

Programmable Clustering

Sreenivas Gollapudi
Ebrary Inc.
318 Cambridge Ave.
Palo Alto, CA 94306
sreeni@ebrary.com

Ravi Kumar
Yahoo! Research
701 First Avenue
Sunnyvale, CA 94089
ravikumar@yahoo-
inc.com

D. Sivakumar*
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
siva@google.com

ABSTRACT

We initiate a novel study of clustering problems. Rather than specifying an explicit objective function to optimize, our framework allows the user of clustering algorithm to specify, via a first-order formula, what constitutes an acceptable clustering to them. While the resulting genre of problems includes, in general, NP-complete problems, we highlight three specific first-order formulae, and provide efficient algorithms for the resulting clustering problems.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*Clustering*; I.5.3 [Pattern Recognition]: Clustering—*Algorithms*; F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations*

General Terms

Algorithms, Theory

Keywords

Clustering; First-order formula

1. INTRODUCTION

The problem of clustering a given finite set of points together with a distance function on them has a long and rich history in computer science. A large number of methods have been proposed and analyzed from various view points (convergence, optimization of various objective functions, empirical quality, provable guarantees under various probabilistic models of the input data, etc.). See the textbooks [2, 6] for highlights.

*Part of this work was performed while the author was at the IBM Almaden Research Center.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

PODS'06, June 26–28, 2006, Chicago, Illinois, USA.
Copyright 2006 ACM 1-59593-318-2/06/0006 ...\$5.00.

One theme is common to this body of work on clustering. Typically an objective function is specified that will be maximized or minimized, then (often) an input distribution is fixed, and then a clustering method is formulated (or picked from a set of well-known methods), and finally, the behavior of the clustering method is established via mathematical and/or empirical methods. The objective function is usually chosen based on experience and intuition, as well as knowledge about the domain of objects to be clustered and the input distribution. Several well-known results have been obtained through this approach — e.g., convergence of the k -means algorithm, reconstructing Gaussians from noisy data, and in the realm of distribution-free algorithms, various NP-hardness and polynomial-time approximation algorithms.

Our approach differs fundamentally from this line of work: we propose to study clustering from an orthogonal viewpoint, namely, we will study the question of whether a given set of points can be clustered into a specified number of clusters so that certain categorically explicit properties are satisfied. The kind of properties we envisage are extremely simple to state, and are not intended to be restatements of various objective functions, that is, more axiomatic rather than optimization-based. Thus our approach may be described as “programmable clustering” — clustering problems where we specify, in some formal language, the desiderata to be satisfied by the clustering of a given set of points. A crucial choice here is the language through which we describe the properties desired of the clustering. We propose the use of sentences in first order logic to describe the desiderata for the clustering.

Our work draws its inspiration from several sources. The first one is developments in the field of social-choice theory, where over two centuries ago, Marie Jean Antoine Nicolas Caritat, marquis de Condorcet, and in the 1950s, Kenneth Arrow, established the study of methods for voting and rank aggregation on a firm mathematical footing by employing an axiomatic approach. Namely, Arrow described a very natural set of axioms that are desirable in rank aggregation methods, and proved the rather disappointing result that no such method exists. There has been an amazing sequence of subsequent developments that have led to careful analyses of various desirable criteria for rank aggregation, and whether various aggregation methods satisfy them.

The field of clustering has not taken a similar axiomatic approach. Indeed, the only work in that direction is due to Kleinberg [7], our second source of inspiration, who proposes certain axioms and establishes that no clustering function

satisfies them. He further establishes that reasonable weakenings of these axioms have unique algorithms that satisfy them. The principal difference between Kleinberg’s work and ours is that we do not attempt to define what constitutes a good clustering function — a function that maps a point metric into a partition of the underlying set of points. We are interested in formulating definitions of good clusterings — definitions that help us decide if a given partition of n points in a metric space into k clusters may be considered good. Furthermore, we wish to do so without recourse to any specific objective function.

A third source of inspiration is Gale and Shapley’s notion of stable marriage, which has led to a extensive body of work at the intersection of game theory and computer science. Here again, one seeks a solution that satisfies certain criteria that admits a clean logical expression.

Our final source of inspiration is Fagin’s logical characterization [3] of the complexity class NP, in which he showed that a graph property is in NP if and only if it is expressible as a logical sentence with second-order existential quantifiers and a first-order formula. Fagin’s theorem has led to a deep understanding of NP in logical terms, and has blossomed into the field of descriptive complexity (see [5]). The beauty of Fagin’s theorem rests in the fact that every NP graph problem can be expressed in the form “find a solution structure (stipulated by the second-order quantifier) that satisfies these elementary properties (specified by the first-order formula).” First-order logic is a simple but rather powerful framework to specify properties desired of the solution. We propose to undertake a similarly principled approach to the study of clustering problems.

