithm is competitive with recently developed proximity methods for clustering large data sets.

The organization of this paper is as follows. Section 2 is a discussion of the advantages and disadvantages of k-MEANS, leading into the presentation of

the medoid-based approach in Section 3. Section 4 details the use of Delaunay triangulations in developing a fast algorithm. Also discussed in this section is the potential of using the Delaunay triangulation to produce a high-quality initial clustering, and for automatic determination of the number of clusters. In Section 5 our method is compared to several others. Section 6 presents an experimental illustration of the robustness of medoid based clustering. We conclude with some final remarks in Section 7.

2 The k-MEANS approach

A distinct characteristic of clustering for data mining applications is the huge size of the data files involved. While the standard hierarchical clustering methods can handle data with numeric and categorical values using complex similarity measures [43, 48], their $O(n^2)$ computational cost makes most of them unacceptable for clustering large data sets [60]. This has prompted data mining researchers [9, 46, 47] to adapt k-MEANS [56] (or Basic Isodata [21]) for efficient processing of large sets with both numeric and categorical attributes.

2.1 The optimization problem

By iteratively improving an initial clustering (perhaps a random clustering), the k-MEANS method produces an approximate solution to the following optimization problem:

$$\text{minimize} \quad M(C) = \sum_{i=0}^{n-1} w_i \, d^2(s_i, \mathsf{rep}[s_i, C]), \tag{1}$$

where

1. $S = \{s_0, s_1, s_2, \ldots, s_{n-1}\}$ is a set of n data items in the m-dimensional real space $\Re^m$;
2. the weight w_u may reflect relevance of the observation s_u, and $d(\vec{x}, \vec{y}) = (\sum_{j=1}^{m} |x_j - y_j|^2)^{1/2}$ is the Euclidean metric;
3. $C = \{c_0, c_1, \ldots, c_{k-1}\}$ is a set of k *centers*, or representative points of $\Re^m$; and
4. $\mathsf{rep}[s_i, C]$ is the closest point in C to s_i; that is,

$$d(s_i, \mathsf{rep}[s_i, C]) = \min_{j \in \{0,\ldots,k-1\}} d(s_i, c_j).$$

Here we focus on the case $m = 2$ (the points s_i are two-dimensional vectors). The partition into clusters is defined by assigning each s_i to its representative $\mathsf{rep}[s_i, C]$. Those data items assigned to the same representative are deemed to be in the same cluster; thus, the k centers encode the partition of the data.

The basic k-MEANS heuristic is a simple one. For each cluster, a new representative is computed by taking a weighted average of the cluster points. Next,

using the new representatives, a new clustering is obtained. These steps are repeated until an iteration occurs in which the clustering does not change.

There are a number of variants of k-Means. These may

- use random or other methods to obtain an initial partition;
- recompute the clustering each time a new representative is computed (the so-called 'combinatorial' reclassification [2]), rather than waiting until the new representatives of all clusters have been determined ('non-combinatorial' reclassification);
- make use of other classification methods.

One variant of k-Means which uses random initialization and non-combinatorial reclassification is Basic Isodata [21]. According to Aldenderfer and Blashfield [2], Basic Isodata was the procedure inside the software package CLUSTAN [92]. Combinatorial reclassification has been favored in Data Mining applications [46, 47].

A third popular variant of k-Means is a combinatorial version in which the update of a representative takes place only if that update would result in a decrease in $M(C)$. This variant goes by several names: hill-climber [2], basic minimum squared error [21], and cluster-swapping [80]. Other variants of k-Means also appear in the literature of vector quantization [13].

2.2 An interpretation of k-Means

All variants of k-Means can be considered a direct simplification of Expectation Maximization (EM), in that they iteratively perform a simplified expectation step and a maximization step. The minimization step of k-Means is the maximization step of EM, when EM is dealing with a mixture of k multivarite normal distributions sharing a known common covariance matrix Σ and the only unknown parameters are the mean vectors $\vec{\mu}_j$ of the components. However, the expectation step is replaced by a classification step that also has some origins in maximum likehood estimation. This classification step simply assigns each observation s_i to the nearest representative c_j. The vector c_j can serve as an estimate for $\vec{\mu}_j$. The underlying intuition follows from the observation that the multivariate normal distribution $N_{\vec{\mu}_j,\Sigma}(\vec{x})$ is large when $\|\vec{x}-\vec{\mu}_j\|^2_{\Sigma^{-1}}$ is small, because the bell-shape of the normal distribution has a peak at $\vec{\mu}_j$ with iso-lines (level contours) defined by the covariance matrix Σ. Thus k-Means can be viewed as an approximation of the squared Mahalanobis distance $\|\vec{x}-\vec{\mu}_j\|^2_{\Sigma^{-1}} = (\vec{x}-\vec{\mu}_j)^{\mathrm{T}}\Sigma^{-1}(\vec{x}-\vec{\mu}_j)$ with the squared Euclidean distance $\|\vec{x}-\vec{\mu}_j\|^2_I = (\vec{x}-\vec{\mu}_j)^{\mathrm{T}}(\vec{x}-\vec{\mu}_j)$, which is easier to compute. The dependence on the squared Euclidean distance (or gravity model) [34] and not simply the Euclidean distance is a source of concern when using k-Means for geographical data [59].

The criterion in (1) is in fact a special case of several based on the *total dispersion matrix* or *total scatter matrix*

$$T = \sum_{i=0}^{n-1} (s_i - \bar{\mu})(s_i - \bar{\mu})^{\mathrm{T}},$$

where $\bar{\mu}$ is the total observed mean vector; that is, $\bar{\mu} = \sum_{i=0}^{n-1} s_i/n$. The criteria occur often in statistics in multivariate analysis of variance, and their motivation derives from the corresponding statistical theory [39]. The entries of T are the sum of squares of cross products determined by the data, and as such do not vary. However, if P is a partition of the data (a clustering), then $T = W(P) + B(P)$ where $W(P)$ is the pooled 'within-group' dispersion matrix and $B(P)$ is the 'between-group' dispersion matrix. Methods more general than k-Means attempt to find a partition that minimizes the size of $W(P)$ (implicitly maximizing the size of $B(P)$). The traditional way to measure size are the trace and the determinant of these matrices. The optimization problem of (1) corresponds to minimizing the trace of $W(P)$. This shows that k-Means favors hyperspherical clusters and that it is sensitive to scaling or similar transformations [2, 21].

2.3 Advantages and disadvantages of k-Means

The attractiveness of k-Means is due to its its computational efficiency. It requires only $O(tmkn)$ time, where t is the number of iterations over the entire data set, m is the dimension, k is the number of clusters, and n is the number of data items. As $t, m, k \ll n$ for data mining applications, in terms of n one may simply describe k-Means as requiring $O(n)$ time.

Despite its efficiency, k-Means variants have other drawbacks well-documented in the literature:

1. From an optimization point of view, it often converges to a local optimum of poor quality.

2. Because k central vectors are means of cluster points, they are commonly adopted as representative of the data points of the cluster. However, it is possible for the average of the coordinates to have no valid interpretation; for example, the average of the coordinates of a group of schools may indicate that the representative school lies in the middle of a lake.

3. k-Means is very sensitive to the presence of noise and outliers, as well as to the initial random clustering [50, page 277]. In particular, much effort has been focused on the sensitivity of k-Means and EM on the set of representatives used to initialize the search [2, 10, 33].

4. The method is statistically biased. For parametric statisticians, this implies that even if provided with the exact number of distributions in a uniform family mixture (for example, all multivariate normal distributions),

and large volumes of data, k-MEANS converges to the wrong parameter values. This has favored other statistical methods such as EM [17]. k-MEANS is also statistically inconsistent. This has favored Bayesian and Minimum Message Length (MML) methods [20, 72, 87] (in fact, Wallace's MML method pioneers these type of alternatives [85]). For an account of these methods and their relation to Minimum Description Length (MDL) [71], see the recent paper of Wallace and Dowe [86]. However, these alternative methods require the user to define a probabilistic model of the classes, and their high sensitivity to the initial random solution has prompted researchers to incorporate initialization mechanisms [33]. Some need to approximately solve NP-hard problems as well, or use dynamic programing algorithms that require $\Omega(n^2)$ time.

3 The k-MEDOIDS problem

Our approach is based on the so called k-MEDOIDS problem [50, 62]. This attempts to solve a variant of the problem shown in (1), with the added restriction that $C \subset S$ (that is, the representatives must be data points), and with no squaring of the the Euclidean distances [50]. More precisely, the problem is:

$$\text{minimize} \quad M(C) = \sum_{i=0}^{n-1} w_i \cdot d(s_i, \mathsf{rep}[s_i, C]), \tag{2}$$

where $C \subset S$. By removing the square we obtain what statisticians refer to as 'least absolute values regression' [77]. The consideration of absolute error is mathematically more difficult and computationally less tractable than the squared error of (1). However, it is statistically more robust [50] and preferred in many geographical situations [89]. Without the restriction $C \subset S$, the case $k = 1$ constitutes the famous Fermat-Weber point problem [34, 57], for which no exact solution is known.

With the restriction $C \subset S$, the optimization problem (2) is also well known to the operations research community as the *discrete p-median problem* [1]. (the operations research literature uses p instead of k for the number of groups). However, the p-median problem is known to be NP-hard. It has a zero-one integer programming formulation [70] with n^2 variables and $n^2 + 1$ constraints, and many heuristic techniques have been developed to for this problem [18, 73, 74, 75, 76, 84]. Lagrangian relaxation has been successfully used for small problems in facility location [16, 61, 90]. While facility location problems may involve perhaps hundreds of points, the p-median problem cannot be expected to be solved optimally for the large number of observations (several thousand or more) that knowledge discovery applications typically involve.

