Journal of Machine Learning Research 13 (2012) 203-225 Submitted 9/10; Revised 12/11; Published 1/12

Active Clustering of Biological Sequences*

Konstantin Voevodski KVODSKI@GOOGLE.COM
Google†
76 Ninth Avenue, 10th Floor
New York, NY 10011, USA

Maria-Florina Balcan NINAMF@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Heiko Röglin HEIKO@ROEGLIN.ORG
Department of Computer Science
University of Bonn
Bonn, Germany

Shang-Hua Teng SHANGHUA@USC.EDU
Computer Science Department
University of Southern California
Los Angeles, CA 90089, USA

Yu Xia YUXIA@BU.EDU
Bioinformatics Program and Department of Chemistry
Boston University
Boston, MA 02215, USA

Editor: Rocco Servedio

Abstract

Given a point set S and an unknown metric d on S, we study the problem of efficiently partitioning S into k clusters while querying few distances between the points. In our model we assume that we have access to *one versus all* queries that given a point $s \in S$ return the distances between s and all other points. We show that given a natural assumption about the structure of the instance, we can efficiently find an accurate clustering using only $O(k)$ distance queries. Our algorithm uses an *active* selection strategy to choose a small set of points that we call landmarks, and considers only the distances between landmarks and other points to produce a clustering. We use our procedure to cluster proteins by sequence similarity. This setting nicely fits our model because we can use a fast sequence database search program to query a sequence against an entire data set. We conduct an empirical study that shows that even though we query a small fraction of the distances between the points, we produce clusterings that are close to a desired clustering given by manual classification.

Keywords: clustering, active clustering, k-median, approximation algorithms, approximation stability, clustering accuracy, protein sequences

∗. A preliminary version of this article appeared under the title *Efficient Clustering with Limited Distance Information* in the Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, AUAI Press, Corvallis, Oregon, 632-641.

†. Most of this work was completed at Boston University.

1. Introduction

Clustering from pairwise distance information is an important problem in the analysis and exploration of data. It has many variants and formulations and it has been extensively studied in many different communities, and many different clustering algorithms have been proposed.

Many application domains ranging from computer vision to biology have recently faced an explosion of data, presenting several challenges to traditional clustering techniques. In particular, computing the distances between all pairs of points, as required by traditional clustering algorithms, has become infeasible in many application domains. As a consequence it has become increasingly important to develop effective clustering algorithms that can operate with limited distance information.

In this work we initiate a study of clustering with limited distance information; in particular we consider clustering with a small number of *one versus all* queries. We can imagine at least two different ways to query distances between points. One way is to ask for distances between pairs of points, and the other is to ask for distances between one point and all other points. Clearly, a one versus all query can be implemented as n pairwise queries, where n is the size of the point set, but we draw a distinction between the two because the former is often significantly faster in practice if the query is implemented as a database search.

Our main motivating example for considering one versus all distance queries is sequence similarity search in biology. A program such as BLAST (Altschul et al., 1990) (Basic Local Alignment Search Tool) is optimized to search a single sequence against an entire database of sequences. On the other hand, performing n pairwise sequence alignments takes several orders of magnitude more time, even if the pairwise alignment is very fast. The disparity in runtime is due to the hashing that BLAST uses to identify regions of similarity between the input sequence and sequences in the database. The program maintains a hash table of all *words* in the database (substrings of a certain length), linking each word to its locations. When a query is performed, BLAST considers each word in the input sequence, and runs a local sequence alignment in each of its locations in the database. Therefore the program only performs a limited number of local sequence alignments, rather than aligning the input sequence to each sequence in the database. Of course, the downside is that we never consider alignments between sequences that do not share a word. However, in this case an alignment may not be relevant anyway, and we can assign a distance of infinity to the two sequences. Even though the search performed by BLAST is heuristic, it has been shown that protein sequence similarity identified by BLAST is meaningful (Brenner et al., 1998).

Motivated by such scenarios, in this paper we consider the problem of clustering a data set with an unknown distance function, given only the capability to ask one versus all distance queries. We design an efficient algorithm for clustering accurately with a small number of such queries. To formally analyze the correctness of our algorithm we assume that the distance function is a metric, and that our clustering problem satisfies a natural approximation stability property with respect to the k-median objective function for clustering. In particular, our analysis assumes the (c,ε) approximation stability property of Balcan et al. (2009). For an objective function Φ (such as k-median), the (c,ε)-property assumes that any clustering that is a c-approximation of Φ is structurally close to some "target" clustering C_T (has error of at most ε with respect to C_T). Given this assumption, our goal is to find a clustering that is structurally close to the target (has error of at most ε), which is what we call an *accurate* clustering.

Our first main contribution is designing an algorithm that given the $(1+\alpha,\varepsilon)$-property for the k-median objective finds an accurate clustering with probability at least $1-\delta$ by using only $O(k+\ln\frac{1}{\delta})$ one versus all queries. Our analysis requires that the clusters of the target clustering have size at least $O(\varepsilon n/\alpha)$. In particular, we use the same assumption as Balcan et al. (2009), and we obtain effectively the same performance guarantees as Balcan et al. but by only using a very small number of one versus all queries. In addition to handling this more difficult scenario, we also provide a much faster algorithm. The algorithm of Balcan et al. (2009) can be implemented in $O(n^3)$ time, where n is the size of the point set, while the one proposed here runs in time $O((k+\ln\frac{1}{\delta})n\log n)$.

Our algorithm uses an *active* selection strategy to choose a small set of landmark points. In each iteration our *Landmark-Selection* procedure chooses one of the farthest points from the ones chosen already, where distance from a point s to a set X is given by $\min_{x\in X} d(s,x)$. This procedure is motivated by the observation that if we select points that are far from all the points chosen already, we can quickly cover all the dense regions of the data set. At the same time, our procedure uses some randomness to avoid choosing outliers. After selecting a small set of landmarks, we use a robust single-linkage clustering procedure that we call *Expand-Landmarks*, which constructs a clustering linking only the landmarks that have $s_{\min}$ points in an r-ball around them, for an appropriate choice of $s_{\min}$ and increasing estimates of r. After our initial work a similar robust single-linkage clustering algorithm has been used in Chaudhuri and Dasgupta (2010), which is a generalization of a procedure presented in Wishart (1969). Our algorithm uses only the distances between landmarks and other points to compute a clustering. Therefore the number of one versus all distance queries required is equivalent to the number of landmarks.

The runtime of our algorithm is $O(|L|n\log n)$, where L is the set of landmarks that have been selected. Our adaptive selection procedure significantly reduces the time and query complexity of the algorithm. We show that using our adaptive procedure it suffices to choose only $O(k+\ln\frac{1}{\delta})$ landmarks to compute an accurate clustering with probability at least $1-\delta$. If we use a non-adaptive selection strategy and simply choose landmarks uniformly at random, we must sample a point from each ground truth cluster and therefore need at least $O(k\ln\frac{k}{\delta})$ landmarks to find an accurate clustering. More exactly, the non-adaptive selection strategy requires $O(\frac{n}{\Delta}\ln\frac{k}{\delta})$ landmarks, where Δ is the size of the smallest ground truth cluster. Therefore if we simply choose landmarks uniformly at random, performance can degrade significantly if some clusters are much smaller than the average cluster size. Our theoretic assumption does require that the ground truth clusters are large, but $O(\varepsilon n/\alpha)$ can still be much smaller than the average cluster size, in which case our adaptive selection procedure gives a more significant improvement in runtime and query complexity of the algorithm.

We use our algorithm to cluster proteins by sequence similarity, and compare our results to gold standard manual classifications given in the Pfam (Finn et al., 2010) and SCOP (Murzin et al., 1995) databases. These classification databases are used ubiquitously in biology to observe evolutionary relationships between proteins and to find close relatives of particular proteins. We find that for one of these sources we obtain clusterings that usually closely match the given classification, and for the other the performance of our algorithm is comparable to that of the best known algorithms using the full distance matrix. Both of these classification databases have limited coverage, so a completely automated method such as ours can be useful in clustering proteins that have yet to be classified. Moreover, our method can cluster very large data sets because it is efficient and does not require the full distance matrix as input, which may be infeasible to obtain for a very large data set.

1.1 Related Work

A theoretical assumption that is related to the (c,ε)-property is ε-separability, which is used in Ostrovsky et al. (2006). This property is also referred to as irreducibility in Badoiu et al. (2002) and Kumar et al. (2005). A clustering instance is ε-separated if the cost of the optimal k-clustering is at most ε^2 times the cost of the optimal clustering using $k-1$ clusters. The ε-separability and (c,ε) properties are related: in the case when the clusters are large the Ostrovsky et al. (2006) condition implies the Balcan et al. (2009) condition (see Balcan et al., 2009).

Ostrovsky et al. also present a sampling method for choosing initial centers, which when followed by a single Lloyd-type descent step gives a constant factor approximation of the k-means objective if the instance is ε-separated. However, their sampling method needs information about the full distance matrix because the probability of picking two points as the first two cluster centers is proportional to their squared distance. A very similar (independently proposed) strategy is used by Arthur and Vassilvitskii (2007) to obtain an $O(\log k)$-approximation of the k-means objective on arbitrary instances. Their work was further extended by Ailon et al. (2009) to give a constant factor approximation using $O(k\log k)$ centers. The latter two algorithms can be implemented with k and $O(k\log k)$ one versus all distance queries, respectively.