The main technical contributions of this work are the following. First we formulate a precise definition of what it means to express a clustering criterion in first-order logic. The essential idea is that the criterion should be expressible with the usual connectives and quantifiers of first-order logic, together with access to comparisons like “ $d(u, v) < d(u, w)$,” etc. While first-order logic is quite powerful and expressive enough to describe a variety of clustering methods, it also has the consequence that, in general, the resulting computational problems are NP-hard. We establish this by a simple reduction from 3-colorability. We then introduce a number of natural clustering criteria expressible in first-order logic; typically, these involve comparisons of intra- and inter-cluster distances. The rest of the paper consists of efficient algorithms for the clustering problems corresponding to these criteria. The algorithms we provide are often very simple, and rely on structural properties of the underlying metric space and associated structures like minimum spanning trees.

Besides computational considerations, we raise another important consideration, namely given n points and a parameter k , is there always a clustering of the n points into k clusters that satisfies a specified criterion? While the answer to this, in general, is negative, we are able to identify a first-order expressible clustering criterion for which clustering into k clusters is possible for every value of k ; in this case, we also show that the desired clusterings can be computed in polynomial time. Finally, we study the complexity of the clustering problems we define for the special case of “line metrics,” or metrics that can be embedded on the real line. Since line metrics are structurally simpler, this class of problems serves as an excellent test bed to understand the

difficulty of various notions of clustering, while at the same time turns out to be computationally much more tractable.

2. MODELS AND DEFINITIONS

We assume that the reader is familiar with basic first order logic, so we will not repeat the definitions here. Instead, we will merely recall that a first order logic consists of a vocabulary that specifies the syntactic structure of its constituent objects (relations, functions, etc. on a set); a *model* appropriate for the vocabulary of a first order logic is a set, together with a 1-1 mapping between the syntactic objects specified in the vocabulary and actual functions/relations on the set.

A distance function on a set X is a mapping $d : X \times X \rightarrow \mathbf{R}$ that satisfies the following two properties:

- (1) for every $x, y \in X$, $d(x, y) = 0$ iff $x = y$;
- (2) (symmetry) for every $x, y \in X$, $d(x, y) = d(y, x)$.

If, in addition, it satisfies the following property, then we say it is a *metric*:

- (3) (triangle inequality) for every $x, y, z \in X$, $d(x, z) \leq d(x, y) + d(y, z)$.

All our algorithms for clustering, except for the ones in Section 4, work for general distance measures; the work in Section 4 is specialized to the case of line metrics. However, we will usually state the results for the case of metrics.

DEFINITION 1 (FIRST-ORDER EXPRESSIBLE CLUSTERING).

A clustering criterion \mathcal{C} is a predicate whose arguments are a set X , a distance function $d : X \times X \rightarrow \mathbf{R}$, and a binary relation $C \subseteq X \times X$.

A clustering criterion \mathcal{C} is said to be first-order expressible if:

- (1) There is a first order formula φ whose vocabulary Σ includes a binary relation symbol C , a binary relation symbol $zero$, a 3-ary relation symbol $triangle$ and 4-ary relation symbols $closer$ and $equal$,

- (2) The set X , together with the natural relations $zero_X$, $triangle_X$, $closer_X$, and $equal_X$, defined below, is an appropriate model for Σ :

$$\begin{aligned} zero_X(x, y) &\Leftrightarrow d(x, y) = 0 \\ triangle_X(x, y, z) &\Leftrightarrow d(x, y) + d(y, z) \geq d(x, z) \\ closer_X(u, v, x, y) &\Leftrightarrow d(u, v) < d(x, y) \\ equal_X(u, v, x, y) &\Leftrightarrow d(u, v) = d(x, y) \end{aligned}$$

(Where it is obvious from context, we will drop the subscript X , and, in fact, write using the natural notations such as $d(x, y) + d(y, z) \geq d(x, z)$, $d(u, v) < d(x, y)$, and so on.)

- (3) For every set X of points, every distance function $d : X \times X \rightarrow \mathbf{R}$, and every binary relation (“clustering”) $C \subseteq X \times X$, C satisfies the clustering criterion \mathcal{C} if and only if

$\varphi(X; C, zero_X, triangle_X, closer_X, equal_X)$ is true. (Here, $C(x, y)$ stands for “ x and y are in the same cluster,” that is, the equivalence relation represented by the clustering C . Naturally, whether C is an equivalence relation can be easily expressed in first-order logic.)

Remark. Naturally, the definition is not intended to be a rigid limitation; for example, if one wishes to add additional natural and meaningful relations on X to express certain types of desirable clusterings, one should be allowed to suitably extend the vocabulary of the logic in question. For ex-

ample, one might wish to include for some constant $\delta \in \mathbf{R}$, the binary relation $\text{equal}_X^\delta(x, y) \Leftrightarrow d(x, y) = \delta$.

We now remark that, in general, first-order expressible clusterings are rather powerful; namely, we show that the well-known NP-complete problem of 3-colorability can be expressed as a first-order expressible clustering problem.