For finding high quality approximate solutions, hill-climbing variations of an interchange heuristic [18, 38, 58, 84] are considered very effective [29, 59, 62]. Other alternatives have been also explored: tabu search [73], genetic algorithms [45, 7, 8, 31, 32], and simulated annealing [58]. The trade-off of effort

versus quality tends to favor hill-climbers over these methods. However, the alternatives occasionally identify solutions which are closer to optimality.

Alternatively, the optimization problem (2) may be amenable to polynomial approximation schemes [5]. These randomized algorithms produce with very high probability a solution that is within a constant factor c from optimality. For example, for the case $m = 2$ there exists an approximation scheme that for any $\epsilon > 0$ produces a solution C with cost $M(C)$ at most $1 + 1/\epsilon$ times the optimum in $O(n^{O(\varepsilon+1)})$ time [5]. While for $\epsilon = 2$ this result seems less effective than the success observed in practice by interchange hill-climbing heuristics, it does offer a guarantee on the quality of the solution.

A heuristic proposed in 1968 by Teitz and Bart [84] is a hill-climber that is regarded as the best known benchmark [44]. It has been remarkably successful in finding local optima of high quality in applications to facility location problems [58, 73], and with some improvements, very accurate for the clustering of large sets of low-dimensional spatial data [29], even in the presence of noise or outliers. We will refer to this heuristic as TB.

The hill-climbing nature of TB local search is clearly revealed if we structure the search space of the k-MEDOIDS problem as a graph with $\binom{n}{k}$ nodes. The nodes of this graph are all sets $C \subset S$ with $|C| = k$, representing a choice of k medoids. The edges of the graph are defined as follows: two nodes C and C' are adjacent if and only if $|C \cap C'| = k - 1$; that is, if they differ in exactly one point.

As every node in the graph represents a feasible solution, we seek to find the node in the graph that minimizes $M(C)$ in (2). The hill-climber interchange heuristic starts at some initial solution C_0, and explores the graph by moving from the current node to one of its neighbors. Letting C_t be the current node at hill-climbing step t, the heuristic examines a set $N(C_t)$ of neighboring nodes of C_t, and considers the best alternative to C_t in this neighborhood: the node $M(C^j) = \min_{C \in N(C_t)} M(C)$. Provided that the new node C^j is an improvement over the old (that is, if $M(C^j) < M(C_t)$), C^j becomes the new current node C_{t+1} for time step $t + 1$. When no better solution is found in the neighborhood $N(C_t)$, the search halts.

The neighborhood $N(C_t)$ is explored in an order influenced by previous failures in the exploration. When searching for a profitable interchange, TB considers the points in turn, according to a fixed circular ordering $(s_0 s_1, s_2, \ldots, s_{n-1})$. When the turn for point s_i arrives, TB verifies whether s_i already belongs to the set of representatives. If so, the point is ignored, and the turn passes to the next point in the circular list, s_{i+1} (or s_0 if $i = n - 1$). If s_i is not a representative, then it is considered for inclusion in the set. The most advantageous interchange C^j of non-representative s_i and representative s_j is determined, over all possible choices of $s_j \in C_t$. Only when $C^j = \{s_j\} \cup C_t \setminus \{s_i\}$ is better than C_t does the set C^j become the new current solution C_{t+1}. Then, the turn passes to the next point in the circular list. When a full cycle through the set of points yields no improvement, a local optimum has been reached, and the search halts.

We now detail the time and space complexity of TB (refer to Figure 1). It is not hard to see that the algorithm requires $O(kn)$ space. Clearly, the

```
1   TB(S, k)
2   begin
3       i ← 0; t ← 0; last_change ← i;
4       repeat
5           if s_i ∉ C_t = {s_{t_0}, s_{t_1}, s_{t_2}, ... s_{t_{k-1}}} then
6               begin
7                   C' ← C_i^{t_j} that minimizes M(C_i^{t_j}) for j = 0, ..., k−1,
                        where C_i^{t_j} = C_t ∪ {s_{t_j}} \ {s_i};
8                   if M(C') < M(C_t) then /* a swap is found */
9                       begin
10                          C_{t+1} ← C'; t++; last_change ← i;
11                      end 12      i ← (i + 1) mod n;
13      until (i = last_change);
14  end
```

Figure 1: Pseudocode for the TB heuristic.

time required to compute $M(C')$ on an adjacent node C' of C is $O(n \log k)$ time. Since C' and C differ on only one point, with appropriate structures for point location, $O(n \log k)$ steps are sufficient to find $\mathsf{rep}[s, C']$ for all $s \in S$ ($\mathsf{rep}[s, C']$ is either unchanged or the new representative s_i). $O(n)$ time suffices to compute $M(C')$ as defined in (2). Therefore, the time required in Line 7 to test representatives of C_t for replacement by s_i is $O(nk \log k)$ time.

Let w be the number of iterations in the repeat loop. By assigning a flag to each element of S the test in Line 5 can be done in $O(1)$ time. The total time of the algorithm is therefore $O(nkw \log k)$ time.

The TB heuristic forbids the reconsideration of s_i for inclusion until all other non-medoid points have been considered as well. The heuristic can be therefore be regarded as a local variant of tabu search [37]. TB's careful design balances the need to explore a variety of possible interchanges against the 'greedy' desire to improve the solution as quickly as possible. Because TB organizes its search in this way, it exhibits better behavior not shared by other hill-climbers [29].

One strength of the TB heuristic is that although $w = O(tn)$ in the worst case, the number of evaluations of M per improvement of C is typically constant [29, 84]. Usually only a constant number of neighbors of C_t are examined for the next interchange. However, in order to terminate, TB requires a complete pass through the data set in which each potential exchange is shown not to improve the value of $M(C)$. Strategies to detect that the last hill-climbing step has been reached have been proposed, but they reduce the time requirements only by a constant factor [29]. Although the TB heuristic is faster than all other known hill-climbers, it nevertheless requires at least $\Omega(n^2)$ time to approximately solve the k-Medoids problem with n data points.

4 Incorporating proximity and improving efficiency

In this section, we present a variant of the k-MEDOIDS approach that requires subquadratic time to terminate. The fundamental idea is that the clustering measure $M(C)$ for k-MEDOIDS should be taken as a guide to clustering only — its optimization is only the means towards that goal. Although evaluating $M(C)$ exactly requires $O(nk \log k)$ time for an arbitrary choice of C, we give a method for approximating $M(C)$ in $O(uk^2 \log k)$ time, after a preprocessing time in $O(n \log n + un \log u)$.

4.1 Fast approximation of the clustering measure

How can $M(C)$ be approximated in time sublinear in n? The idea is to consider only the most important contributions to the sum, according to what $M(C)$ is meant to measure about a clustering.

The minimization of $M(C)$ can be viewed as an attempt to minimize the expected distance between a point and its representative. Minimizing $M(C)$ is equivalent to minimizing $\frac{M(C)}{W}$, where $W = \sum_{i=0}^{n-1} w_i$ is the sum of all weights. Note also that if K_j is the cluster represented by a medoid $\mathsf{rep}[K_j]$, then

$$\begin{aligned} \frac{M(C)}{W} &= \sum_{i=0}^{n-1} \frac{w_i}{W} d(s_i, \mathsf{rep}[s_i, C]) \\ &= \sum_{j=0}^{k-1} \sum_{s_i \in K_j} \frac{w_i}{W} d(s_i, \mathsf{rep}[K_j]). \end{aligned}$$

Here, $\frac{M(C)}{W}$ consists of k terms, each measuring the expected discrepancy (lack of homogeneity) within a cluster. For each cluster, $M(C)$ incorporates the weighted discrepancies between the points in the cluster and their representative.

The purpose of clustering is to identify subsets, each of whose points are highly similar to one another. However, the greatest individual contributions to that portion of $M(C)$ associated with a cluster K_j are made by outliers assigned to K_j, points which exhibit the least similarities to other points, and which often should not be considered to be part of any cluster. Moreover, in the early approximations explored by TB, the set C_t of representatives exhibit more outliers (inappropriate assignments of points to representatives) for those representatives that are not in areas of high density. That is, in the early stages, there are poor representatives in C_t which can be detected because their nearest neighbors are far.

To eliminate the inappropriate contributions of outliers towards the expected discrepancy within clusters, the strategy we adopt is to estimate the expected discrepancy amongst non-outlier points only. To do this, we limit the contributions to $M(C)$ to those points which lie amongst the closest to the medoids of C. Instead of finding a medoid set which best represents the entire set of

points S, we propose that a medoid set be found which best represents the set of points *in its own vicinity*: specifically, the u nearest neighbors of each of the representatives in C.

In order to be able to efficiently determine a set of uk points in the vicinity of a set C_t of medoids, we preprocess the full set of n points as follows:

1. For each point $s_i \in S$, we find its u nearest neighbors.
2. Using this information, we construct the *nearest-neighbor directed graph* [24] $G = (S, E)$ of regular out-degree u and with the set of n points $s_0, \ldots, s_{n-1}$ as nodes. The edge set E consists of those pairs of nodes (s_i, s_j) for which s_j is one of the u nearest neighbors of s_i. The adjacency representation of this regular graph has $O(un)$ size.