Awasthi et al. (2010) have since improved the approximation guarantee of Ostrovsky et al. (2006) and some of the results of Balcan et al. (2009). In particular, they show a way to arbitrarily closely approximate the k-median and k-means objective when the Balcan et al. (2009) condition is satisfied and all the target clusters are large. In their analysis they use a property called weak deletion-stability, which is implied by the Ostrovsky et al. (2006) condition and the Balcan et al. (2009) condition when the target clusters are large. However, in order to find a c-approximation (and given our assumption a clustering that is ε-close to the target) the runtime of their algorithm is $n^{O(1/(c-1)^2)}k^{O(1/(c-1))}$. On the other hand, the runtime of our algorithm is completely independent of c, so it remains efficient even when the (c,ε)-property holds only for some very small constant c.

Approximate clustering using sampling has been studied extensively in recent years (see Mishra et al., 2001; Ben-David, 2007; Czumaj and Sohler, 2007). The methods proposed in these papers yield constant factor approximations to the k-median objective using at least $O(k)$ one versus all distance queries. However, as the constant factor of these approximations is at least 2, the proposed sampling methods do not necessarily yield clusterings close to the target clustering C_T if the (c,ε)-property holds only for some small constant $c<2$, which is the interesting case in our setting.

Clustering using coresets is another approach that can be effective in the limited information setting. A coreset is a small representative sample D of the original point set P, which has the property that computational problems on P can be reduced to problems on D. In particular, Feldman and Langberg (2011) give ways to construct coresets such that a $(1+\alpha)$-approximation to the k-median problem on D gives a $(1+\alpha)$-approximation on the full data set. Therefore by using these coresets we can find a $(1+\alpha)$-approximation to the k-median problem using a number of one-versus-all distance queries that is equal to the size of the coreset. However, the size of the coreset of Feldman and Langberg (2011) is $O(k\log(1/\alpha)/\alpha^3)$, so this approach may require significantly more queries to find an accurate clustering in our model if the (c,ε)-property holds only for some very small constant c.

Our landmark selection strategy is related to the *farthest first traversal* used by Dasgupta (2002). In each iteration this traversal selects the point that is farthest from the ones chosen so far, where as in our algorithm the distance from a point s to a set X is given by $\min_{x\in X} d(s,x)$. This traversal

was originally used by Gonzalez (1985) to give a 2-approximation to the k-center problem. The same procedure is also used in the FastMap algorithm in Faloutsos and Lin (1995) as a heuristic to find a pair of distant objects. Farthest first traversal is used in Dasgupta (2002) to produce a hierarchical clustering where for each k the induced k-clustering is a constant factor approximation of the optimal k-center clustering. Our selection strategy is somewhat different from this traversal because in each iteration we uniformly at random choose one of the farthest points from the ones selected so far. In addition, the theoretical guarantees we provide are quite different from those of Gonzales and Dasgupta.

To our knowledge, our work is the first to provide theoretical guarantees for active clustering (clustering by adaptively using only some of the distances between the objects) under a natural condition on the input data. Following the initial publication of this work, Eriksson et al. (2011) have provided another active clustering procedure with theoretical guarantees for hierarchical clustering under a different condition on the input data.

2. Preliminaries

Given a metric space $M = (X,d)$ with point set X, an unknown distance function d satisfying the triangle inequality, and a set of points $S \subseteq X$ with cardinality n, we would like to find a k-clustering C that partitions the points in S into k sets $C_1,\ldots,C_k$ by using *one versus all* distance queries.

In our analysis we assume that S satisfies the (c,ε)-property of Balcan et al. (2009) for the k-median objective function. The k-median objective is to minimize $\Phi(C) = \sum_{i=1}^{k}\sum_{x\in C_i} d(x,c_i)$, where c_i is the median of cluster C_i, which is the point $y \in C_i$ that minimizes $\sum_{x\in C_i} d(x,y)$. Let $\mathrm{OPT}_\Phi = \min_C \Phi(C)$, where the minimum is over all k-clusterings of S, and denote by $C^* = \{C_1^*,\ldots,C_k^*\}$ a clustering achieving this value.

To formalize the (c,ε)-property we need to define a notion of distance between two k-clusterings $C = \{C_1,\ldots,C_k\}$ and $C' = \{C_1',\ldots,C_k'\}$. As in Balcan et al. (2009), we define the distance between C and C' as the fraction of points on which they disagree under the optimal matching of clusters in C to clusters in C':

$$\mathrm{dist}(C,C') = \min_{f\in F_k} \frac{1}{n}\sum_{i=1}^{k} |C_i - C'_{f(i)}|,$$

where F_k is the set of bijections $f:\{1,\ldots,k\} \to \{1,\ldots,k\}$. Two clusterings C and C' are ε-*close* if $\mathrm{dist}(C,C') < \varepsilon$.

We assume that there exists some unknown relevant "target" clustering C_T and given a proposed clustering C we define the error of C with respect to C_T as $\mathrm{dist}(C,C_T)$. Our goal is to find a clustering of low error.

The (c,ε)-property is defined as follows.

Definition 1 *We say that the instance (S,d) satisfies the (c,ε)-property for the k-median objective function with respect to the target clustering C_T if any clustering of S that approximates* OPT_Φ *within a factor of c is ε-close to C_T, that is, $\Phi(C) \le c\cdot\mathrm{OPT}_\Phi \Rightarrow \mathrm{dist}(C,C_T) < \varepsilon$.*

In the analysis of the next section we denote by c_i^* the center point of C_i^*, and use OPT to refer to the value of C^* using the k-median objective, that is, $\mathrm{OPT} = \Phi(C^*)$. We define the *weight* of point x to be the contribution of x to the k-median objective in C^*: $w(x) = \min_i d(x,c_i^*)$. Similarly, we use $w_2(x)$ to denote x's distance to the second-closest cluster center among $\{c_1^*,c_2^*,\ldots,c_k^*\}$. In addition, let w be the average weight of the points: $w = \frac{1}{n}\sum_{x\in S} w(x) = \frac{\mathrm{OPT}}{n}$.

3. Clustering With Limited Distance Information

Algorithm 1 Landmark-Clustering($S, \alpha, \varepsilon, \delta, k$)

$b = (1 + 17/\alpha)\varepsilon n$;
$q = 2b$;
iter $= 4k + 16 \ln \frac{1}{\delta}$;
$s_{\min} = b + 1$;
$n' = n - b$;
L = **Landmark-Selection**(q, iter, S);
C' = **Expand-Landmarks** $(k, s_{\min}, n', L, S)$;
Choose some working landmark l_i from each cluster C'_i;
for each $x \in S$ **do**
  Insert x into the cluster C''_j for $j = \text{argmin}_i d(x, l_i)$;
end for
return C'';

In this section we present a new algorithm that accurately clusters a set of points assuming that the clustering instance satisfies the (c, ε)-property for $c = 1 + \alpha$, and the clusters in the target clustering C_T are not too small. The algorithm presented here is much faster than the one given by Balcan et al., and does not require all pairwise distances as input. Instead, we only require $O(k + \ln \frac{1}{\delta})$ one versus all distance queries to achieve the same performance guarantee as in Balcan et al. (2009) with probability at least $1 - \delta$.

Our clustering method is described in Algorithm 1. We start by using the *Landmark-Selection* procedure to *adaptively* select a small set of landmarks. This procedure repeatedly chooses uniformly at random one of the q farthest points from the ones selected so far (for an appropriate q), where the distance from a point s to a set X is given by $\min_{x \in X} d(s, x)$. We use $d_{\min}(s)$ to refer to the minimum distance between s and any point selected so far. Each time we select a new landmark l, we use a one versus all distance query to get the distances between l and all other points in the data set, and update $d_{\min}(s)$ for each point $s \in S$. To select a new landmark in each iteration, we choose a random number $i \in \{n - q + 1, \ldots, n\}$ and use a linear time selection algorithm to select the ith farthest point. The complete description of this procedure is given in Algorithm 2. We note that our algorithm uses only the distances between landmarks and other points to produce a clustering.

Expand-Landmarks then expands a ball B_l around each landmark $l \in L$. We use the variable r to denote the radius of all the balls: for each landmark $l \in L$, $B_l = \{s \in S \mid d(s, l) \leq r\}$. For each ball there are at most n relevant values of r, each adding at least one more point to it, which results in at most $|L|n$ values of r to try in total. We call a landmark l *working* if B_l contains *at least* $s_{\min}$ points. The algorithm maintains a graph $G_B = (V_B, E_B)$, where $v_l \in V_B$ represents the ball B_l around working landmark l, and two vertices are connected by an (undirected) edge if the corresponding balls overlap on any point: $(v_{l_1}, v_{l_2}) \in E_B$ iff $B_{l_1} \cap B_{l_2} \neq \emptyset$. We emphasize that this graph considers only the balls that have *at least* $s_{\min}$ points in them. In addition, we maintain the set of points in these balls Clustered $= \{s \in S \mid \exists l : s \in B_l \text{ and } v_l \in V_B\}$, and a list of the connected components of G_B, which we refer to as Components$(G_B) = \{\text{Comp}_1, \ldots, \text{Comp}_m\}$.