PROPOSITION 2. *There is a first-order expressible clustering problem that is NP-complete.*

PROOF. Given a graph $G = (V, E)$, where we wish to know if G is 3-colorable, we express the problem as a clustering problem in the following way. Define a metric d on V by letting $d(u, v)$ be the shortest distance metric on G . Now, if there is a clustering C that satisfies the first-order sentence

$$\Phi \equiv (\forall u, v, w, x)[(u \neq v \wedge u \neq w \wedge u \neq x \wedge v \neq w \wedge v \neq x \wedge w \neq x) \rightarrow (C(u, v) \vee C(u, w) \vee C(u, x) \vee C(v, w) \vee C(v, x) \vee C(w, x))] \wedge (\forall u, v)[d(u, v) = 1 \rightarrow \neg C(u, v)]$$

expresses the conditions “for every set of four distinct points, at least two must be in the same cluster” (equivalently, there are no more than three clusters) and “for every edge (u, v) , the end points belong to different clusters,” which together imply that G is 3-colorable. Therefore, deciding if a clustering C that satisfies Φ exists is equivalent to deciding if G is 3-colorable. \square

Let $X = \{x_1, \dots, x_n\}$ denote a set of n points. Let $d : X \times X \rightarrow \mathbf{R}$ denote a distance function so that (X, d) is a metric space. Let C denote a partition of X into equivalence classes called *clusters*; for $x, y \in X$, we denote by $C(x, y)$ the predicate that is true iff x and y are in the same cluster of C . Sometimes, abusing notation, we denote by $C(x)$ the cluster that x belongs to. For $x \in X$ and $r \in \mathbf{R}$, we denote by $B(x, r)$ the open ball of radius r around x , that is, $B(x, r) = \{y \in X \mid d(x, y) < r\}$.

DEFINITION 3 (GLOBALLY WELL-SEPARATED). *A clustering C of the metric (X, d) is globally well-separated if for every $u, v, x, y \in X$, if $C(u, v)$ and $\neg C(x, y)$, then $d(u, v) < d(x, y)$. In other words, every intra-cluster distance is strictly less than every inter-cluster distance.*

DEFINITION 4 (LOCALLY WELL-SEPARATED). *A clustering C of the metric (X, d) is locally well-separated if for every $x, y, z \in X$, if $C(x, y)$ and $\neg C(x, z)$, then $d(x, y) < d(x, z)$. In other words, no point is closer to a point from a different cluster than it is to any point from its own cluster.*

It is easy to see that Definitions 3 and 4 are first-order expressible. It is more or less obvious from the definitions that every globally well-separated clustering is also locally well-separated. It is also easy to see that for any n and k , there are instances of clustering problems that do not have any clustering globally well-separated or locally well-separated clustering into k clusters. Motivated by this, we attempt to further weaken the notion of locally well-separated.

DEFINITION 5 (LOCALLY WELL-SUPPORTED). *A clustering C of the metric (X, d) is weakly locally well-supported if for every x and every radius $r \leq \text{diameter}(C(x))$ and every cluster $C' \neq C(x)$, we have $|B(x, r) \cap C'| \leq |B(x, r) \cap C(x)|$. In other words, every ball (of small enough radius) around*

x contains more points from $C(x)$ than from any other cluster, or equivalently, the plurality of the points in every ball (of small enough radius) around x belongs to $C(x)$. (NB: This definition allows for the possibility that for some $r \leq \text{diameter}(C(x))$, $|B(x, r) \cap C(x)| < |B(x, r) \setminus C(x)|$, or equivalently, it is possible that the majority of the points in some ball around x does not belong to $C(x)$.)

Once again, it is obvious from the definitions that every locally well-separated clustering is also locally well-supported. However, locally well-supported clustering is probably not first-order expressible (since it involves counting, which is known not to be first-order expressible). Therefore, we weaken it even further to arrive at the next two definitions, which are clearly first-order expressible.

DEFINITION 6 (NON-SINGULAR CLUSTERING). *A clustering C of the metric (X, d) is said to admit singular points if for some $x \in X$, there is a $z \in X$ such that $\neg C(x, z)$ and for every $y \in X, y \neq x$ such that $C(x, y)$, we have $d(x, z) < d(x, y)$. A clustering C of the metric (X, d) is non-singular if it does not admit singular points.*

DEFINITION 7 (NEAREST-NEIGHBOR SUPPORTED). *A clustering C of the metric (X, d) is nearest-neighbor supported if for every $x \in X$, either x occurs as a singleton in C or there is a $y \in X$ such that y is a nearest neighbor of x (that is, $d(x, y) \leq d(x, z)$ for every $z \in X$) and $C(x, y)$. In other words, every point is in a cluster of its own, or at least one of its nearest neighbors is in the same cluster.*

PROPOSITION 8. *A clustering C of (X, d) is non-singular iff it is nearest-neighbor supported.*

DEFINITION 9 (UNIQUE NEAREST NEIGHBOR METRIC). *A metric (X, d) is said to be a Unique Nearest Neighbor Metric if for every $x \in X$, there is a unique $y \in X, y \neq x$ such that for all $z, d(x, y) < d(x, z)$.*

3. ALGORITHMS

In this section we describe the basic algorithms to find k clusters under various notions of clusterings expressible in first order logic. All our algorithms take a parameter k , which specifies the exact number of desired clusters. The input to our algorithms will be a metric space (X, d) ; in fact, our algorithms will only use only the symmetry property (i.e., *equal*) of the metric space.