During the hill-climbing process, whenever TB evaluates a candidate set C_t of representatives, only uk non-representatives will be examined. First, only those points that are adjacent in the nearest neighbor digraph to some representative $c_j \in C_t$ are ordinarily allowed to contribute to the evaluation of $M(C_t)$. We call this set $P(C)$ of points the *party* of C. To be precise, $P(C) = \{s_i \in S | (c_j, s_i) \in E \text{ and } c_j \in C\}$. However, since two medoids in C_t may share neighbors in the nearest neighbor digraph, the situation may arise where fewer than uk points are evaluated, (i.e. $\|P(C_t)\| < uk$). In order for the hill-climbing process not to be attracted to medoid sets where fewer than uk points are evaluated, two strategies can be applied to pad the number of evaluations out to exactly uk.

The first strategy fills the quota of uk points by randomly selecting from among the remaining points. The second strategy fills the quota from among the points of the proximity graph by repeatedly adding the contribution of the point which is farthest from its representative medoid, as many times as necessary to bring the total number of contributions to $M(C)$ up to exactly uk. More precisely, the point chosen is the one which maximizes the following expression:

$$\max_{\{j | 0 \le j \le k-1 \ \& \ c_j \in C_t\}} \ \max_{\{s_i \in P(C_t) | \ \mathsf{rep}[C_t, s_i] = c_j\}} d(s_i, c_j).$$

In our implementations, we have opted for the latter strategy to assure convergence. Unlike the former strategy, the latter is deterministic, and preserves the hill-climbing nature of TB.

Using the proximity digraph in this way allows us to obtain an approximation $M'(C)$ to $M(C)$ in $O(uk)$ time. Note that our approach does not restrict the candidates for a swap in the search by TB, but rather it achieves its time savings by effectively simplifying the graph for the p-median problem that TB must solve. Restricting swaps in TB on the basis of their distance is common in the statistical literature, but has been shown to result in solutions of poor quality [29, 83].

We complete the description of our approach with the details of the preprocessing we proprose for computing G.

4.2 Delaunay triangulations

Given a set of data points $S = \{s_0, \ldots, s_{n-1}\}$ in the plane, the Voronoi region of $s_i \in S$ is the locus of points (not necessarily data points) which have s_i as a nearest neighbor; that is, $\{x \in \Re^2 | \forall j \neq i, d(x, s_i) \leq d(x, s_j)\}$. Taken together, the n Voronoi regions of S form the Voronoi diagram of S (also called the Dirichlet tessellation or the proximity map). The regions are (possibly unbounded) convex polygons, and their interiors are disjoint.

The Delaunay triangulation $\mathcal{D}(S)$ of S is a planar graph embedding defined as follows: the nodes of $\mathcal{D}(S)$ consist of the data points of S, and two nodes s_i, s_j are joined by an edge if the boundaries of the corresponding Voronoi regions share a line segment.

Delaunay triangulations capture in a very compact form the proximity relationships among the points of S. They have many useful properties, the most relevant to our application being the following:

1. If s_i is the nearest neighbor of s_j from among the data points of S, then (s_i, s_j) is an edge in $\mathcal{D}(S)$. That is, the 1-nearest neighbor digraph is a subgraph of the Delaunay triangulation.

2. The number of edges in $\mathcal{D}(S)$ is at most $3n - 6$.

3. The average number of neighbors of a site s_i in $\mathcal{D}(S)$ is less than 6.

4. The Delaunay triangulation is the most well-proportioned over all triangulations of S, in that the size of the minimum angle over all its triangles is the maximum possible.

5. If s_i, s_j, and s_j form a triangle in $\mathcal{D}(S)$, then the interior of this triangle contains no other point of S.

6. The triangulation $\mathcal{D}(S)$ can be robustly computed in $O(n \log n)$ time.

7. The minimum spanning tree is a subgraph of the Delaunay triangulation, and in fact, a single-linkage clustering (or dendrogram) can be found in $O(n \log n)$ time from $\mathcal{D}(S)$.

8. The u nearest neighbors of a point s_i can be found in $O(u \log u)$ expected time from $\mathcal{D}(S)$ [19]. The algorithm is simple and practical. Place the Delaunay neighbors of $s_i \in S$ in a priority queue with the Euclidean distance to s_i as its key. Repeatedly extract the item with smallest key and place its Delaunay neighbors not already examined in the priority queue (again, the key is the Euclidean distance to s_i). When u items have been extracted, then terminate: these are the u-nearest neighbors.

Figure 2 shows a set of 100 data points and its corresponding Delaunay triangulation. More information regarding Delaunay triangulations and Voronoi diagrams can be found in [64, 68]. Figure 3 shows the clustering obtained with our algorithm on the data shown in Figure 2, using $k = 10$ and $u = 6$. The positions of the 10 medoids are indicated with the $\odot$ symbol.

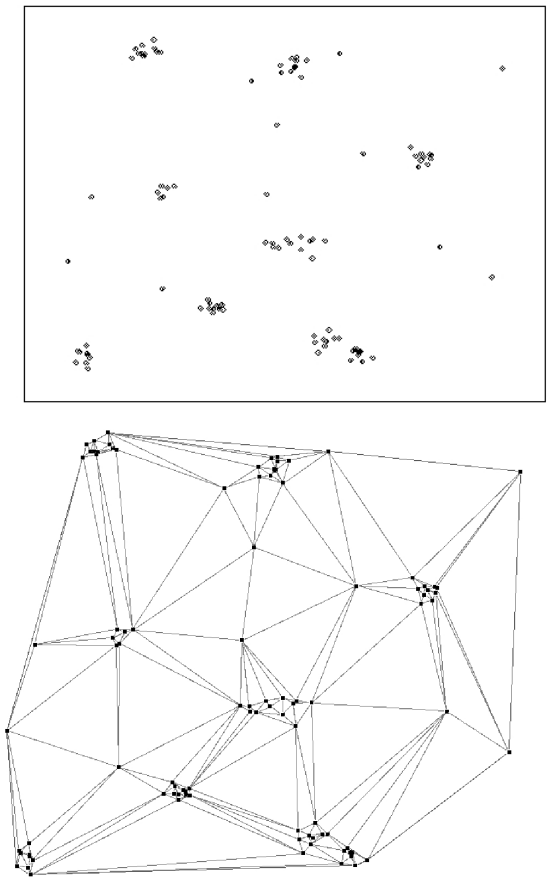

Figure 2: A data set and its Delaunay triangulation.

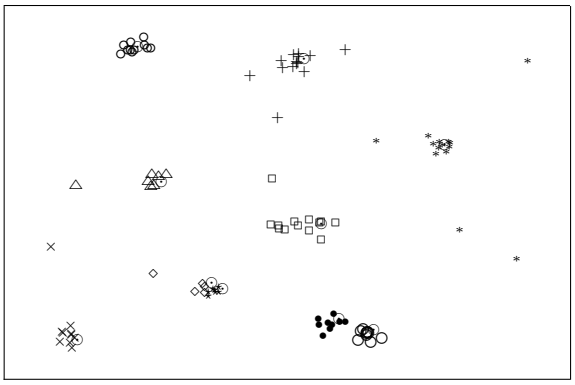

Figure 3: Clustering produced using a proximity graph derived from the Delaunay triangulation.

The total time required to find u neighbors for all s_i is in $O(un \log u)$.

Since $\Theta(n \log n)$ time is required for computing a Delaunay triangulation, setting $u = \Theta(\frac{\log n}{\log \log n})$ allows the nearest neighbor graph G to be constructed in $O(n \log n)$ total time. Thereafter, each evaluation of $M(C)$ would take $\Theta(\frac{k^2 \log k \log n}{\log \log n})$ time. The TB heuristic would then require $O(n \log n + \frac{tk^2 n \log k \log n}{\log \log n})$ time, where t is the number of hill-climbing improvements. Keeping in mind that k and $t \ll n$, if one assumes k and t to be constant, the total time bound simplifies to $O(n \log n)$.

Of course, the user is free to choose larger or smaller values of u. The larger the value of u, the closer the performance becomes to that of the original TB heuristic, and the more time taken. Small choices of u result in very fast execution times, at the cost of a degradation in quality. In practice, the user could base the choice of u according to a time budget.

4.3 Initialization using the Delaunay triangulation

As a result of the experiments described in Section 6, we discovered that the degradation in quality for small choices of u is most evident in the increased sensitivity of the method to the choice of initial set of representatives. One way of dealing with this increased sensitivity is to provide the heuristic with an initial set which is known to be of high quality. For the heuristic to be effective, this initial set must itself be efficiently computed.

The Delaunay triangulation used in the preprocessing of the u-nearest-neighbour graph G can be of use here as well. Efficient methods based on the Delaunay triangulation have recently been proposed for non-representative-based clustering in space [23, 49]. Given a threshold value $\delta > 0$, all edges of the triangulation

with length greater than δ are deleted. If δ is sufficiently large, the remaining edges form a disconnected graph. If a connected component contains at least a minimum number of nodes ν (say, 2 or 3), it can be interpreted as a cluster.

Given a desired value of k, it is possible to determine the threshold value δ which gives rise to k clusters. This can be done as follows:

1. Initialize a set union-find structure, with each data point in its own set.
2. Sort the edges of the Delaunay triangulation in increasing order of length.
3. For each edge in the list, merge the sets containing its endpoints. With every merge, keep track of the number of sets having at least ν nodes.
4. Stop when the number of such sets reaches k.