In each iteration we expand one of the balls by a point, and update G_B, Components(G_B), and Clustered. If G_B has exactly k components, and $|\text{Clustered}| \geq n'$, we terminate and report a clustering

Algorithm 2 Landmark-Selection(q, iter, S)

```
Choose l ∈ S uniformly at random;
L = {l};
for each d(l,s) ∈ QUERY-ONE-VS-ALL(l,S) do
   d_min(s) = d(l,s);
end for
for i = 1 to iter − 1 do
   Let s_1, ..., s_n be an ordering of points in S such that d_min(s_j) ≤ d_min(s_{j+1}) for j ∈ {1,...,n−1};
   Choose l ∈ {s_{n−q+1}, ..., s_n} uniformly at random;
   L = L ∪ {l};
   for each d(l,s) ∈ QUERY-ONE-VS-ALL(l,S) do
      if d(l,s) < d_min(s) then
         d_min(s) = d(l,s);
      end if
   end for
end for
return L;
```

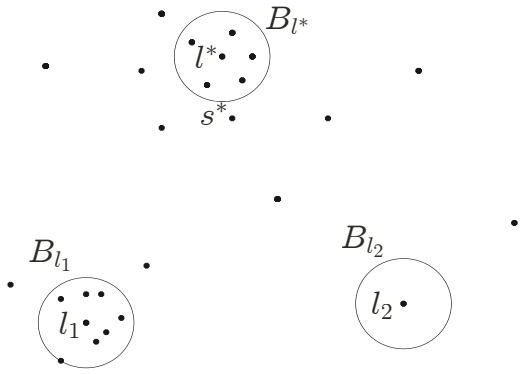


Figure 1: Balls around landmarks are displayed, with the next point to be added to a ball labeled as s^*.

that has a cluster C_i for each component Comp_i, where each C_i contains points in balls in Comp_i. If this condition is never satisfied, we report **no-cluster**. A sketch of this algorithm is given in Algorithm 3. In our description Expand-Ball() is an abstraction for expanding one of the balls by a single point, which is performed by finding the next closest landmark-point pair (l^*, s^*), and adding s^* to B_{l^*} (see Figure 1). In Section 4 we give a full description of an efficient implementation of *Expand-Landmarks*.

The last step of our algorithm takes the clustering C' returned by *Expand-Landmarks* and improves it. We compute a set L' that contains exactly one working landmark from each cluster $C'_i \in C'$ (any working landmark is sufficient), and assign each point $x \in S$ to the cluster corresponding to the closest landmark in L'.

We now present our main theoretical guarantee for Algorithm 1.

Algorithm 3 Expand-Landmarks($k, s_{\min}, n', L, S$)

```
r = 0;
while ((l*, s*) = Expand-Ball()) != null do
    r = d(l*, s*);
    update G_B, Components(G_B), and Clustered;
    if |Components(G_B)| = k and |Clustered| ≥ n' then
        return C = {C_1, ..., C_k} where C_i = {s ∈ S | ∃l: s ∈ B_l and v_l ∈ Comp_i};
    end if
end while
return no-cluster;
```

Theorem 2 *Given a metric space $M = (X,d)$, where d is unknown, and a set of points $S \subseteq X$, if the instance (S,d) satisfies the $(1+\alpha,\varepsilon)$-property for the k-median objective function and if each cluster in the target clustering C_T has size at least $(4+51/\alpha)\varepsilon n$, then Landmark-Clustering outputs a clustering that is ε-close to C_T with probability at least $1-\delta$ in time $O((k+\ln\frac{1}{\delta})|S|\log|S|)$ using $O(k+\ln\frac{1}{\delta})$* one versus all *distance queries.*

Before we prove the theorem, we will introduce some notation and use an analysis similar to the one in Balcan et al. (2009) to argue about the structure of the clustering instance that follows from our assumptions. Let $\varepsilon^* = \text{dist}(C_T, C^*)$. By our assumption that the k-median clustering of S satisfies the $(1+\alpha,\varepsilon)$-property we have $\varepsilon^* < \varepsilon$. Because each cluster in the target clustering has at least $(4+51/\alpha)\varepsilon n$ points, and the *optimal k-median clustering* C^* differs from the target clustering by $\varepsilon^* n \leq \varepsilon n$ points, each cluster in C^* must have at least $(3+51/\alpha)\varepsilon n$ points.

Let us define the *critical distance* $d_{\text{crit}} = \frac{\alpha w}{17\varepsilon}$. We call a point x *good* if both $w(x) < d_{\text{crit}}$ and $w_2(x) - w(x) \geq 17 d_{\text{crit}}$, else x is called *bad*. In other words, the *good* points are those points that are close to their own cluster center and far from any other cluster center. In addition, we will break up the *good* points into *good sets* X_i, where X_i is the set of the *good* points in the optimal cluster C_i^*. So each set X_i is the "core" of the optimal cluster C_i^*.

Note that the distance between two points $x, y \in X_i$ satisfies $d(x,y) \leq d(x,c_i^*) + d(c_i^*,y) = w(x) + w(y) < 2d_{\text{crit}}$. In addition, the distance between any two points in different good sets is greater than $16 d_{\text{crit}}$. To see this, consider a pair of points $x \in X_i$ and $y \in X_{j\neq i}$. The distance from x to y's cluster center c_j^* is at least $17 d_{\text{crit}}$. By the triangle inequality, $d(x,y) \geq d(x,c_j^*) - d(y,c_j^*) > 17 d_{\text{crit}} - d_{\text{crit}} = 16 d_{\text{crit}}$.

If the k-median instance (M,S) satisfies the $(1+\alpha,\varepsilon)$-property with respect to C_T, and each cluster in C_T has size at least $2\varepsilon n$, then

1. less than $(\varepsilon - \varepsilon^*)n$ points $x \in S$ on which C_T and C^* agree have $w_2(x) - w(x) < \frac{\alpha w}{\varepsilon}$.
2. at most $17\varepsilon n/\alpha$ points $x \in S$ have $w(x) \geq \frac{\alpha w}{17\varepsilon}$.

The first part is proved by Balcan et al. (2009). The intuition is that if too many points on which C_T and C^* agree are close enough to the second-closest center among $\{c_1^*, c_2^*, \ldots, c_k^*\}$, then we can move them to the clusters corresponding to those centers, producing a clustering that is structurally far from C_T, but whose objective value is close to OPT, violating the $(1+\alpha,\varepsilon)$-property. The second part follows from the fact that $\sum_{x\in S} w(x) = OPT = wn$.

Then using these facts and the definition of ε^* it follows that at most $\varepsilon^* n + (\varepsilon - \varepsilon^*)n + 17\varepsilon n/\alpha = \varepsilon n + 17\varepsilon n/\alpha = (1 + 17/\alpha)\varepsilon n = b$ points are bad. Hence each $|X_i| = |C_i^* \backslash B| \geq (2 + 34/\alpha)\varepsilon n = 2b$.

In the remainder of this section we prove that given this structure of the clustering instance, *Landmark-Clustering* finds an accurate clustering. We first show that almost surely the set of landmarks returned by *Landmark-Selection* has the property that each of the cluster cores has a landmark near it, which we refer to as the *landmark spread* property. We then argue that given a set of such landmarks, *Expand-Landmarks* finds a partition C' that clusters most of the points in each core correctly. We conclude with the proof of the theorem, which argues that the clustering returned by the last step of our procedure is a further improved clustering that is very close to C^* and C_T.

The *Landmark-Clustering* algorithm first uses *Landmark-Selection*(q, iter, S) to choose a set of landmarks. We say that the *landmark spread* property holds if there is a landmark closer than $2d_{\text{crit}}$ to some point in each good set. The following lemma proves that for $q = 2b$ after selecting only $\text{iter} = O(k + \ln \frac{1}{\delta})$ points the chosen landmarks will have this property with probability at least $1 - \delta$.

Lemma 3 *Given a set of landmarks L = Landmark-Selection $(2b, 4k + 16\ln\frac{1}{\delta}, S)$, the landmark spread property is satisfied with probability at least $1 - \delta$.*

Proof Because there are at most b bad points and in each iteration we uniformly at random choose one of $2b$ points, the probability that a good point is added to L is at least 1/2 in each iteration. Using a Chernoff bound, we show in Lemma 4 that the probability that fewer than k good points have been added to L after $t > 2k$ iterations is less than $e^{-t(1-\frac{2k}{t})^2/4}$. For $t = 4k + 16\ln\frac{1}{\delta}$

$$e^{-t(1-\frac{2k}{t})^2/4} < e^{-(4k+16\ln\frac{1}{\delta})0.5^2/4} < e^{-16\ln\frac{1}{\delta}/16} = \delta.$$

Therefore after $t = 4k + 16\ln\frac{1}{\delta}$ iterations this probability is smaller than δ.