We first describe the algorithm to find k globally well-separated clusters, if such a clustering exists for the given k . Next, we describe two algorithms to find k locally well-separated clusters, if such a clustering exists. Finally, we present an algorithm to find a nearest neighbor supported clusters. Unlike the previous two, this clustering exists for every k . We also present a simpler and more efficient algorithm for this case, if the metric satisfies some reasonable uniqueness properties; this case is interesting because such uniqueness properties can be achieved by a tiny perturbation of the given metric space.

3.1 Globally well-separated clusters

Our first algorithm is to find k globally well-separated clusters, if it exists. The basic idea behind the algorithm is to identify a global threshold that essentially determines whether such a clustering exists or not.

THEOREM 10. *For any metric (X, d) and integer k , $1 \leq k \leq n$, it can be decided in $O(n^2 \log n)$ time whether there is a clustering of (X, d) into k clusters that is globally well-separated.*

PROOF. Given a number $\tau > 0$, define the graph G_τ as follows. The vertices of G_τ correspond to the points in X and for every pair $x, y \in X$, the edge (x, y) is present in G_τ if and only if $d(x, y) < \tau$.

We first examine the structure of a globally well-separated clustering, if one exists. Let X_1, \dots, X_k be the clusters of X . Let $d_\ell = \max_{x, y \in C(x, y)} d(x, y)$, i.e., d_ℓ is the largest intra-cluster distance and let $d_u = \min_{x, y \in -C(x, y)} d(x, y)$, i.e., d_u is the smallest inter-cluster distance. Since the clustering is globally well-separated, it follows $d_\ell < d_u$. Let $\tau \in (d_\ell, d_u)$ and consider the graph G_τ . We claim that G_τ is a disjoint union of k cliques. This claim follows since for each x, y such that $C(x, y)$, we have $d(x, y) \leq d_\ell < \tau$ and therefore, the edge (x, y) is present in G_τ ; thus, the vertices corresponding to each cluster form a clique in G_τ . Likewise, since the inter-cluster distances are at least $d_u > \tau$, we have that for every x, y such that $-C(x, y)$, there is no edge (x, y) in G_τ . Thus, there are no edges in G_τ between the k different cliques. The claim follows.

Let $d_1 \leq d_2 \leq \dots \leq d_s$ be the ordered list of distances in X , where $s = \binom{n}{2}$. If X has a globally well-separated clustering, then by the above claim, there is a τ such that G_τ is a union of k cliques. If such a τ exists, then it can be found by examining all the s different intervals in the above ordered sequence. Given a candidate τ , constructing G_τ and testing if it is a union of k different cliques can be done in $O(n^2)$ time.

Note that if $\tau < \tau'$, then G_τ is subgraph of $G_{\tau'}$. Therefore, a binary search on the above ordered sequence will identify the correct τ that will correspond to the globally well-separated clustering, if one exists. This makes the total running time $O(n^2 \log n)$. \square

3.2 Locally well-separated clusters

In this section we present two algorithms to find locally well-separated clusters, if such a clustering exists for the given k . The first algorithm runs in time $O(n^k)$; this algorithm is based on a structural characterization of locally well-separated clusters. The second algorithm is more efficient and runs in time $O(n^2 2^k)$; this is based on a more careful analysis of the structure of locally well-separated clusters.

THEOREM 11. *For any metric (X, d) and integer k , $1 \leq k \leq n$, it can be decided in time $n^k \text{poly}(n)$ whether there is a clustering of (X, d) into k clusters that is locally well-separated.*

PROOF. We first examine the structure of any locally well-separated clustering with k clusters, assuming such a clustering exists. Let X_1, \dots, X_k be the clusters and for each i , let $x_i \in X_i$ be an arbitrary point in the i -th cluster. Now, consider any $x'_i \in X_i$ and any $x_j \in X_j$ where $j \neq i$. Since $-C(x_i, x_j)$, i.e., x_i and x_j belong to different clusters, by the property of locally well-separated clustering, we have $d(x'_i, x_i) < d(x'_i, x_j)$. In other words, each point in the i -th cluster is strictly closer to x_i than to any other point in any other cluster. Therefore, by assigning every $x \in X$ to the x_i that is the closest, we can reconstruct X_i . Note that this step is not well defined if a locally well-separated clustering

does not exist. Thus, the set of points $\{x_1, \dots, x_k\}$ induces the locally well-separated clustering, if one exists.

To find the locally well-separated clustering, we first enumerate all possible subsets of k points. For each such subset $\{x_1, \dots, x_k\}$, we assign points in X by assigning every $x \in X$ to the x_i that is the closest to x . We then check if the resulting clustering is locally well-separated. If a locally well-separated clustering exists, one of the subsets enumerated is guaranteed to hit each of the k clusters and by the above observation, such a clustering will be found. Clearly, this algorithm can be implemented in time $O(n^k)$. \square

We now obtain a different algorithm that runs in time $O(n^2 2^k)$. This is possible by a more careful analysis of the structure of locally well-separated clustering in terms of the minimum spanning tree of the metric space.