From each of the k clusters produced, one point may be selected arbitrarily as an initial representative for the modified TB heuristic. The method is efficient, as the time complexity required is dominated by the cost of constructing the Delaunay triangulation, namely $O(n \log n)$.

The technique described above is precisely that used to produce a form of proximity tree called a *dendrogram* [54, 91], which has already seen much use in geographical data analysis. Initially, each data point is associated with its own leaf, and each leaf constitutes a cluster with only one member. Iteratively, the pair of nodes representing the two closest clusters are joined under a new parent node, and their data points together form a new cluster associated with that parent node. The two clusters are deleted from the pool of available clusters, and the new merged cluster is added to the pool. The process terminates when only one cluster remains.

The technique is also essentially the same as the so-called Kruskal algorithm for constructing a Minimum Spanning Tree (MST) [15] — the MST is known to be a subgraph of the Delaunay triangulation, and can even be obtained from the triangulation in $O(n)$ time [69]. The same connected components can be obtained simply by deleting the largest edges of the MST until the desired number of connected components is obtained.

The initial clustering produced using this technique is more than just a convenient byproduct of the use of Delaunay triangulations or dendrograms; although non-representative-based, it is still possible to say something about the quality of the clustering. Let $K = \{K_1, K_2, \ldots, K_{k(\delta)}\}$ be the clustering resulting from any choice of threshold value δ. Let δ_b be the minimum distance between two clusters; that is, $\delta_b = \min_i\{d(s_i, s_j) | \, s_i \in K_i \& s_j \notin K_j\}$. Also, for cluster K_i, consider the partition of K_i which would result in the greatest separation between elements of the two partitions. Let δ_w be the maximum such separation over all clusters. It is easy to see that $\delta_w \leq \delta < \delta_b$. Informally speaking, no gap within a cluster can be greater than the smallest distance between clusters.

Conversely, a suggestion for the number of clusters k can be obtained through an appropriate choice of δ. A profile of threshold values versus number of edges retained can be generated in $O(n \log n)$ time by sorting the edges according to

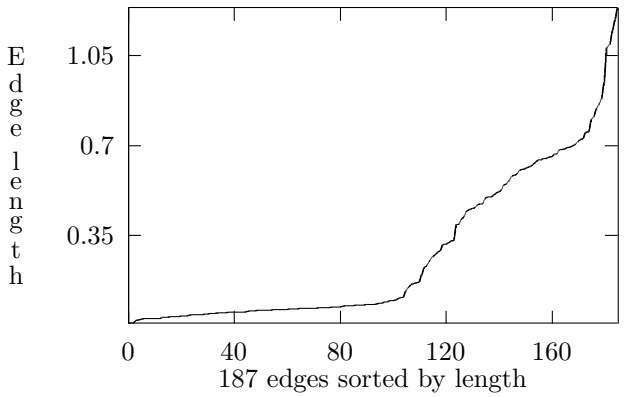

Figure 4: Distribution of edge lengths for the data in Figure 2.

their lengths (see Figure 4 for the profile of the data shown in Figure 2). If clusters are indeed present, the profile would initially rise at a gentle slope to a point or region of inflection, and more steeply thereafter. Picking δ just below the point of inflection would eliminate all of the steep part of the curve, leaving most of the edges joining elements within a cluster. This value of δ could then be used to determine k.

Picking a value of δ significantly below the inflection region can result in the more loosely-associated clusters breaking apart. However, in those situations where the clusters are less fragile, even these low values of δ would be suitable.

5 Comparison and Discussion

We now compare our approach with recent approaches for clustering and for determining an appropriate number of groups.

5.1 On finding clusters in subquadratic time

The first agglomerative algorithm to require $\Theta(n \log n)$ expected time is DBSCAN [27]. The algorithm is regulated by two parameters, which specify the density of the clusters to be retrieved. The algorithm places the data points in an R^*-tree, and uses the tree to perform u-nearest-neighbor queries (usually $u = 4$) to achieve the claimed performance in an amortized sense. An extra $\Theta(n \log n)$ expected time is taken in helping the users determine the density parameters, by sorting all distances between a point and its 4-nearest neighbors, and finding a valley in the distribution of these distances.

DBSCAN presents the user with the profile of distances to 4-nearest neighbors, ranked from farthest to nearest; for the data of Figure 2, this is shown in Figure 5 (a), with an enlarged portion in Figure 5 (b). It is up to the user to de-

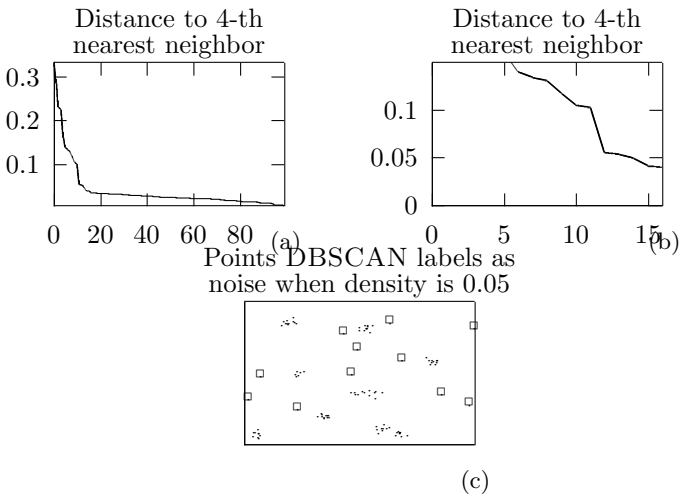

Figure 5: Profile produced by DBSCAN to assist user in selecting density parameter.

cide where in the profile the valley begins. Selecting a density of approximately 0.05 results in the points marked □ in Figure 5 (c) being identified as outliers, and filtered out. With the remaining data, DBSCAN then finds a partition into 9 clusters equivalent to the grouping in Figure 3.

It has been found [88, 93] that determining the parameters of DBSCAN is difficult for large spatial databases. An alternative that does not require input parameters is DBCLASD [93]. Like DBSCAN, DBCLASD can find clusters of arbitrary shape, not necessarily convex; however, DBCLASD is significantly slower than DBSCAN.

The clusters produced by DBSCAN and our approach seem to be of equal quality (see Section 6 for an example). Our approach shares with DBSCAN the interesting feature that it does not require any assumptions or declarations concerning the distribution of the data. We have also found that the implementations of both approaches require roughly the same computational time, once the parameters of DBSCAN have been discovered. Otherwise, DBSCAN must continually ask for assistance from the user.

Another type of clustering method is based on imposing a grid on the data points [14, 82, 81, 88, 94]. The idea is a natural one: when a grid is imposed on the data, those grid boxes containing a large number of data points would indicate good candidates for clusters. The difficulty for the user is in determining the granularity of the grid. Maximum entropy discretization [14] allows for the automatic determination of the grid granularity, but the size of the grid generally grows quadratically in the number of data points. Later, the BIRCH method saw the introduction of a hierarchical structure for the economical storage of grid information, called a Clustering Feature Tree (CF-Tree) [94].

The recent STING method [88] combines aspects of these two approaches. STING constructs a hierarchical data structure whose root covers the region of analysis. In a fashion similar to that of a quadtree [79], each region has 4 children representing 4 sub-regions. However, in STING, all leaves are at equal depth in the structure, and all leaves represent areas of equal size in the data domain. For each node ν, statistical information is computed — namely, the total number t_ν of points that correspond to the area covered by the node ν, the center $\vec{c}_\nu$ of mass (average), the standard deviation $\vec{\sigma}_\nu$, the largest values $\vec{\max}_\nu$, and so on. The structure is built by finding information at the leaves and propagating it to the parents according to arithmetic formulae; for example, the total number of points under a parent node is obtaining by summing the total number of points under each of its children.

STING's data structure is similar to that of a multidimensional database, and thus can be queried by OLAP users using an SQL-like language. When used for clustering, the query proceeds from the root down, using information about the distribution to eliminate branches from consideration. As only those leaves that are reached are relevant, the data points under these leaves can be agglomerated. It is claimed that once the search structure is in place, the time taken by STING to produce a clustering will be sublinear. However, as we indicated earlier, determining the depth of the structure is a challenge.

STING is a statistical parametric method, and as such can only be used in limited applications. It assumes the data is a mixture model and works best with knowledge of the distributions involved. However, under these conditions, methods such as EM [17], AUTOCLASS [12], MML [87] and Gibb's sampling are perhaps more effective for clustering. Recall that both DBSCAN and our approach are non-parametric — we are not learning parameters of a probability distribution nor do we assume their existence.

A second problem with STING is that the grid to be imposed on the data grows very rapidly with the number of dimensions. Thus, with bidimensional points, if the grid is divided into s slabs in each dimension, the time and space requirements of the algorithm are at least $\Omega(s^2)$. This limits the number of slabs to $O(n^{1/2})$, if linear time and storage is desired. This in turn imposes limitations upon the granularity of the cells in the grid. Figure 6 illustrates the bottom level grid constructed using STING on the data set of Figure 2. Our implementation of STING uses $O(n)$ space, as it limits the number of cells at the bottom level to $g = 4^c$, where c is the first integer such that $4^c > n$. Figure 6 also shows the value of the *count* statistic at all levels of the STING hierarchical data structure. The bottom level cells have the total number of points in the cell. Each parent cell contains the sum of its 4 offspring.