We argue that once we select k good points using our procedure, one of them must be closer than $2d_{\text{crit}}$ to some point in each good set. As in Algorithm 2, we use $d_{\min}(s)$ to denote the minimum distance from point s to any landmark that has been selected so far: $d_{\min}(s) = \min_{l \in L'} d(l, s)$, where L' is the set of landmarks that have been selected so far.

There are two possibilities regarding the first k good points added to L: we select them from distinct good sets, or at least two points are selected from the same good set. If the former is true, the *landmark spread* property trivially holds. If the latter is true, consider the first time that a second point is chosen from the same good set X_i. Let us call these two points x and y (they may be the same point), and assume that y is chosen after x. The distance between x and y must be less than $2d_{\text{crit}}$ because they are in the same good set. Therefore when y is chosen, $d_{\min}(y) \leq d(x, y) < 2d_{\text{crit}}$. Moreover, y is chosen from $\{s_{n-2b+1}, \ldots, s_n\}$, where $d_{\min}(s_j) \leq d_{\min}(s_{j+1})$. Therefore when y is chosen, at least $n - 2b + 1$ points $s \in S$ (including y) satisfy $d_{\min}(s) \leq d_{\min}(y) < 2d_{\text{crit}}$. Because each good set satisfies $|X_i| \geq 2b$, it follows that there must be a landmark closer than $2d_{\text{crit}}$ to some point in each good set.

■

Lemma 4 *The probability that fewer than k good points have been chosen as landmarks after $t > 2k$ iterations of Landmark-Selection is less than $e^{-t(1-\frac{2k}{t})^2/4}$.*

Proof Let X_i be an indicator random variable defined as follows: $X_i = 1$ if the point chosen in iteration i is a good point, and 0 otherwise. Let $X = \sum_{i=1}^{t} X_i$, and μ be the expectation of X. In other words, X is the number of good points chosen after t iterations of the algorithm, and μ is its expected value.

Because in each round we uniformly at random choose one of $2b$ points and there are at most b bad points in total, $\mathrm{E}[X_i] \geq 1/2$ and hence $\mu \geq t/2$. By the Chernoff bound, for any $\delta > 0$, $\Pr[X < (1-\delta)\mu] < e^{-\mu\delta^2/2}$.

If we set $\delta = 1 - \frac{2k}{t}$, we have $(1-\delta)\mu = (1-(1-\frac{2k}{t}))\mu \geq (1-(1-\frac{2k}{t}))t/2 = k$. Assuming that $t \geq 2k$, it follows that $\Pr[X < k] \leq \Pr[X < (1-\delta)\mu] < e^{-\mu\delta^2/2} = e^{-\mu(1-\frac{2k}{t})^2/2} \leq e^{-t/2(1-\frac{2k}{t})^2/2}$. ■

The algorithm then uses the *Expand-Landmarks* procedure to find a k-clustering C'. The following lemma states that C' is an accurate clustering, and has an additional property that is relevant for the last part of the algorithm.

Lemma 5 *Given a set of landmarks L that satisfy the landmark spread property, Expand-Landmarks with parameters $s_{\min} = b+1$, and $n' = n - b$ returns a k-clustering $C' = \{C'_1, C'_2, \ldots C'_k\}$ in which each cluster contains points from a single distinct good set X_i. If we let σ be a bijection mapping each good set X_i to the cluster $C'_{\sigma(i)}$ containing points from X_i, the distance between c_i^* and any working landmark l in $C'_{\sigma(i)}$ satisfies $d(c_i^*, l) < 5d_{\mathrm{crit}}$.*

Proof Lemma 6 argues that because the good sets X_i are well-separated, for $r < 4d_{\mathrm{crit}}$ no ball of radius r can overlap (intersect) more than one X_i, and two balls that overlap different X_i cannot share any points. Lemma 7 argues that because there is a landmark near each good set, there is a value of $r^* < 4d_{\mathrm{crit}}$ such that each X_i is contained in some ball of radius r^*. Moreover, because we only consider balls that have more than b points in them, and the number of bad points is at most b, each ball in G_B must overlap some good set. We can use these facts to argue for the correctness of the algorithm. We refer to the clustering computed in line 6 of *Expand-Landmarks* as the clustering *induced* by G_B, in which each cluster contains points in balls that are in the same component in G_B.

First we observe that for $r = r^*$, G_B has exactly k components and each good set X_i is contained within a distinct component of the induced clustering. Each ball in G_B overlaps with some X_i, and because $r^* < 4d_{\mathrm{crit}}$, we know that each ball in G_B overlaps with exactly one X_i. We also know that balls that overlap different X_i cannot share any points and are thus not connected in G_B. Therefore balls that overlap different X_i will be in different components in G_B. Moreover, each X_i is contained in some ball of radius r^*. For each good set X_i let us designate by B_i a ball that contains all the points in X_i (Figure 2), which is in G_B because the size of each good set satisfies $|X_i| > b$. Any ball in G_B that overlaps X_i will be connected to B_i, and will thus be in the same component as B_i. Therefore for $r = r^*$, G_B has exactly k components, one for each good set X_i, with the corresponding cluster containing all the points in X_i.

There are at least $n - b$ points that are in some X_i, therefore for $r = r^*$ the number of clustered points is at least $n - b$. Hence for $r = r^*$ the condition in line 5 of *Expand-Landmarks* will be satisfied and in line 6 the algorithm will return a k-clustering in which each cluster contains points from a single distinct good set X_i.

Now let us suppose that we start with $r = 0$. Consider the first value of $r = r'$ for which the condition in line 5 is satisfied. At this point G_B has exactly k components and the number of points that are not clustered is at most b. It must be the case that $r' \leq r^* < 4d_{\mathrm{crit}}$ because we know that

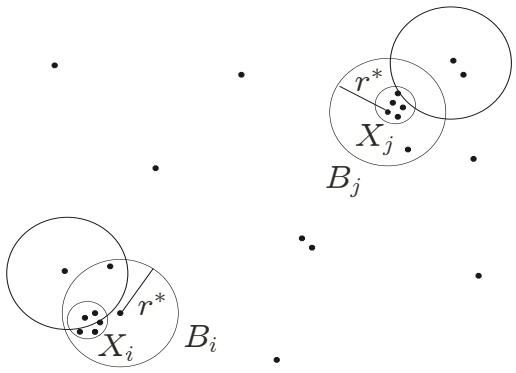


Figure 2: Balls B_i and B_j of radius r^* are shown, which contain good sets X_i and X_j, respectively. The radius of the balls is small in comparison to the distance between the good sets.

the condition is satisfied for $r = r^*$, and we are considering all relevant values of r in ascending order. As before, each ball in G_B must overlap some good set X_i. Again using Lemma 6 we argue that because $r < 4d_{\text{crit}}$, no ball can overlap more than one X_i and two balls that overlap different X_i cannot share any points. It follows that each cluster induced by G_B contains points from a single X_i (so we cannot merge the good sets). Moreover, because the size of each good set satisfies $|X_i| > b$, and there are at most b points that are not clustered, each induced cluster must contain points from a distinct X_i (so we cannot split the good sets). Thus we will return a k-clustering in which each cluster contains points from a single distinct good set X_i.

To prove the second part of the statement, let σ be a bijection matching each good set X_i to the cluster $C'_{\sigma(i)}$ containing points from X_i. Clearly, for any working landmark l in $C'_{\sigma(i)}$ it must be the case that B_l overlaps X_i. Let s^* denote any point on which B_l and X_i overlap. By the triangle inequality, the distance between c_i^* and l satisfies $d(c_i^*, l) \leq d(c_i^*, s^*) + d(s^*, l) < d_{\text{crit}} + r < 5d_{\text{crit}}$. Therefore the distance between c_i^* and any working landmark $l \in C'_{\sigma(i)}$ satisfies $d(c_i^*, l) < 5d_{\text{crit}}$. ■

Lemma 6 *A ball of radius $r < 4d_{\text{crit}}$ cannot contain points from more than one good set X_i, and two balls of radius $r < 4d_{\text{crit}}$ that overlap (intersect) different X_i cannot share any points.*

Proof To prove the first part, consider a ball B_l of radius $r < 4d_{\text{crit}}$ around landmark l. In other words, $B_l = \{s \in S \mid d(s,l) \leq r\}$. If B_l overlaps more than one good set, then it must have at least two points from different good sets $x \in X_i$ and $y \in X_j$. By the triangle inequality it follows that $d(x,y) \leq d(x,l) + d(l,y) \leq 2r < 8d_{\text{crit}}$. However, we know that $d(x,y) > 16d_{\text{crit}}$, giving a contradiction.