THEOREM 12. *For any metric (X, d) and integer k , $1 \leq k \leq n$, it can be decided in time $O(n^2 2^k)$ whether there is a clustering of (X, d) into k clusters that is locally well-separated.*

PROOF. We first obtain a strong structural characterization of a locally well-separated clustering, assuming it exists. Let T be a minimum spanning tree of the metric space (X, d) and let X_1, \dots, X_k be the k clusters. We claim that for each i , the induced subgraph T_i of T on X_i is connected. To see this claim, assume the contrary. That is, there is a T_i that is not connected. Now, we identify three points $x_i, x'_i, x''_i \in X_i$ with the following three properties:

- (i) x_i and x'_i are in two different components of T_i ;
- (ii) there is a path in T_i from x_i to x''_i ;
- (iii) there is an x_j such that (x''_i, x_j) where x_j does not belong to X_i .

It is easy to see that three such points always exists. Now, consider x'_i, x''_i, x_j . By choice, we have $C(x''_i, x'_i)$ and $-C(x''_i, x_j)$. By the locally well-separated property, this implies $d(x''_i, x'_i) < d(x''_i, x_j)$. Therefore, the tree $T' = T \cup \{(x''_i, x'_i)\} \setminus \{(x''_i, x_j)\}$ has weight less than T , contradicting the minimality of T .

Therefore, if a locally well-separated clustering exists, then it can be constructed using the minimum spanning tree T by removing exactly $k - 1$ edges from the tree T . The resulting k subtrees will correspond to locally well-separated clusters. Now, the only remaining task is to find which $k - 1$ edges in T have to be removed.

We now claim that if a locally well-separated clustering exists, then the heaviest edge in T will not be present inside any cluster, i.e., the heaviest edge in T cannot be an intra-cluster edge. The proof is by contradiction. Let (x_i, x'_i) be the heaviest edge in T and let $x_i, x'_i \in X_i$. Now, consider an x''_i with the following properties:

$$x''_i \in X_i; \quad (x''_i, x_j) \in T; \quad x_j \notin X_i.$$

Clearly, such an x''_i always exists. We now have

$$d(x_i, x''_i) \geq d(x_i, x'_i) \geq d(x''_i, x_j), \quad (1)$$

where the first inequality follows since T was the minimum spanning tree and the second inequality follows since (x_i, x'_i) was the heaviest edge in T . Thus, we have x_i, x''_i, x_j such that $C(x''_i, x'_i)$ and $-C(x''_i, x_j)$. Once again, by the locally well-separated property, we get $d(x_i, x''_i) < d(x''_i, x_j)$, which contradicts (1).

Removing the heaviest edge in T splits the tree into two subtrees T_1 and T_2 , which implicitly partitions the clusters X_1, \dots, X_k into two subsets S_1 and S_2 . Note that we do not know the sizes of the subsets S_1 and S_2 . An argument similar to the earlier can be made with respect to T_1 (resp, T_2) and S_1 (resp, S_2) to show that the heaviest edge in T_1 (resp, T_2) cannot be present in any cluster in S_1 (resp, S_2). Recursively proceeding to $k - 1$ levels, we can enumerate all possible decompositions of T into k subtrees where each subtree is connected; this can be done in time 2^k . As noted earlier, the points of X in each connected subtree correspond to the clusters.

So, by building the minimum spanning tree and applying the above observations, we obtain 2^k candidate clusterings, each with k clusters. We can then check if any of these is locally well-separated. \square

3.3 Nearest neighbor supported clusters

In this section we present an algorithm for finding nearest neighbor supported clusters. Unlike the previous two cases, such a clustering is guaranteed to exist for every k . We also present a more efficient algorithm when the metric space satisfies certain uniqueness properties.

THEOREM 13. *For any metric (X, d) and integer k , $1 \leq k \leq n$, a clustering of (X, d) into k clusters that is nearest neighbor supported can be found in time $O(n^2k)$.*

PROOF. We start with an arbitrary partition of X into k non-empty pieces X_1, \dots, X_k . The main idea is to take this arbitrary partition and fix all the violations of the requirements of the nearest neighbor supported clustering. For each point $x \in X$, call x ‘happy’ if x is either a singleton or at least one of the nearest neighbors of x is in the same partition as x ; call x ‘unhappy’ otherwise. It is easy to see that if each $x \in X$ is happy, then the partition is a legal nearest neighbor clustering with k clusters. The goal now is to achieve this state by identifying each unhappy x and making it happy.