The construction of the hierarchical data structure requires $\Omega(n \log n)$ time if there are $\Omega(n)$ leaves, as the construction of any tree by iterative insertion requires time proportional to the external path length of the tree. If a linearization of space is used, then the arithmetic for summation from offspring to parents demands operations on bit patterns of the indices of slabs. This costs $\Omega(\log n)$ per point. The STING construction runs in linear time if the depth is of the structure is restricted to 4 levels [88].

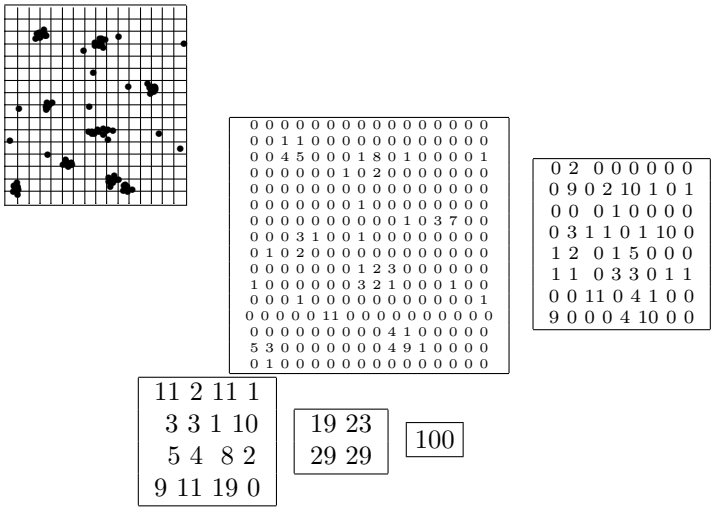

Figure 6: The bottom level grid constructed by STING, and the hierarchy of counts per region.

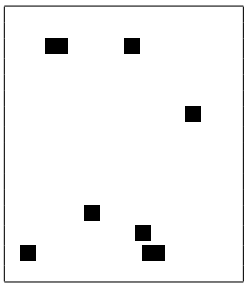

Figure 7: Two-color map produced by STING.

STING finds clusters that are restricted to orthogonal polygons, and its result is a two-color map where disjoint regions are assumed to indicate different clusters. For example, Figure 7 is the result of clustering with STING the data of Figure 2. The results produced by STING are maps of cells in the bottom level grid. The cells are colored black, if in a cluster, or white, if not in a cluster. A statistical test on density is made on bottom level cells to decide whether or not they belong to a cluster. In our example, only cells with a count of 4 or larger are accepted as belonging to a cluster. This is justified by the observation that when $n = 100$ points are distributed among the 256 cells independently and uniformly, the probability that the count of a particular cell equaling 3 is 0.816, while the probability that the count equals 4 is 0.124. Thus, if a cells with a count of 3 were to be accepted, there would be an approximately 82% chance of declaring a cell to be a cluster when it actually is filled with random data. Accepting cells with a count of 4 or larger implies, under this model, a

risk of less than 1 chance in 8 of accepting a random pattern as a group.

In our sample data, only 6 groups are found by STING, although the data set has at least 8 obvious clusters. Many points that are clearly members of clusters are not labeled as such, as they were found themselves in cells with too low a count. This poor clustering reflects many of the problems with this type of statistical approach when attempting to keep the algorithmic time requirements close to $O(n \log n)$. First, the results are not robust with respect to the grid granularity and to the positions of the slabs. The same fine grid, translated by a small amount, may give very different results. Second, the statistical model or assumed probability distributions may not be accurate for the data. Other tests may accept or reject more cells. In our sample data, a cell with a count of 1 besides a cell with a count of 9 should probably be accepted. Incorporating such robust statistical tests that consider spatial autocorrelation would be very useful; however, they would have a serious impact on performance.

STING seems very well suited for an OLAP environment and SQL-type queries specifying several characteristics on the attributes of geo-referenced records. In such settings, an indexing hierarchical structure such as STING's allows for the elimination of many sub-nodes on the basis of attributes values, thereby arriving at a relatively small number of cells at the bottom level where the statistical density tests are performed. In fact, it may be the SQL query itself which specifies the minimum density threshold. However, our interest here is in robust spatial clustering based on geo-reference and proximity.

Another hierarchical approach for clustering two-dimensional points in $O(n \log n)$ time has been presented recently [54], based on dendrograms. Unfortunately, such hierarchical approaches had generally been disregarded for knowledge discovery in spatial databases, since it is often unclear how to use the proximity tree to obtain associations, or to find associations between two proximity trees built from distinct data layers in a GIS [27]. Such approaches do not suggest any agglomerative conditions by which these search trees can be pruned [27].

5.2 On finding the number of groups in subquadratic time

There are two classical approaches to finding the number k of groups: AUTOCLASS [12] and MML [87]. Both demand a declaration of a probabilistic mixture model.

AUTOCLASS [12] searches for the classes using a Bayesian statistical technique. It requires an explicit declaration of how members of a class are distributed in order to form a probabilistic class model. AUTOCLASS uses a variant of EM [17], and thus is a randomized hill-climber similar to k-MEANS or TB, with additional techniques for escaping local maxima. It also labels some data points as noise. Figure 8 presents clusters obtained by AUTOCLASS on the data of Figure 2. AUTOCLASS obtains clusterings of a quality similar to that of k-MEDOIDS, but it requires more CPU time.

Similarly, MML methods [87] require the declaration of a model for which to describe the data, in two parts. The first part is an encoding of parameters of the mixture model; the second is an encoding of the data given the parameters.

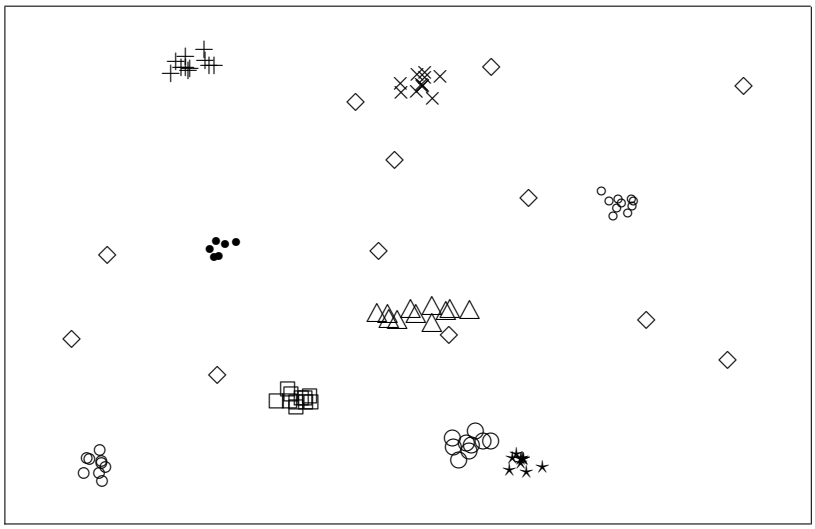

Figure 8: Clustering produced using AUTOCLASS.

There is a trade-off between the complexity of the model and the quality of fit to the data. There are also hard optimization problems that must be solved heuristically when encoding parameters in the fewest number of bits. Figure 9 presents clusters obtained by SNOB [87] on the same data as Figure 2. SNOB obtains only 7 clusters, and requires more CPU time than k-MEDOIDS.

For an approach that does not assume a predefined mixture model, Ng and Han [62] proposed to run CLARANS once for each k, from 2 to n. For each of the discovered clusterings, the *silhouette coefficient* [50] is calculated. The clustering of maximum silhouette coefficient is chosen, determining the number of classes. This is prohibitively expensive for large values of n, since it implies invoking CLARANS $\Theta(n)$ times.

Compared to these methods, our approach is either more generally applicable, in that it does not require the declaration of a mixture model, or far much more efficient, as in the case of the proposal of Ng and Han [62]. Also, as shall be seen in Section 6, our approach does not sacrifice quality of the results.

6 An illustration of the method

In this section we present an experimental illustration contrasting k-MEANS, the original TB, and our algorithms, based on a simple generator of bi-dimensional data sets and mechanisms for adding noise. We first describe this generator.

The generator program takes three input values: the two integer inputs

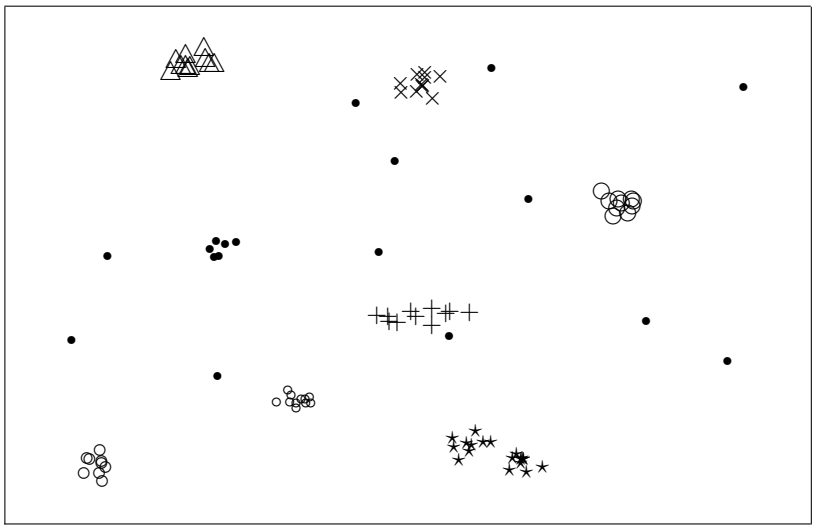

Figure 9: Clustering produced using SNOB.