To prove the second part, consider two balls B_{l_1} and B_{l_2} of radius $r < 4d_{\text{crit}}$ around landmarks l_1 and l_2. In other words, $B_{l_1} = \{s \in S \mid d(s,l_1) \leq r\}$, and $B_{l_2} = \{s \in S \mid d(s,l_2) \leq r\}$. Assume that they overlap with different good sets X_i and X_j: $B_{l_1} \cap X_i \neq \emptyset$ and $B_{l_2} \cap X_j \neq \emptyset$. For the purpose of contradiction, let's assume that B_{l_1} and B_{l_2} share at least one point: $B_{l_1} \cap B_{l_2} \neq \emptyset$, and use s^* to refer to this point. By the triangle inequality, it follows that the distance between any point $x \in B_{l_1}$ and $y \in B_{l_2}$ satisfies $d(x,y) \leq d(x,s^*) + d(s^*,y) \leq [d(x,l_1) + d(l_1,s^*)] + [d(s^*,l_2) + d(l_2,y)] \leq 4r < 16d_{\text{crit}}$.

Because B_{l_1} overlaps with X_i and B_{l_2} overlaps with X_j, it follows that there is a pair of points $x \in X_i$ and $y \in X_j$ such that $d(x,y) < 16d_{\text{crit}}$, a contradiction. Therefore if B_{l_1} and B_{l_2} overlap different good sets, $B_{l_1} \cap B_{l_2} = \emptyset$. ■

Lemma 7 *Given a set of landmarks L that satisfy the landmark spread property, there is some value of $r^* < 4d_{\text{crit}}$ such that each X_i is contained in some ball B_l around landmark $l \in L$ of radius r^*.*

Proof For each good set X_i choose a point $s_i \in X_i$ and a landmark $l_i \in L$ that satisfy $d(s_i, l_i) < 2d_{\text{crit}}$. The distance between l_i and each point $x \in X_i$ satisfies $d(l_i,x) \leq d(l_i,s_i) + d(s_i,x) < 2d_{\text{crit}} + 2d_{\text{crit}} = 4d_{\text{crit}}$. Consider $r^* = \max_{l_i} \max_{x \in X_i} d(l_i,x)$. Clearly, each X_i is contained in a ball B_{l_i} of radius r^* and $r^* < 4d_{\text{crit}}$. ■

Lemma 8 *Suppose the distance between c_i^* and any working landmark l in $C'_{\sigma(i)}$ satisfies $d(c_i^*, l) < 5d_{\text{crit}}$. Then given a point $x \in C_i^*$ that satisfies $w_2(x) - w(x) \geq 17d_{\text{crit}}$, for any working landmark $l_1 \in C'_{\sigma(i)}$ and any working landmark $l_2 \in C'_{\sigma(j \neq i)}$ it must be the case that $d(x,l_1) < d(x,l_2)$.*

Proof We will show that $d(x,l_1) < w(x) + 5d_{\text{crit}}$, and $d(x,l_2) > w(x) + 12d_{\text{crit}}$. This implies that $d(x,l_1) < d(x,l_2)$.

To prove the former, by the triangle inequality $d(x,l_1) \leq d(x,c_i^*) + d(c_i^*,l_1) = w(x) + d(c_i^*,l_1) < w(x) + 5d_{\text{crit}}$.

To prove the latter, by the triangle inequality $d(x,l_2) \geq d(x,c_j^*) - d(l_2,c_j^*)$. Because $d(x,c_j^*) \geq w_2(x)$ and $d(l_2,c_j^*) < 5d_{\text{crit}}$, we have

$$d(x,l_2) > w_2(x) - 5d_{\text{crit}}. \tag{1}$$

Moreover, because $w_2(x) - w(x) \geq 17d_{\text{crit}}$, we have

$$w_2(x) \geq 17d_{\text{crit}} + w(x). \tag{2}$$

Combining Equations 1 and 2 it follows that $d(x,l_2) > 17d_{\text{crit}} + w(x) - 5d_{\text{crit}} = w(x) + 12d_{\text{crit}}$. ■

Proof [Theorem 2] After using Landmark-Selection to choose $O(k + \ln \frac{1}{\delta})$ points, with probability at least $1 - \delta$ there is a landmark closer than $2d_{\text{crit}}$ to some point in each good set. Given a set of landmarks with this property, each cluster in the clustering $C' = \{C'_1, C'_2, \ldots C'_k\}$ output by *Expand-Landmarks* contains points from a single distinct good set X_i. This clustering can exclude up to b points, all of which may be good. Nonetheless, this means that C' may disagree with C^* on only the bad points and at most b good points. The number of points that C' and C^* disagree on is therefore at most $2b = O(\varepsilon n/\alpha)$. Thus, C' is at least $O(\varepsilon/\alpha)$-close to C^*, and at least $O(\varepsilon/\alpha + \varepsilon)$-close to C_T.

Moreover, C' has an additional property that allows us to find a clustering that is ε-close to C_T. If we use σ to denote a bijection mapping each good set X_i to the cluster $C'_{\sigma(i)}$ containing points from X_i, any working landmark $l \in C'_{\sigma(i)}$ is closer than $5d_{\text{crit}}$ to c_i^*. We can use this observation to find all points that satisfy one of the properties of the good points: points x such that $w_2(x) - w(x) \geq 17d_{\text{crit}}$. Let us call these points the *detectable* points. To clarify, the detectable points are those points that

are much closer to their own cluster center than to any other cluster center in C^*, and the *good* points are a subset of the detectable points that are also very close to their own cluster center.

To find the detectable points using C', we choose some working landmark l_i from each C'_i. For each point $x \in S$, we then insert x into the cluster C''_j for $j = \text{argmin}_i d(x, l_i)$. Lemma 8 argues that each detectable point in C^*_i is closer to every working landmark in $C'_{\sigma(i)}$ than to any working landmark in $C'_{\sigma(j\neq i)}$. It follows that C'' and C^* agree on all the detectable points. Because there are fewer than $(\varepsilon - \varepsilon^*)n$ points on which C_T and C^* agree that are not detectable, it follows that $\text{dist}(C'', C_T) < (\varepsilon - \varepsilon^*) + \text{dist}(C_T, C^*) = (\varepsilon - \varepsilon^*) + \varepsilon^* = \varepsilon$.

Therefore using $O(k + \ln \frac{1}{\delta})$ landmarks we compute an accurate clustering with probability at least $1 - \delta$. The runtime of *Landmark-Selection* is $O(|L|n)$ if we use a linear time selection algorithm to select the next point in each iteration, where $|L|$ is the number of landmarks. Using a min-heap to store all landmark-point pairs and a disjoint-set data structure to keep track of the connected components of G_B, *Expand-Landmarks* can be implemented in $O(|L|n \log n)$ time. A detailed description of this implementation is given in Section 4. The last part of our procedure takes $O(kn)$ time, so the overall runtime of our algorithm is $O(|L|n \log n)$. Therefore to compute an accurate clustering with probability at least $1 - \delta$ the runtime of our algorithm is $O((k + \ln \frac{1}{\delta})n \log n)$. Moreover, we only consider the distances between the landmarks and other points, so we only use $O(k + \ln \frac{1}{\delta})$ one versus all distance queries. ■

4. Implementation of Expand-Landmarks

In order to efficiently expand balls around landmarks, we build a min-heap H of landmark-point pairs (l, s), where the key of each pair is the distance between l and s. In each iteration we find $(l^*, s^*) = H$.deleteMin(), and then add s^* to items(l^*), which stores the points in B_{l^*}. We store points that have been clustered (points in balls of size at least $s_{\min}$) in the set Clustered.

Our implementation assigns each clustered point s to a "representative" landmark, denoted by $lm(s)$. The representative landmark of s is the landmark l of the first large ball B_l that contains s. To efficiently update the components of G_B, we maintain a disjoint-set data structure U that contains sets corresponding to the connected components of G_B, where each ball B_l is represented by landmark l. In other words, U contains a set $\{l_1, l_2, l_3\}$ iff $B_{l_1}, B_{l_2}, B_{l_3}$ form a connected component in G_B.

For each large ball B_l our algorithm will consider all points $s \in B_l$ and perform Update-Components(l, s), which works as follows. If s does not have a representative landmark we assign it to l, otherwise s must already be in $B_{lm(s)}$, and we assign B_l to the same component as $B_{lm(s)}$. If none of the points in B_l are assigned to other landmarks, it will be in its own component. A detailed description of the algorithm is given in Algorithm 4.

During the execution of the algorithm the connected components of G_B must correspond to the sets of U (where each ball B_l is represented by landmark l). Lemma 9 argues that if B_{l_1} and B_{l_2} are *directly* connected in G_B, l_1 and l_2 must be in the same set in U. It follows that whenever B_{l_1} and B_{l_2} are in the same connected component in G_B, l_1 and l_2 will be in the same set in U. Moreover, if B_{l_1} and B_{l_2} are not in the same component in G_B, then l_1 and l_2 cannot be in the same set in U because both start in distinct sets (line 22), and it is not possible for a set containing l_1 to be merged with a set containing l_2.