Let x be an unhappy point and let $x \in X_i$. By definition, x is not a singleton and none of its nearest neighbors is in X_i . Let y be an arbitrary nearest neighbor that is in X_j , $i \neq j$. We move x from X_i to X_j ; clearly this will make x happy. Unfortunately, this might make some other points z_1, \dots, z_ℓ unhappy since they are in X_i and x might have been their only nearest neighbor that was also in X_i . We now make each of z_1, \dots, z_ℓ happy by moving them to X_j . Again, this might create further unhappy points and we continue to do the same. The important point to note is that once a point becomes happy midway through this process, it will continue to stay happy. Since there are only finitely many points, this process eventually terminates. At the end of this process, since x , who was originally unhappy, became happy and the happiness of other points is preserved, the total number of unhappy points decreases.

We repeat this process until there are no unhappy points. Now using the first observation, we obtain a nearest neighbor supported clustering with k clusters. It can be seen that the running time of this algorithm is $O(n^2k)$. \square

3.3.1 Unique nearest neighbor metric

In this section we present the algorithm for finding nearest neighbor supported clusters, if the metric satisfies certain uniqueness condition, defined below.

DEFINITION 14 (UNIQUE NEAREST NEIGHBOR METRIC). *A metric (X, d) is said to be a Unique Nearest Neighbor Metric if for every $x \in X$, there is a unique $y \in X$, $y \neq x$ such that for all z , $d(x, y) < d(x, z)$.*

As we mentioned earlier, for unique nearest neighbor metrics, we obtain a simpler and more efficient algorithm. This algorithm is interesting because it is possible to perturb any given metric (X, d) to obtain (X, d') such that the latter is a unique nearest neighbor metric and d' and d are close for every pair of points in X . Alternately, this version can be viewed as a ‘smoothed’ analysis.

THEOREM 15. *For any unique nearest neighbor metric (X, d) and integer k , $1 \leq k \leq n$, it can be decided in $O(n^2)$ time whether there is a clustering of (X, d) into k clusters that is nearest neighbor supported.*

PROOF. Define the natural directed graph G_N where the vertices are the points in X and the directed edge (x, y) exists if and only if y is the nearest neighbor of x .

Let X be a unique nearest neighbor metric and consider G_N . By the uniqueness property of X , each vertex in G_N has out-degree of exactly one.

First, we claim that G_N consists of vertex disjoint ‘loop’, where a ‘loop’ is a sequence of vertices x_1, \dots, x_ℓ , where (x_i, x_{i+1}) is an edge in G_N for $1 \leq i < \ell$ and (x_ℓ, x_1) is also an edge in G_N . To see this claim, start a deterministic walk at x_1 , following the unique out-neighbor at each step. The only way this walk can terminate is when it ends up at a vertex already encountered in the path, i.e., it produces a directed cycle at the end. Thus, all we need to prove is that G_N can have only trivial cycles, i.e., cycles of length 2. Suppose y_1, \dots, y_m is a cycle in G_N , meaning y_{i+1} is the nearest neighbor of y_i for $1 \leq i < m$ and y_1 is the nearest neighbor of y_m . The only way this can happen is if $d(y_1, y_2) > d(y_2, y_3) > \dots > d(y_{m-1}, y_1) = d(y_1, y_{m-1})$, where the equality follows from the symmetry of d . But, this is a contradiction since y_2 was supposed to be the nearest neighbor of y_1 . Hence, $m = 2$.

It is easy to see that each loop defines a cluster and such a cluster can be a legal member of any nearest neighbor supported clustering. Moreover, it is also easy to see that a union of two loops also produces a cluster that can be a legal member of any nearest neighbor supported clustering. Finally, if x_1, \dots, x_ℓ is a loop, x_1 by itself as a singleton and x_2, \dots, x_ℓ can both be legal members of any nearest neighbor supported clustering.

Let k' be the number of loops in G_N . Depending on whether k' is larger or smaller than k and using the above observation, it is easy to see that we can obtain a nearest neighbor supported clustering with exactly k clusters. The running time of this algorithm is $O(n^2)$, where the majority of the time is spent in constructing the graph G_N . \square

4. ALGORITHMS FOR LINE METRICS

In this section, we consider the clustering problems we have introduced for the special case of line metrics.

DEFINITION 16 (LINE METRIC). *A metric space (X, d) is said to be a line metric if for every three distinct points $x, y, z \in X$ exactly one of the following holds: $d(x, y) + d(y, z) = d(x, z)$ or $d(x, z) + d(y, z) = d(x, y)$ or $d(x, y) + d(x, z) = d(y, z)$.*

PROPOSITION 17. *A finite metric space (X, d) is a line metric if and only if it can be embedded isometrically (that is, with no distortion) on the real line.*

Proof omitted.

Line metrics are one of the simplest kind of metrics; they are simultaneously ℓ_1 and ℓ_2 metrics. Furthermore, they arise naturally in many applications involving Euclidean metrics in higher dimensions, via the process of random projections. Indeed, when a set of points in real Euclidean space (of any dimension) is projected onto a random line (where a random line is picked by picking a random vector, each of whose coordinates is drawn from the standard normal distribution $N(0, 1)$), for any two points u and v with projections $\pi(u)$ and $\pi(v)$, respectively, it is easy to see that $E[(\pi(u) - \pi(v))^2] = \|u - v\|_2^2$. Thus to perform clustering in high-dimensional Euclidean space, one may project points onto various random lines, perform 1-dimensional clustering (satisfying criteria of one's choice), and then "lift" them into a clustering in the original space via techniques like consensus clustering [4, 1].