$k > 0$ and $n > 0$, and a real value $0 \leq \phi \leq 1$. The program produces as output n points within (or very close to) the unit square $[0,1] \times [0,1]$. The points generated are distributed in k natural clusters. The input parameter ϕ designates the proportion of noise — approximately ϕn of the points are simply generated uniformly at random within the unit square. The remaining $(1-\phi)n$ points are designated as cluster data, and are generated within k circles of equal radii. Initially, the centers $\vec{c}_0, \ldots, \vec{c}_{k-1}$ are chosen at random uniformly and independently within the unit square. The program then computes the common circle radius $r = \sigma/2$ from the the separation of the closest pair of centres $\sigma = \min_{i \neq j} d(\vec{c}_i, \vec{c}_j)$. Then, the program repeats n times the following procedure to generate a point in $\Re^2$.

1. It generates a random number ρ uniformly in $[0,1]$.
2. If $\rho \leq \phi$, it outputs a random pair in $[0,1] \times [0,1]$, by uniformly and independently chosing two numbers $x_{i1} \in [0,1]$ and $x_{i2} \in [0,1]$ and outputing $\vec{s}_i^{\mathrm{T}} = (x_{i1}, x_{i2})$.
3. If $\rho > \phi$, the program selects a circle by choosing an index $t \in \{0, \ldots, k-1\}$ uniformly at random, and the real polar coordinates (γ, θ) with $\gamma \in [0, r]$ and $\theta \in [0, 2\pi)$, again uniformly at random. It then outputs the point $\vec{s}_i^{\mathrm{T}} = (x_{i1}, x_{i2}) = \vec{c}_t^{\mathrm{T}} + (\gamma \sin\theta, \gamma \cos\theta)$. $\vec{s}_i^{\mathrm{T}}$ is guaranteed to lie within the tth circle.

Note that if a virtual center $\vec{c}_j$ lies near the boundary of the unit square,

the circle centered at $\vec{c}_j$ with radius $r = \sigma/2$ may intersect the exterior of the square. Also note that each virtual circle is the result of mapping a uniform distribution over $[0, r]$ and $[0, 2\pi)$ around the circle. This results in a non-uniform distribution with a peak at the center of the circle. This simple model illustrates the case of data drawn from a bounded region (the unit square), with cluster distributions cluster distributions within a bounded region (a circle).

This generator can be represented by a mixture model that produces a random vector X taking values in the sample space $\Re^2$. Finite mixture models represent the distribution of such a random variable or vector X (with sample space $\mathcal{X}$) by a probability density function $p(x)$ of the form

$$p(x) = \pi_0 f_0(x) + \ldots + \pi_{k-1} f_{k-1}(x),$$

where $\sum_{j=0}^{k-1} \pi_j = 1$, $\pi_j > 0$, and $f_j(\cdot)$ is probability density functions for $0 \leq j \leq k-1$. The parameters π_j are called the *mixing weights* and the $f_j(\cdot)$ are called the *component densities* of the mixture. If the component densities are parametric, $p(x)$ has the following more explicit representation

$$\begin{aligned} p(x) &= \pi_0 f_0(x|\vec{\theta_0}) + \ldots + \pi_{k-1} f_{k-1}(x|\vec{\theta_{k-1}}), \\ &= \sum_{j=0}^{k-1} \pi_j f_j(x|\vec{\theta_j}) \end{aligned}$$

where $\vec{\theta_j}$ denotes the vector of parameters defining $f_j(\cdot)$.

For the generator presented above we first describe the component densities. We write $U(S)$ for the uniform distribution over the set S. Thus, $U([0,1]\times[0,1])$ is the uniform distribution over the unit square; that is, regions of equal area have equal probability and the unit square has probability one. We write $P(\vec{c}+r)$ for the peak distribution over the circle centered at $\vec{c}$ and radius r. Then, the mixture of GENERATOR produces a random vector $\vec{s}_i^{\mathrm{T}} = (x_{i1}, x_{i2}) \in \Re^2$ with a probability density function given by

$$p(\vec{x}) = \frac{1-\phi}{k} \cdot P(\vec{c}_0 + r) + \ldots + \frac{1-\phi}{k} \cdot P(\vec{c}_{k-1} + r) + \phi \cdot P([0,1] \times [0,1]).$$

While the parametric statistical inference exercise would emphasize the finding of k, ϕ, r and $\vec{c}_i$ for $i = 0, \ldots, k-1$, we focus on recuperating the clusters as generated. The generator program can secretly label each data point s_i either as noise or with the index of the virtual circle used, leaving to the clustering algorithm the task of deciding which points are which. The quality of a partition is the percentage of non-noise points that are labeled correctly by the clustering algorithm. The fewer the mislabeled points, the higher the quality of the partition.

In this setting, we have modeled noise as additive noise, because it appears as an additive term in the finite mixture model. Our experiments compare clustering algorithms over differing levels of additive noise — that is, differing values of ϕ. For this purpose, the generator is modified to initially produce

a data set with no noise, and then randomly delete the replace the desired proportion (ϕn) points with noise drawn from $U([0,1] \times [0,1])$.

Also, for the same data set and level of additive noise, we compare with different levels of multiplicative noise — the term for it will appear as a multiplicative factor in the finite mixture model. A second generator program takes each data point and perturbs it slightly. The idea here is to test the numerical robustness (or robustness to multiplicative noise) of the algorithms. The model by which we perturb each point is to use each point s_i of the data set as a center of a peak distribution $P(\vec{s_i} + \psi r)$ to generate a replacement point no further away that ψr from s_i. The levels of multiplicative noise are regulated by the parameter ψ. The case $\psi = 0$ replaces each point by itself, introducing no multiplicative noise. As ψ grows, the original circle clusters become less dense, and more overlap occurs.

Table 10 shows comparisons of the algorithms for one data set of 300 data items generated by Equation (3). The algorithms compared are k-Means, TB, and our modified TB. The original TB heuristic is initialized always with a random set of representatives, and takes as input the number k of clusters. It represents the desired quality to be achieved by our modified TB.

Our heuristic produces poor results with random initialization when u is small. However, we present experiments in which it is initialized using edge-length clustering to obtain an initial set of representatives. The modified TB is not told the value of k, but instead must automatically determine the number of clusters to use. Despite this handicap, the modified TB performs well in comparison to k-Means.

Since k-Means improves with good guesses of initial representatives, we also compare k-Means using initialization from connected components of a minimum spanning tree, as described in Section 4.3. This results in a version of k-Means that knows the number k of clusters, but now requires $\Omega(n \log n)$ time. Nevertheless, its higher quality makes it a better benchmark than ordinary k-Means.

Each entry of the first table is an average of 10 runs on one data set. The second table shows averages of 100 runs on 10 different data sets. Each average value is presented with a 95% confidence interval.

The results of the experiments indicate that whereas k-Means is competitive with TB and modified TB in the absence of noise, it quickly succumbs as the level of noise rises. On the other hand, modified TB with good initialization matches the quality of the original TB to a very high degree when only additive noise is present. In the presence of multiplicative noise, the modified TB does not perform as well as the original TB, yet significantly better than MST-initialized k-Means.

7 Final remarks

Recently, the identification of clustering as a central task in Knowledge Discovery and Data Mining (KDDM) has attracted researchers to investigate the

One data set 10 runs per set $n = 300$, $u = 6$, $k = 10$		Error in Group Assignment			
		Algorithm			
		k-MEANS		TB	Modified TB
Initialization		Random	MST	Random	(variable k) DT edge length
Noise Mult.	Noise Additive				
$\psi = 0$	$\phi = 0$	39% ±7	8%	0%±0	9%
	$\phi = 0.1$	30% ±4	27%	8%±0	9%
	$\phi = 0.2$	30% ±5	30%	8%±0	11%
$\psi = 0.5$	$\phi = 0$	29% ±5	12%±4	6%±0	17%±4
	$\phi = 0.1$	30% ±4	30%±6	8%±0	14%±4
	$\phi = 0.2$	30% ±5	31%±3	8%±0	14%±4
$\psi = 1.0$	$\phi = 0$	33% ±5	19%±4	8%±0	15%±3
	$\phi = 0.1$	26% ±6	36%±9	9%±1	16%±6
	$\phi = 0.2$	35% ±6	30%±5	11%±1	15%±4
$\psi = 1.5$	$\phi = 0$	34% ±4	22%±3	8%±0	16%±5
	$\phi = 0.1$	30% ±4	31%±6	11%±1	14%±3
	$\phi = 0.2$	34% ±4	30%±5	10%±1	24%±4

10 data sets 10 runs per set $n = 300$, $u = 6$, $k = 10$		Error in Group Assignment			
		Algorithm			
		k-MEANS		TB	Modified TB
Initialization		Random	MST	Random	(variable k) DT edge length
Noise Mult.	Noise Additive				
$\psi = 0$	$\phi = 0$	31% ±4	7%±2	3%±1	10%±2
	$\phi = 0.1$	30% ±4	23%±5	8%±1	10%±2
	$\phi = 0.2$	30% ±5	30%±3	8%±1	11%±2
$\psi = 0.5$	$\phi = 0$	30% ±4	12%±4	6%±1	15%±4
	$\phi = 0.1$	30% ±4	30%±4	8%±1	15%±4
	$\phi = 0.2$	30% ±5	30%±5	8%±1	16%±4
$\psi = 1.0$	$\phi = 0$	31% ±5	20%±4	8%±1	15%±4
	$\phi = 0.1$	30% ±6	32%±4	9%±1	16%±4
	$\phi = 0.2$	31% ±6	30%±5	11%±1	16%±4
$\psi = 1.5$	$\phi = 0$	33% ±4	22%±4	8%±0	15%±5
	$\phi = 0.1$	32% ±4	31%±5	10%±1	16%±4
	$\phi = 0.2$	32% ±4	31%±5	11%±1	25%±4

Figure 10: Comparisons of k-MEANS, TB and modified TB.

scaling of clustering methods to large data sets [10]. The perspective of KDDM on clustering has generally been that of a density estimation problem solved through k-MEANS [9] or EM [33]. In particular, much effort has focused on the sensitivity of k-MEANS and EM on the set of representatives used to initialize the search.