Algorithm 4 Expand-Landmarks($k, s_{\min}, n', L, S$)

A = ();
for each $s \in S$ **do**
  $lm(s)$ = null;
  for each $l \in L$ **do**
    A.add($(l,s), d(l,s)$);
  end for
end for
H = build-heap(A);
for each $l \in L$ **do**
  items(l) = ();
end for
Set Clustered = ();
U = ();
while H.hasNext() **do**
  (l^*, s^*) = H.deleteMin();
  items(l^*).add(s^*);
  if items(l^*).size() $> s_{\min}$ **then**
    Update-Components(l^*, s^*);
    Clustered.add($s*$);
  end if
  if items(l^*).size() == $s_{\min}$ **then**
    U.MakeSet(l^*);
    for each $s \in$ items(l^*) **do**
      Update-Components(l^*, s);
      Clustered.add(s);
    end for
  end if
  if Clustered.size() $\geq n'$ and U.size() == k **then**
    return Format-Clustering();
  end if
end while
return no-cluster;

Algorithm 5 Update-Components(l, s)

if $lm(s)$ == null **then**
  $lm(s) = l$;
else
  $c_1 = U$.find(l);
  $c_2 = U$.find($lm(s)$);
  U.union(c_1, c_2);
end if

Algorithm 6 Format-Clustering()

```
1:  C = ();
2:  for each Set L' in U do
3:     Set Cluster = ();
4:     for each l ∈ L' do
5:        for each s ∈ items(l) do
6:           Cluster.add(s);
7:        end for
8:     end for
9:     C.add(Cluster);
10: end for
11: return C;
```

Lemma 9 *If balls B_{l_1} and B_{l_2} are directly connected in G_B, then landmarks l_1 and l_2 must be in the same set in U.*

Proof If B_{l_1} and B_{l_2} are directly connected in G_B, then B_{l_1} and B_{l_2} must overlap on some point s. Without loss of generality, suppose s is added to B_{l_1} before it is added to B_{l_2}. When s is added to B_{l_1}, $lm(s) = l_1$ if s does not yet have a representative landmark (lines 1-2 of Update-Components), or $lm(s) = l'$ and both l_1 and l' are put in the same set (lines 4-6 of Update-Components). When s is added to B_{l_2}, if $lm(s) = l_1$, then l_1 and l_2 will be put in the same set in U. If $lm(s) = l'$, l' and l_2 will be put in the same set in U, which also contains l_1. ■

It takes $O(|L|n)$ time to build H (linear in the size of the heap). Each deleteMin() operation takes $O(\log(|L|n))$ (logarithmic in the size of the heap), which is equivalent to $O(\log(n))$ because $|L| \leq n$. If U is implemented by a union-find algorithm, Update-Components takes amortized time of $O(\alpha(|L|)$, where α denotes the inverse Ackermann function. Moreover, Update-Components may be called at most once for each iteration of the while loop in Expand-Landmarks (for a pair (l^*, s^*) it is either called on line 18 if B_{l^*} is large enough, or it is called on line 24 when B_{l^*} grows large enough). All other operations also take time proportional to the number of landmark-point pairs. So the runtime of this algorithm is $O(|L|n)$ + iter $\cdot\, O(\log n + \alpha(|L|))$, where iter is the number of iterations of the while loop. As the number of iterations is bounded by $|L|n$, and $\alpha(|L|)$ is effectively constant, this gives a worst-case running time of $O(|L|n\log n)$.

5. Empirical Study

We use our *Landmark Clustering* algorithm to cluster proteins using sequence similarity. As mentioned in the Introduction, one versus all distance queries are particularly relevant in this setting because of sequence database search programs such as BLAST (Altschul et al., 1990) (Basic Local Alignment Search Tool). BLAST aligns a queried sequence to sequences in the database, and produces a "bit score" for each alignment, which is a measure of its quality (we invert the bit score to make it a distance). However, BLAST does not consider alignments with some of the sequences in the database, in which case we assign distances of infinity to the corresponding sequences. We observe that if we define distances in this manner they almost form a metric in practice: when we

draw triplets of sequences at random and check the distances between them the triangle inequality is almost always satisfied. Moreover, BLAST is very successful at detecting sequence homology in large sequence databases, therefore it is plausible that k-median clustering using these distances is approximately stable with respect to a relevant target clustering C_T, which groups together sequences with shared evolutionary ancestry.

We perform experiments on data sets obtained from two classification databases: Pfam (Finn et al., 2010), version 24.0, October 2009; and SCOP (Murzin et al., 1995), version 1.75, June 2009. Both of these sources classify proteins by their evolutionary relatedness, therefore we can use their classifications as a ground truth to evaluate the clusterings produced by our algorithm and other methods.

Pfam classifies proteins using hidden Markov models (HMMs) that represent multiple sequence alignments. There are two levels in the Pfam classification hierarchy: family and clan. In our clustering experiments we compare with a classification at the family level because the relationships at the clan level are less likely to be discerned with sequence alignment. In each experiment we randomly select several large families (of size between 1000 and 10000) from Pfam-A (the manually curated part of the classification), retrieve the sequences of the proteins in these families, and use our *Landmark-Clustering* algorithm to cluster the data set.

SCOP groups proteins on the basis of their 3D structures, so it only classifies proteins whose structure is known. Thus the data sets from SCOP are much smaller in size. The SCOP classification is also hierarchical: proteins are grouped by class, fold, superfamily, and family. We consider the classification at the superfamily level because this seems most appropriate given that we are only using sequence information. As with the Pfam data, in each experiment we create a data set by randomly choosing several superfamilies (of size between 20 and 200), retrieve the sequences of the corresponding proteins, and use our *Landmark-Clustering* algorithm to cluster the data set.

Once we cluster a particular data set, we compare the clustering to the manual classification using the distance measure from the theoretical part of our work. To find the fraction of misclassified points under the optimal matching of clusters in C to clusters in C' we solve a minimum weight bipartite matching problem where the cost of matching C_i to $C'_{f(i)}$ is $|C_i - C'_{f(i)}|/n$. In addition, we compare clusterings to manual classifications using the F-measure, which is used in another study that clusters protein sequences (Paccanaro et al., 2006). The F-measure is a similarity score between 0 and 1, where 1 indicates an exact match between the two clusterings (see Appendix A). This measure has also been used in other studies (see Cheng et al., 2006), and is related to our notion of clustering distance (see Lemma 10 in Appendix A). Surprisingly, the F-measure is not symmetric; in our experiments we compute the similarity of a clustering C to the manual classification C_M as $F(C_M, C)$.

5.1 Choice of Parameters

To run *Landmark-Clustering*, we set k using the number of clusters in the ground truth clustering. For each Pfam data set we use $5k$ landmarks/queries, and for each SCOP data set we use $10k$ landmarks/queries. In addition, our algorithm uses three parameters $(q, s_{\min}, n')$ whose value is set in the proof based on α and ε, assuming that the clustering instance satisfies the $(1+\alpha, \varepsilon)$-property. In practice we must choose some value for each parameter. In our experiments we set q as a function of the average size of the ground truth clusters (ave-size), $s_{\min}$ as a function of the size of the smallest ground truth cluster (min-size), and n' as a function of the number of points in the data set. For the

Pfam data sets we set $q =$ ave-size, $s_{\min} = 0.25 \cdot$ min-size, and $n' = 0.7n$. Because the selection of landmarks is randomized, for each data set we compute several clusterings, compare each to the ground truth, and report the median quality.

Landmark-Clustering is most sensitive to the $s_{\min}$ parameter, and will not report a clustering if $s_{\min}$ is too small or too large. We recommend trying several values of this parameter, in increasing or decreasing order, until one gets a clustering and none of the clusters are too large. If the user gets a clustering where one of the clusters is very large, this likely means that several ground truth clusters have been merged. This may happen because $s_{\min}$ is too small causing balls of outliers to connect different cluster cores, or $s_{\min}$ is too large causing balls intersecting different cluster cores to overlap.

In our SCOP experiments we have to use the above-mentioned heuristic to set the $s_{\min}$ parameter. We start with $s_{\min} =$ min-size, and decrement it until we get exactly k clusters and none of the clusters are too large (larger than twice the size of the largest ground truth cluster). For the SCOP data sets we set $q =$ ave-size, and $n' = 0.5n$. As before, for each data set we compute several clusterings, compare each to the ground truth, and report the median quality.

Our algorithm is less sensitive to the n' parameter. However, if the user sets n' too large some ground truth clusters may be merged, so we recommend using a smaller value ($0.5n \leq n' \leq 0.7n$) because all of the points are still clustered during the last step. Again, for some values of n' the algorithm may not output a clustering, or output a clustering where some of the clusters are too large.