THEOREM 18. *Given a line metric (X, d) and integer k , $1 \leq k \leq n$, it can be decided in $\text{poly}(n)$ time whether there is a clustering of (X, d) into k clusters that is globally well-separated, locally well-separated, locally well-supported, or nearest-neighbor supported.*

PROOF. For the case of line metrics, an important simplification occurs for clustering problems in that we only need to choose $k - 1$ "gaps" between "adjacent" points on the line. Formally, we say two points of X are *adjacent* if, in their embedding on the real line, there is no other point between them. Note that in any clustering of a line metric into k clusters, there are exactly $k - 1$ inter-cluster distances between adjacent points, and it suffices to identify these to fully specify the clustering.

Globally well-separated clustering As described in Section 3, there is a simple algorithm that decides if (X, d) can be partitioned into k clusters that are globally well-separated; the running time of that algorithm is $O(n^4 \log n)$. Here we describe a greedy algorithm with running time $O(n \log n)$ for the case of line metrics.

The key ideas behind the algorithm are the following. Let A denote the set of $n - 1$ distances between adjacent points on the line. For simplicity, let us assume that the $k - 1$ largest values in A are well-defined, that is, there is no ties in choosing the $(k - 1)$'s largest value; the general case is similar, but clumsier to describe. We claim that every globally well-separated k -clustering of X must designate the $k - 1$ largest distances in A as inter-cluster distances. (In the general case with ties, the claim is that if there is a globally well-separated clustering of X into k clusters, then there is one where the $k - 1$ largest pairwise distances in A are inter-cluster distances.) For if not, there will be a pair of adjacent points, say u and v , in the same cluster, and a pair of adjacent points, say x and y , in different clusters, such that $d(u, v) > d(x, y)$, which is inadmissible. The algorithm is now obvious: sort the $n - 1$ distances between adjacent pairs of points, and see if the clustering that results from designating the $k - 1$ largest distances among adjacent points as inter-cluster distances satisfies the condition for globally well-separated. If it is, then we have found such a clustering;

if not, none exists by the argument just outlined. The time to perform this check is $O(n)$, and the initial sorting step takes $O(n \log n)$ time.

Locally well-separated clustering For the case of locally well-separated clustering into k clusters, we employ a dynamic programming based algorithm. For integer m , let $[m]$ denote the set $\{1, \dots, m\}$. Also, wlog., we will refer to the points in X as x_1, \dots, x_n , with the ordering $x_1 < x_2 < \dots < x_n$. For convenience, we define $x_0 = -\infty$ and $x_{n+1} = \infty$. For $i < j$, we will denote the cluster of points consisting of x_i, \dots, x_j by $[i, j]$. We define the binary predicate *viable* : $[n] \times [k]$, defined by

$\text{viable}(j, t) \Leftrightarrow$ there is a clustering of the set of points x_1, \dots, x_j into exactly t clusters that are locally well-separated and the rightmost cluster $[r, j]$ satisfies $x_j - x_r < x_{j+1} - x_j$.

The definition is motivated by the following observation. For any cluster $[i, j]$ in any clustering of a line metric, the maximum intra-cluster distance within this cluster is $x_j - x_i$ and the minimum inter-cluster distance involving this cluster is either $x_i - x_{i-1}$ or $x_{j+1} - x_j$. Therefore, for $[i, j]$ to be a viable cluster in a locally well-separated clustering of X , the only condition that matters is whether $x_j - x_i$ is smaller than both these quantities.

The predicate *viable* admits a recursive decomposition:

- (1) $\text{viable}(1, 1) = \text{true}$;
- (2) $\text{viable}(j, 1) = \text{true}$ for if $x_j - x_1 < x_{j+1} - x_j$;
- (3) $\text{viable}(j, t) = \text{true}$ for $1 < j \leq n$ and $1 < t \leq k$ if

$$(\exists i, 1 \leq i < j) [\text{viable}(i, t - 1) \wedge x_j - x_{i+1} < \min\{x_{i+1} - x_i, x_{j+1} - x_j\}].$$

The last, recursive, step essentially says that we have a viable clustering of x_1, \dots, x_j into t clusters if for some $i < j$, we have a viable clustering of x_1, \dots, x_i into $t - 1$ clusters (such that its rightmost cluster has no violation with respect to the distance $x_{i+1} - x_i$) and, in addition, if we group x_{i+1}, \dots, x_j into a cluster of points, its largest intra-cluster distance $x_j - x_{i+1}$ produces no violation with respect to the distances $x_{i+1} - x_i$ and $x_{j+1} - x_j$.

Now it is easy to see that this recursion, which fills a table of size nk , each in time $O(n)$, leads to an $O(n^2k)$ time algorithm.