The medoids approach is robust with respect to random initialization, additive noise and multiplicative noise. This comes from considering a loss function or risk function [13] that is not based on the total squared distance (as with k-MEANS), but on the total absolute distance. Heuristics for approximating the medoid clustering require very careful design in order to be applied to large data sets. We have presented a fast heuristic based on the precomputation of the Delaunay triangulation and the u-nearest-neighbour graph.

Researchers have recently identified a set of desiderata for clustering methods [10, 9, 33] with which our medoid approach complies. Namely, the clustering method should be stoppable and resumable, with the capacity to obtain a clustering solution at any time, and to be able to improve on the quality of the solution given more computational resources. Also, since clustering is central to spatial generalization, it has been suggested [88] that clustering methods should find groups directly from the data; this favors the medoid approach over others such as k-MEANS.

Thus, we have presented a clustering method which exhibits the characteristics of density-based clustering (as does DBSCAN), but one which does not demand a density model from the user, and which is robust to outliers and noise.

The medoid approach has limitations in that is fundamentally based around the Euclidean metric, and thus uses the same inductive principle [13] as k-MEANS. It is a representative-based, one-shot partition clustering method. The improvements on time complexity presented here can not be directly extended to three or more dimensions, as the complexity of the size of the Delaunay triangulation is no longer subquadratic. However, we expect that the technique of using the u-nearest neighbour graph can be generalized. Finally, our approach is sensitive to the initial set of representatives. Obtaining an initial set of representatives is itself a clustering problem where we are prepared to trade quality for speed. Our choice of using information from the Delaunay triangulation and the minimum spanning tree may require futher investigation.

8 Acknowledgments

We thank Jack Snoeyink, who pointed out some important references, and the anonymous referees, who suggested many constructive revisions to an earlier version of this paper.

References

[1] P. Agarwal, M. Sharir, and E. Welzl. The discrete 2-center problem. In *Proceedings of the 13th ACM Symposium on Computational Geometry*, pages 147–155. ACM, ACM Press, 1997.

[2] M.S. Aldenderfer and R.K. Blashfield. *Cluster Analysis*. Sage Publications, Beverly Hills, USA, 1984.

[3] M. Ankerst, M.M. Breunig, H-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. *SIGMOD Record*, 28(2):49, 1999. Proceedings of ACM SIGMOD 99 International Conference on Management of Data.

[4] S. Aronoff. *Geographic Information Systems: A Management Perspective*. WDL Publications, Ottawa, Canada, 3rd edition, 1993.

[5] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC)*. ACM Press, 1998.

[6] M.J.A. Berry and G. Linoff. *Data Mining Techniques — for Marketing, Sales and Customer Support*. John Wiley & Sons, NY, USA, 1997.

[7] G. Bianchi and R. Church. A non-binary encoded genetic algorithm for a facility location problem. Working Paper, 1992. Department of Geography, University of California, Santa Barbara.

[8] B. Bozkaya, J. Zhang, and E. Erkut. An effective genetic algorithm for the p-median problem. Paper presented at INFORMS Conference in Dallas, October 1997.

[9] P.S. Bradley and U. Fayyad. Refining the initial points in k-means clustering. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 91–99, San Mateo, CA, 1998. Morgan Kaufmann Publishers.

[10] P.S. Bradley, U. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. In R. Agrawal and P. Stolorz, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 9–15. AAAI Press, 1998.

[11] Y. Cai, N. Cercone, and J. Han. Attribute-oriented induction in relational databases. In G. Piatetsky-Shapiro and W.J. Frawley, editors, *Knowledge Discovery in Databases*, pages 213–228, Menlo Park, CA, 1991. AAAI Press.

[12] P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz. Bayesian classification. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 607–611, Palo Alto, CA, 1988. AAAI, Morgan Kaufmann Publishers.

[13] V. Cherkassky and F. Muller. *Learning from Data — Concept, Theory and Methods*. John Wiley & Sons, NY, USA, 1998.

[14] D.K.Y Chiu, A.K.C. Wong, and B. Cheung. Information discovery through hierarchical maximum entropy discretization and synthesis. In G. Piatetsky-Shapiro and W.J. Frawley, editors, *Knowledge Discovery in Databases*, pages 125–140, Menlo Park, CA, 1991. AAAI, AAAI Press.

[15] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge Massachusetts, 1990.

[16] G. Cornuejols, M. Fisher, and G. Nemhauser. Location of bank accounts to optimize float: An analystic study of exact and approximate algorithms. *Management Science*, 23:789–910, 1977.

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

[18] P. Densham and G. Rushton. A more efficient heuristic for solving large p-median problems. *Papers in Regional Science*, 71:307–329, 1992.

[19] M.T. Dickerson, R.L.S. Drysdale, and J.-R. Sack. Simple algorithms for enumerating interpoint distances and finding k nearest neighbours. *International Journal of Computational Geometry & Applications*, 2(3):221–239, 1992.

[20] D. Dowe, R.A. Baxter, J.J. Oliver, and C. Wallace. Point estimation using the Kullback-Leibler loss function and MML. In X. Wu, R. Kotagiri, and K.K. Korb, editors, *Proceedings of Second Pacific-Asia Conference on Knowledge Discovery and Data Mining PAKDD-98*, pages 87–95, Melbourne, Australia, 1998. Springer-Verlag Lecture Notes in Artificial Intelligence 1394.

[21] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, NY, USA, 1973.

[22] M.J. Egenhofer. Geographic database systems: Issues and research needs. In *Proceedings of the Fifteen ACM-SIGACT-SIGMOD-SIGART Symposium of Principles of Database Systems*, page 80, NY, USA, 1996. ACM, ACM Press.

[23] C. Eldershaw and M. Hegland. Cluster analysis using triangulation. In B.J. Noye, M.D. Teibner, and A.W. Gill, editors, *Proceedings of Computational Techniques with Applications CTAC97*, pages 201–208. World Scientific Singapore, 1997.

[24] D. Eppstein, M.S. Paterson, and F.F. Yao. On nearest-neighbour graphs. *Discrete & Computational Geometry*, 17:263–282, 1997.

[25] M. Ester, A. Frommelt, H.P. Kriegel, and J. Sander. Algorithms for characterization of trend detection in spatial databases. In R. Agrawal and P. Stolorz, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 44–50. AAAI Press, 1998.

[26] M. Ester, H.P. Kriegel, and J. Sander. Spatial data mining: A database approach. In A. School and A. Voisard, editors, *Advances in Spatial Databases, 5th International Symposium, SDD-97*, pages 47–66, Berlin, Germany, 1997. Springer-Verlag Lecture Notes in Computer Science 1262.

[27] M. Ester, H.P. Kriegel, S. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231, Menlo Park, CA, 1996. AAAI, AAAI Press.

[28] M. Ester, H.P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: Focusing techniques for efficient class identification. In M.J. Egenhofer and J.R. Herring, editors, *Advances in Spatial Databases SDD-95*, pages 70–82, Portland, ME, USA, 1995. Springer-Verlag Lecture Notes in Computer Science 951.

[29] V. Estivill-Castro and A.T. Murray. Discovering associations in spatial data - an efficient medoid based approach. In X. Wu, R. Kotagiri, and K.K. Korb, editors, *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-98)*, pages 110–121, Melbourne, Australia, 1998. Springer-Verlag Lecture Notes in Artificial Intelligence 1394.

[30] V. Estivill-Castro and A.T. Murray. Mining spatial data via clustering. In T.K. Poiker and N. Chrisman, editors, *Proceedings of the 8th International Symposium on Spatial Data Handling (SDH-98)*, pages 522–532. International Geographical Union, 1998.

[31] V. Estivill-Castro and A.T. Murray. Spatial clustering for data mining with genetic algorithms. In *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems EIS-98*. CD-ROM ISBN 3-906454-11-8, 1998.

[32] V Estivill-Castro and R. Torres-Velázquez. Hybrid genetic algorithm for solving the p-median problem. In A Yao, R.I. McKay, C.S. Newton, J.-H Kim, and T. Furuhashi, editors, *Proceedings of Second Asia Pacific Conference On Simulated Evolution and Learning SEAL-98*, pages 18–25. Springer Verlag Lecture Notes in Artificial Intelligence 1585, 1999.

[33] U. Fayyad, C. Reina, and P.S. Bradley. Initialization of iterative refinement clustering algorithms. In R. Agrawal and P. Stolorz, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 194–198. AAAI Press, 1998.

[34] R.L. Francis. *Facility layout and location: An analytical approach*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1974.

[35] A.A. Freitas and S.H. Lavington. *Mining Very Large Databases with Parallel Processing*. Kluwer Academic Publishers, London, 1998.