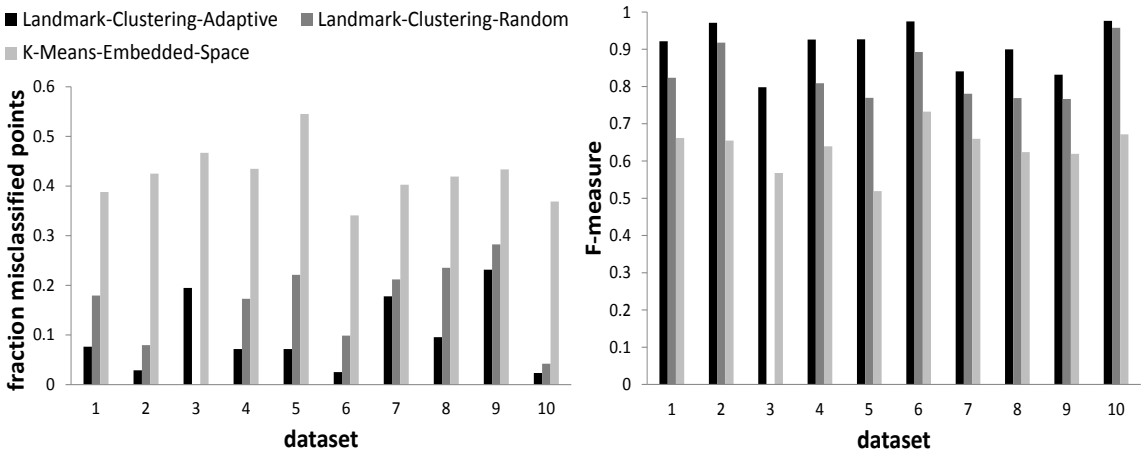
It is important to not choose an extreme value for the q parameter. The value of q must be large enough to avoid repeatedly choosing outliers (if $q = 1$ we are likely to choose an outlier in each iteration), but small enough to quickly find a landmark near each cluster core. If we set $q = n$, the algorithm selects landmarks uniformly at random, and we may need significantly more landmarks to choose one from each cluster core by chance.

In our experiments we compare the algorithm that uses the adaptive selection strategy with the alternative that chooses landmarks uniformly at random. The alternative algorithm uses exactly the same number of landmarks, and other parameters stay the same as well. When the data has the structure that follows from our assumptions, the non-adaptive selection strategy may require significantly more landmarks to cover all cluster cores (especially if the sizes of the ground truth clusters are not well-balanced). Therefore when the data has the right structure and we cannot afford to use many landmarks, we expect to find more accurate clusterings with the adaptive selection strategy.

5.2 Results

Figure 3 shows the results of our experiments on the Pfam data sets. As discussed earlier, to test our adaptive landmark selection strategy we compare our algorithm, which is labeled *Landmark-Clustering-Adaptive*, with the same algorithm that chooses landmarks uniformly at random, which we refer to as *Landmark-Clustering-Random*. We can see that for a lot of the data sets *Landmark-Clustering-Adaptive* finds a clustering that is quite close to the ground truth. The alternative algorithm does not perform as well, and for data set 3 fails to find a clustering altogether.

The Pfam data sets are very large, so as a benchmark for comparison we can only consider algorithms that use a comparable amount of distance information (because we do not have the full distance matrix). A natural choice is the following algorithm: uniformly at random choose a set of



(a) Comparison using fraction of misclassified points

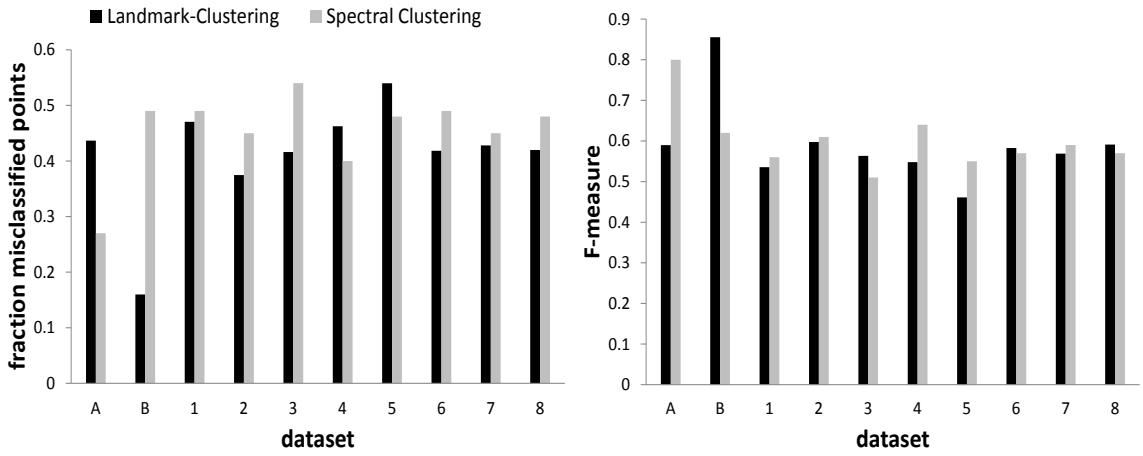
(b) Comparison using the F-measure

Figure 3: Comparing the performance of *Landmark-Clustering-Adaptive*, *Landmark-Clustering-Random*, and k-means in the embedded space on 10 data sets from Pfam. Data sets *1-10* are created by randomly choosing 8 families from Pfam of size s, $1000 \leq s \leq 10000$. *(a)* Comparison using the distance measure from the theoretical part of our work. *(b)* Comparison using the F-measure.

landmarks L, $|L| = d$; embed each point in a d-dimensional space using distances to L; use k-means clustering in this space (with distances given by the Euclidean norm). This embedding scheme is a Lipschitz embedding with singleton subsets (see Tang and Crovella, 2003), which gives distances with low distortion for points near each other in a metric space.

Notice that this procedure uses exactly d one versus all distance queries, so we can set d equal to the number of queries used by our algorithm. We expect this algorithm to work well, and if we look at Figure 3 we can see that it finds reasonable clusterings. Still, the clusterings reported by this algorithm do not closely match the Pfam classification, showing that our results are indeed significant.

Figure 4 shows the results of our experiments on the SCOP data sets. For these data sets we find less accurate clusterings, which is likely because the SCOP classification is based on biochemical and structural evidence in addition to sequence evidence. By contrast, the Pfam classification is based entirely on sequence information. Still, because the SCOP data sets are much smaller, we can compare our algorithm with methods that require distances between all the points. In particular, Paccanaro et al. (2006) show that spectral clustering using sequence similarity data works well when applied to the proteins in SCOP. Thus we use the exact method described by Paccanaro et al. (2006) as a benchmark for comparison on the SCOP data sets. Moreover, other than clustering randomly generated data sets from SCOP, we also consider the two main examples from Paccanaro et al., which are labeled *A* and *B* in the figure. From Figure 4 we can see that the performance of *Landmark-Clustering* is comparable to that of the spectral method, which is very good considering that the spectral clustering algorithm significantly outperforms other clustering algorithms on this



(a) Comparison using fraction of misclassified points

(b) Comparison using the F-measure

Figure 4: Comparing the performance of *Landmark-Clustering* and spectral clustering on 10 data sets from SCOP. Data sets *A* and *B* are the two main examples from Paccanaro et al. (2006), the other data sets (*1-8*) are created by randomly choosing 8 superfamilies from SCOP of size *s*, $20 \leq s \leq 200$. *(a)* Comparison using the distance measure from the theoretical part of our work. *(b)* Comparison using the F-measure.

data (Paccanaro et al., 2006). Moreover, the spectral clustering algorithm requires the full distance matrix as input, and takes much longer to run.

For the SCOP data sets we do not see any significant difference in performance when we compare the adaptive and non-adaptive landmark selection strategies. This is likely because we are using a lot of landmarks (10 times the number of clusters), and selecting landmarks uniformly at random is sufficient to cover the dense groups of points. Unfortunately for these data the algorithm has little success if we use fewer than $10k$ landmarks (it usually cannot find a clustering altogether), so we cannot test how the two selection strategies perform when we use fewer landmarks.

5.3 Testing the (c, ε)-property

To see whether approximation stability of the *k*-median objective function is a reasonable assumption for our data, we look at whether our data sets resemble the structure that is implied by our assumption. We do this by measuring the separation of the ground truth clusters in our data sets. For each data set in our study, we sample some points from each ground truth cluster. We then look at whether the sampled points are more similar to points in the same cluster than to points in other clusters. More specifically, for each point we record the median within-cluster similarity, and the maximum between-cluster similarity. If our data sets indeed have well-separated cluster cores, as implied by our assumption, then for a lot of the points the median within-cluster similarity should be significantly larger than the maximum between-cluster similarity. We can see that this is indeed the case for the Pfam data sets. However, this is not typically the case for the SCOP data sets, where most points have little similarity to the majority of the points in their ground truth cluster. These observations explain our results on the two sets of data: we are able to accurately cluster the Pfam

data sets, and our algorithm is much less accurate on the SCOP data sets. The complete results of these experiments can be found at `http://xialab.bu.edu/resources/ac`.

Testing whether the (c,ε)-property holds for the *k*-median objective is an NP-complete problem (Schalekamp et al., 2010). Moreover, in our experiments when we set the parameters of the algorithm we don't preserve the relationships between them as in Algorithm 1. In particular, in our experiments when we set n' to $n - s_{\min} + 1$ as in Algorithm 1, the algorithm usually fails to report a clustering no matter what value of $s_{\min}$ we try. This means that these data sets in fact do not satisfy our exact theoretic assumptions. Still, when we only slightly break the dependence between the parameters, we are able to find accurate clusterings for the Pfam data sets. For the SCOP data sets we have to further break the dependence between the parameters, and use an additional heuristic to estimate $s_{\min}$, which is not surprising because these data do not have the structure that the algorithm exploits.