Nearest-neighbor supported clustering For clustering a line metric into nearest-neighbor supported clustering into k clusters, we once again adopt a dynamic programming approach.

We say that a point x in a clustering is *satisfied via* y if y is a nearest neighbor of x and belongs to the same cluster as x . We say that a point x in a clustering is *satisfied* if it is satisfied via some y .

We define the ternary predicate *viable* : $[n] \times [k] \times \{\text{small}, \text{large}\}$, defined by

$\text{viable}(j, t, \text{small}) \Leftrightarrow$ there is a clustering of the set of points x_1, \dots, x_j into exactly t clusters where every x_i , $1 \leq i \leq j$, is satisfied and the rightmost cluster has size 1;

$\text{viable}(j, t, \text{large}) \Leftrightarrow$ there is a clustering of the set of points x_1, \dots, x_j into exactly t clusters where every x_i , $1 \leq i < j$, is satisfied and the rightmost cluster has size > 1 ;

Notice here that in the latter case, the point x_j is allowed to be not satisfied. The reason is that within the set $\{x_1, \dots, x_j\}$, every x_i , with the exception of x_j , has both

its options for nearest neighbor present, and hence we require them to be satisfied; x_j , on the other hand, still has an unexplored option, namely x_{j+1} .

The predicate *viable* admits the following recursive decomposition:

- (1) $viable(1, 1, small) = true$;
- (2) $viable(1, 1, large) = false$;
- (3) $viable(j, 1, small) = false$ for all $j > 1$;
- (4) $viable(j, 1, large) = true$ for $j > 1$;
- (5) $viable(j, t, small) = true$ for $j > 1$ and $t > 1$ if $viable(j-1, t-1, small) = true$ or $(viable(j-1, t-1, large) = true$ and x_{j-1} is satisfied via x_{j-2});
- (6) $viable(j, t, large) = true$ for all $j > 1$ and $t > 1$.

It is easy to see that we can compute the predicate *viable* for all values of j , and t in $O(nk)$ time. Finally, we output YES if one of $viable(n, k, small)$ and $viable(n, k, large)$ is true.

Locally well-supported clustering Finally, we present an $O(n^3k^2)$ time algorithm for the problem of deciding if a given line metric (X, d) can be clustered into k clusters that are locally well-supported. This algorithm is based on the following idea: for $1 \leq i \leq j \leq n$, whether the set of points x_i, \dots, x_j is a locally well-supported cluster can be determined independently of all other such pairs; this computation depends only on whether various balls around x_i and x_j include sufficiently many points from the cluster $[i, j]$ compared to points to the left of x_i or to the right of x_j . In our algorithm, this predicate *acceptable*(i, j) is first computed for all i, j . Given this information, we define a ternary predicate *viable*(i, j, t), for $1 \leq t \leq k$, defined by

$viable(i, j, t) \Leftrightarrow$ the set of points $\{x_i, \dots, x_j\}$ can be partitioned into t clusters that are all locally well-supported.

The predicate *viable* admits the recurrence given below.

- (1) $viable(i, i, 1) = true$ for all i ;
- (2) $viable(i, j, 1) = true$ for $i < j$ if *acceptable*(i, j);
- (3) $viable(i, j, t) = true$ for $i < j$ and $t > 1$ if $(\exists l : 1 \leq l < t) (\exists k : i \leq k < j) [viable(i, k, l) \wedge viable(k+1, j, t-l)]$.

Now it is easy to see that this recurrence can be implemented in time $O(n^3k^2)$ (since there are n^2k entries to be computed, and each computation takes $O(nk)$ time). \square

5. CONCLUSIONS

We have introduced a formal approach to classify and understand the notion of a good clustering. The use of first-order expressible criteria leads to a variety of interesting and natural clustering problems; we have studied three such notions in this paper. Our work is focused on providing efficient algorithms (either low polynomial time or fixed-parameter tractable) for the problems we studied. One direction for future work is to capture the “semantics” of well-known clustering algorithms, or their natural variants, in terms of first-order expressible criteria. Other possibilities include identifying other interesting criteria and devising efficient algorithms for those, and identifying special-cases of metrics (e.g., shortest-path metric on graphs) and providing efficient algorithms for the resulting first-order expressible clustering problems.

6. REFERENCES

- [1] N. Ailon, M. Charikar, and A. Newman. Aggregating consistent information: Ranking and clustering. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 684–693, 2005.
- [2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2000.
- [3] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In R. Karp, editor, *Complexity of COmputation, SIAM-AMS Proceedings 7*, pages 43–73, 1974.
- [4] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. In *Proceedings of the 21st International Conference on Data Engineering*, pages 341–352, 2005.
- [5] N. Immerman. *Descriptive Complexity*. Springer, 1998.
- [6] N. Jardine and R. Sibson. *Mathematical Taxonomy*. Wiley, 1971.
- [7] J. Kleinberg. An impossibility theorem for clustering. In *Advances in Neural Information Processing Systems 15*, pages 446–453, 2002.