[36] M. Garza-Jinich, P. Meer, and V. Medina. Robust retrieval of three-dimensional structures from image stacks. *Medical Image Analysis*, 3(1):21–35, 1999.

[37] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 5:533–549, 1986.

[38] M Goodchild and V. Noronha. Location-allocation for small computers. Monograph 8, University of Iowa, 1983.

[39] R.P. Haining. *Spatial data analysis in the social and enviromental sciences*. Cambridge University Press, UK, 1990.

[40] J. Han, S. Chee, and J.Y. Chiang. Issues for On-Line analytical mining of data warehouses. In *Proceedings of 1998 SIGMOD-96 Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD-98)*, Seattle, Washington, 1998.

[41] K. Han, J. Koperski and N. Stefanovic. GeoMiner: A system prototype for spatial data mining. *SIGMOD Record*, 26(2):553–556, 1997.

[42] J. Hershberger and S. Suri. Finding tailored partitions. *Journal of Algorithms*, 12:431–463, 1991.

[43] A. Hitchinson. *Algorithmic Learning*. Clarendon Press, Oxford, UK, 1994. Graduate Texts in Computer Science.

[44] M. Horn. Analysis and computation schemes for p-median heuristics. *Environment and Planning A*, 28:1699–1708, 1996.

[45] C. Hosage and M. Goodchild. Discrete space location-allocation solutions from genetic algorithms. *Annals of Operations Research*, 6:35–46, 1986.

[46] Z. Huang. Clustering large data sets with mixed numeric and categorical values. In *Proceedings of the First Asia-Pacific Conference on Knowledge Discovery and Data Mining*, Singapore, 1997. World Scientific.

[47] Z. Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, Tucson, Arizona, May 1997. SIGMOD-DMKD97.

[48] M. Jambu and M-O. Leebaux. *Cluster Analysis and Data Analysis*. North-Holland, Amsterdam, 1983.

[49] I.-S. Kang, T.-W. Kim, and K.-J Li. A spatial data mining method by delaunay triangulation. In *Proceedings of the Fifth ACM Workshop on Geographic Information Systems*, Las Vegas, Nevada, 1997.

[50] L. Kaufman and P.J. Rousseuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, NY, USA, 1990.

[51] E.M. Knorr, R.T. Ng, and D.L.. Shilvock. Finding boundary shape matching relations in spatial data. In A. School and A. Voisard, editors, *Advances in Spatial Databases, 5th International Symposium, SDD-97*, pages 29–46, Berlin, Germany, 1997. Springer-Verlag Lecture Notes in Computer Science 1262.

[52] K. Koperski and J. Han. Discovery of spatial association in geographic information databases. In M.J. Egenhofer and J.R. Herring, editors, *Advances in Spatial Databases — Proceedings of the 4th International Symposium SDD-95*, pages 47–66, Portland, ME, USA, 1995. Springer-Verlag Lecture Notes in Computer Science 951.

[53] K. Koperski, J. Han, and J. Adhikari. Mining knowledge in geographical data. *Communications of the ACM*. to appear.

[54] D. Krznaric and C. Levcopoulos. Fast algorithms for complete linkage clustering. *Discrete & Computational Geometry*, 19:131–145, 1998.

[55] W. Lu, J. Han, and B.C. Ooi. Discovery of general knowledge in large spatial datasets. In *Proceedings of the 1993 Far East Workshop on Knowledge and Data Engineering, FEGIS-93*, pages 275–279, Singapore, June 1993.

[56] J. MacQueen. Some methods for classification and analysis of multivariate observations. In L. Le Cam and J. Neyman, editors, *5th Berkley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967. Volume 1.

[57] Z.A. Melzak. *Companion to concrete mathematics; mathematical techniques and various applications*. John Wiley & Sons, NY, USA, 1973.

[58] A.T. Murray and R.L. Church. Applying simulated annealing to location-planning models. *Journal of Heuristics*, 2:31–53, 1996.

[59] A.T. Murray and V. Estivill-Castro. Cluster discovery techniques for exploratory spatial data analysis. *International Journal of Geographic Information Systems*, 12(5):431–443, 1998.

[60] F. Murtagh. Comments of "Parallel algorithms for hierarchical clustering and cluster validity". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):1056–1057, 1992.

[61] S. Narula, U. Ogbu, and H. Samuelsson. An algorithm for the p-median problem. *Operations Research*, 25:709–713, 1977.

[62] R.T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In J. Bocca, M. Jarke, and C. Zaniolo, editors, *Proceedings of the 20th Conference on Very Large Data Bases (VLDB)*, pages 144–155, San Francisco, CA, 1994. Santiago, Chile, Morgan Kaufmann Publishers.

[63] B-W. Oh, J-K. Yun, and K-J. Han. ASK-ME: A spatial data mining system based on clustering. *Bulletin of the International Rough Set Society*, 3(1/2):11–17, March 1999.

[64] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tesselations - Concepts and applications of Voronoi diagrams*. John Wiley & Sons, NY, USA, 1992. ISBN 0471-93430-5.

[65] S. Openshaw. Two exploratory space-time-attribute pattern analysers relevant to GIS. In S. Fotheringham and P. Rogerson, editors, *Spatial Analysis and GIS*, pages 83–104, London, UK, 1994. Taylor and Francis.

[66] S. Openshaw, M. Charlton, C. Wymer, and A. Craft. A Mark 1 geographical analysis machine for the automated analysis of point data sets. *International Journal of Geographical Information Systems*, 1(4):335–358, 1987.

[67] S. Openshaw and I. Turton. A parallel Kohonen algorithm for the classification of large spatial datasets. *Computers and Geosciences*, 22(9):1019–1026, 1996.

[68] J. O'Rourke. *Computational Geometry in C*. Cambridge University Press, UK, 1994.

[69] F.P. Preparata and Shamos M.I. *Computational Geometry An Introduction*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1985.

[70] C. ReVelle and R. Swain. Central facilities location. *Geographical Analysis*, 2:30–42, 1970.

[71] J. Rissanen. Stochastic complexity. *Journal of the Royal Statistical Society, Series B*, 49(3):223–239, 1987.

[72] J. Rissanen. Fast universal coding with context models. *IEEE Transactions on Information Theory*, 45(4):1065–1071, May 1999.

[73] D. Rolland, E. Schilling and J. Current. An efficient tabu search procedure for the p-median problem. *European Journal of Operations Research*, 96:329–342, 1996.

[74] K. Rosing. An optimal method for solving the (generalized) multi-Weber problem. *European Journal of Operations Research*, 58:414–426, 1992.

[75] K. Rosing, E. Hillsman, and H. Rosing. A note comparing optimal and heuristic solutions to the p-median problem. *Geographical Analysis*, 11:86–89, 1979.

[76] K.E. Rosing, C.S. ReVelle, and H. Rosing-Voyelaar. The p-median and its linear programming relaxation: An approach to large problems. *Journal of the Operational Research Society*, 30:815–823, 1979.

[77] P.J. Rousseeuw and A.M. Leroy. *Robust regression and outlier detection*. John Wiley & Sons, NY, USA, 1987.

[78] G. Rushton and P. Lononis. Exploratory spatial analysis of birth defect rates in an urban population. CD-ROM The University of Iowa, Iowa, US, September 1997. Version 2.0.

[79] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Publishing Co., Reading, MA, 1989.

[80] R.J. Schalkoff. *Pattern Recognition — Statistical, Structural and Neural Approaches*. John Wiley & Sons, Inc., New York, 1992.

[81] E. Schicuta and M. Erhart. The BANG-clustering system: Grid-based data analysis. In *Proceedings of the Second International Symposium IDA-97*. Springer-Verlag Lecture Notes in Computer Science 1280, 1997.

[82] E. Schikuta. Grid: clustering: An efficient hierarchical clustering method for very large data sets. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 2, pages 101–105, 1996.

[83] P. Sorensen. Analysis and design of heuristics for the p-median location-allocation problem. Master's thesis, Department of Geography, University of California, Santa Barbara, 1994.

[84] M.B. Teitz and P. Bart. Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research*, 16:955–961, 1968.

[85] C.S. Wallace and D.M. Boulton. An information measure for classification. *Computer Journal*, 11:185–195, 1968.

[86] C.S. Wallace and D.L. Dowe. Minimum message length and kolmogorov complexity. in press, to apepar, 1999.

[87] C.S. Wallace and P.R. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society, Series B*, 49(3):240–265, 1987.

[88] W. Wang, J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. In M. Jarke, editor, *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 186–195, Athens, Greece, August 1997. VLDB, Morgan Kaufmann Publishers.

[89] C. Watson-Gandy. A note on the centre of gravity in depot location. *Management Science*, 18:B478–481, 1972.

[90] J. Weaver and R. Church. A median location model with nonclosest facility service. *Transportation Science*, 19:107–119, 1985.

[91] R. Webster and M.A. Oliver. *Statistical Methods in Soil and Land Resource Survey*. Spatial Information Systems. Oxford University Press, Oxford, UK, 1990.

[92] D. Wishart. *Supplement, CLUSTAN user manual*. Prigram Library Unit, Edinburgh University, UK, third edition, 1982.

[93] X. Xu, M. Ester, H.P. Kriegel, and J. Sander. A distribution-based clustering algorithm for mining large spatial databases. In *Proceedings of the 14th International Conference on Data Engineering*, pages 324–33, Orlando, FL, February 1998. IEEE, IEEE Computer Society.

[94] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH:an efficient data clustering method for very large databases. *SIGMOD Record*, 25(2):103–114, June 1996. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data.