6. Conclusion and Open Questions

In this work we presented a new algorithm for clustering large data sets with limited distance information. As opposed to previous settings, our goal was not to approximate some objective function like the *k*-median objective, but to find clusterings close to the ground truth. We proved that our algorithm yields accurate clusterings with only a small number of one versus all distance queries, given a natural assumption about the structure of the clustering instance. This assumption has been previously analyzed by Balcan et al. (2009), but in the full distance information setting. By contrast, our algorithm uses only a small number of queries, it is much faster, and it has effectively the same formal performance guarantees as the one introduced by Balcan et al. (2009).

To demonstrate the practical use of our algorithm, we clustered protein sequences using a sequence database search program as the one versus all query. We compared our results to gold standard manual classifications of protein evolutionary relatedness given in Pfam (Finn et al., 2010) and SCOP (Murzin et al., 1995). We find that our clusterings are quite accurate when we compare with the classification given in Pfam. For SCOP our clusterings are as accurate as state of the art methods, which take longer to run and require the full distance matrix as input.

Our main theoretical guarantee assumes large target clusters. It would be interesting to design a provably correct algorithm for the case of small clusters as well. It would also be interesting to study other objective functions for clustering under similar approximation stability assumptions. In particular, Voevodski et al. (2011) study the implications of the (c,ε)-property for the *min-sum* objective function. However, the algorithm presented there is not as efficient and is less accurate in clustering protein sequences.

Acknowledgments

Konstantin Voevodski was supported by an IGERT Fellowship through NSF grant DGE-0221680 awarded to the ACES Training Program at BU Center for Computational Science. Maria Florina Balcan was supported in part by NSF grant CCF-0953192, by ONR grant N00014-09-1-0751, AFOSR grant FA9550-09-1-0538, and a Google Research Award. Heiko Röglin was supported by a Veni grant from the Netherlands Organisation for Scientific Research. Shang-Hua Teng was supported in part by NSF grant CCR-0635102.

Appendix A.

In this section we give the definition of F-measure, which is another way to compare two clusterings. We also show a relationship between our measure of distance and the F-measure.

A.1 F-measure

The F-measure compares two clusterings C and C' by matching each cluster in C to a cluster in C' using a harmonic mean of Precision and Recall, and then computing a "per-point" average. If we match C_i to C'_j, Precision is defined as $P(C_i,C'_j) = \frac{|C_i \cap C'_j|}{|C_i|}$. Recall is defined as $R(C_i,C'_j) = \frac{|C_i \cap C'_j|}{|C_j|}$. For C_i and C'_j the harmonic mean of Precision and Recall is then equivalent to $\frac{2 \cdot |C_i \cap C'_j|}{|C_i|+|C'_j|}$, which we denote by $\mathrm{pr}(C_i,C'_j)$ to simplify notation. The F-measure is then defined as

$$F(C,C') = \frac{1}{n} \sum_{C_i \in C} |C_i| \max_{C'_j \in C'} \mathrm{pr}(C_i,C'_j).$$

Note that this quantity is between 0 and 1, where 1 corresponds to an exact match between the two clusterings.

Lemma 10 *Given two clusterings C and C', if* $\mathrm{dist}(C,C') = d$ *then* $F(C,C') \geq 1 - 3d/2$.

Proof Denote by σ the optimal matching of clusters in C to clusters in C', which achieves a misclassification of dn points. We show that just considering $\mathrm{pr}(C_i,C'_{\sigma(i)})$ for each $C_i \in C$ achieves an F-measure of at least $1-3d/2$:

$$F(C,C') \geq \frac{1}{n} \sum_{C_i \in C} |C_i| \mathrm{pr}(C_i,C'_{\sigma(i)}) \geq 1 - 3d/2.$$

To see this, for a match of C_i to $C'_{\sigma(i)}$ we denote by m_i^1 the number of points that are in C_i but not in $C'_{\sigma(i)}$, and by m_i^2 the number of points that are in $C'_{\sigma(i)}$ but not in C_i: $m_i^1 = |C_i - C'_{\sigma(i)}|$, $m_i^2 = |C'_{\sigma(i)} - C_i|$. Because the total number of misclassified points is dn it follows that

$$\sum_{C_i \in C} m_i^1 = \sum_{C_i \in C} m_i^2 = dn.$$

By definition, $|C_i \cap C'_{\sigma(i)}| = |C_i| - m_i^1$. Moreover, $|C'_{\sigma(i)}| = |C'_{\sigma(i)} \cap C_i| + m_i^2 \leq |C_i| + m_i^2$. It follows that

$$\mathrm{pr}(C_i,C'_{\sigma(i)}) = \frac{2(|C_i| - m_i^1)}{|C_i| + |C'_{\sigma(i)}|} \geq \frac{2(|C_i| - m_i^1)}{2|C_i| + m_i^2} = \frac{2|C_i| + m_i^2}{2|C_i| + m_i^2} - \frac{m_i^2 + 2m_i^1}{2|C_i| + m_i^2} \geq 1 - \frac{m_i^2 + 2m_i^1}{2|C_i|}.$$

We can now see that

$$\frac{1}{n} \sum_{C_i \in C} |C_i| \mathrm{pr}(C_i,C'_{\sigma(i)}) \geq \frac{1}{n} \sum_{C_i \in C} |C_i| (1 - \frac{m_i^2 + 2m_i^1}{2|C_i|}) = \frac{1}{n} \sum_{C_i \in C} |C_i| - \frac{1}{2n} \sum_{C_i \in C} m_i^2 + 2m_i^1 = 1 - \frac{3dn}{2n}.$$

■

References

N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. In *Advances in Neural Information Processing Systems*, 2009.

S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, 2007.

P. Awasthi, A. Blum, and O. Sheffet. Stability yields a PTAS for k-median and k-means clustering. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, 2010.

M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via coresets. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, 2002.

M. F. Balcan, A. Blum, and A. Gupta. Approximate clustering without the approximation. In *Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms*, 2009.

S. Ben-David. A framework for statistical clustering with constant time approximation algorithms for *k*-median and *k*-means clustering. *Machine Learning*, 66(2-3):243–257, 2007.

S. E. Brenner, C. Chothia, and T. J. Hubbard. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proceedings of the National Academy of Sciences USA*, 95(11):6073–6078, 1998.

K. Chaudhuri and S. Dasgupta. Rates of convergence for the cluster tree. In *Advances in Neural Information Processing Systems*, 2010.

D. Cheng, R. Kannan, S. Vempala, and G. Wang. A divide-and-merge methodology for clustering. *ACM Transactions on Database Systems*, 31(4):1499–1525, 2006.

A. Czumaj and C. Sohler. Sublinear-time approximation algorithms for clustering via random sampling. *Random Structures and Algorithms*, 30(1-2):226–256, 2007.

S. Dasgupta. Performance guarantees for hierarchical clustering. In Jyrki Kivinen and Robert Sloan, editors, *Computational Learning Theory*, volume 2375 of *Lecture Notes in Computer Science*, pages 235–254. Springer Berlin / Heidelberg, 2002.

B. Eriksson, G. Dasarathy, A. Singh, and R. Nowak. Active clustering: Robust and efficient hierarchical clustering using adaptively selected similarities. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 2011.

C. Faloutsos and K. Lin. Fastmap: A fast algorithm for indexing, datamining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, 1995.

D. Feldman and M. Langberg. A unified framework for approximating and clustering data. In *Proceedings of the 43rd ACM Symposium on Theory of Computing*, 2011.

R. D. Finn, J. Mistry, J. Tate, P. Coggill, A. Heger, J. E. Pollington, O. L. Gavin, P. Gunesekaran, G. Ceric, K. Forslund, L. Holm, E. L. Sonnhammer, S. R. Eddy, and A. Bateman. The Pfam protein families database. *Nucleic Acids Research*, 38:D211–222, 2010.

T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

A. Kumar, Y. Sabharwal, and S. Sen. Linear time algorithms for clustering problems in any dimensions. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming*, 2005.

N. Mishra, D. Oblinger, and L Pitt. Sublinear time approximate clustering. In *Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms*, 2001.

A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: A structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247:536–540, 1995.

R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science*, 2006.

A. Paccanaro, J. A. Casbon, and M. A. S. Saqi. Spectral clustering of protein sequences. *Nucleic Acids Research*, 34(5):1571–1580, 2006.

F. Schalekamp, M. Yu, and A. van Zuylen. Clustering with or without the approximation. In *Proceedings of the 16th Annual International Computing and Combinatorics Conference*, 2010.

L. Tang and M. Crovella. Virtual landmarks for the internet. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, 2003.

K. Voevodski, M. F. Balcan, H. Röglin, S. Teng, and Y. Xia. Min-sum clustering of protein sequences with limited distance information. In *Proceedings of the 1st International Workshop on Similarity-Based Pattern Analysis and Recognition*, 2011.

D. Wishart. Mode analysis: A generalization of nearest neighbor which reduces chaining effects. In *Proceedings of the Colloquium on Numerical Taxonomy held in the University of St. Andrews*, 1969.