Performance guarantees for hierarchical clustering

Sanjoy Dasgupta

Department of Computer Science and Engineering
University of California, San Diego

Philip M. Long

Genome Institute of Singapore

Abstract

We show that for any data set in any metric space, it is possible to construct a hierarchical clustering with the guarantee that for *every* k, the induced k-clustering has cost at most eight times that of the optimal k-clustering. Here the cost of a clustering is taken to be the maximum radius of its clusters. Our algorithm is similar in simplicity and efficiency to popular agglomerative heuristics for hierarchical clustering, and we show that these heuristics have unbounded approximation factors.

Key words: Hierarchical clustering, complete linkage, k-center

1 Introduction

A *hierarchical clustering* of n data points is a recursive partitioning of the data into $2, 3, 4, \ldots$ and finally n, clusters. Each intermediate clustering is made more fine-grained by dividing one of its clusters. Figure 1 shows one possible hierarchical clustering of a five-point data set.

Such hierarchical representations of data have long been a staple of biologists and social scientists, and since the sixties or seventies they have been a standard part of the statistician's toolbox. Their popularity is easy to understand.

Email addresses: dasgupta@cs.ucsd.edu (Sanjoy Dasgupta), gislongp@nus.edu.sg (Philip M. Long).

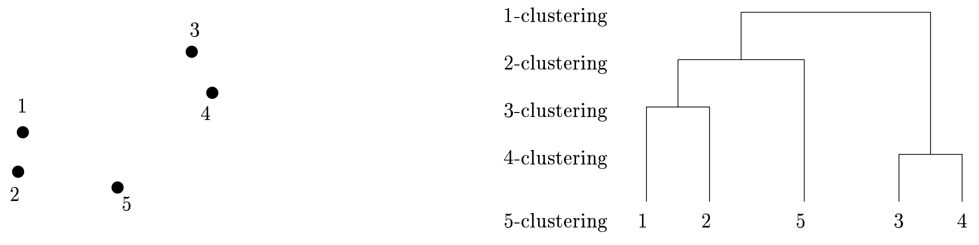

Fig. 1. A hierarchical clustering of five points.

They require no prior specification of the number of clusters, they permit the data to be understood simultaneously at many levels of granularity, and there are some simple, greedy heuristics that can be used to construct them.

It is very useful to be able to view data at different levels of detail, but the requirement that these clusterings be nested within each other presents some fundamental difficulties. Consider the data set of Figure 2, consisting of six evenly-spaced collinear points in the Euclidean plane. The most commonly-used clustering cost functions, such as that of k-means, strive to produce clusters of small radius or diameter. Under such criteria, the best 2-clustering (grouping into two clusters) of this data is unambiguous, as is the best 3-clustering. However, they are hierarchically incompatible. This raises a troubling question: by requiring a hierarchical structure, do we doom ourselves to intermediate clusterings of poor quality?

To rephrase this more constructively, must there always exist a hierarchical clustering in which, for *every* k, the induced k-clustering (grouping into k clusters) is close to the optimal k-clustering under some reasonable cost function? As we have already seen, it is quite possible that the optimal cost-based k-clustering cannot be obtained by merging clusters of the optimal $(k+1)$-clustering. Can they be so far removed that they cannot be reconciled even approximately into a hierarchical structure? Despite the large body of theoretical work on hierarchical clustering (see, for instance, [8] and the references therein), this fundamental existence question has remained unanswered. We resolve it via the following reassuring result.

Theorem 1 *Take the cost of a clustering to be the largest radius of its clusters. Then, any data set in any metric space has a hierarchical clustering in which, for each k, the induced k-clustering has cost at most eight times that of the optimal k-clustering.*

Remark A simple modification of our analysis shows that this result also holds if the cost of a clustering is taken to be the largest diameter of its clusters.

We present an algorithm for constructing such a hierarchy which is similar in simplicity and efficiency to standard heuristics for hierarchical clustering. It is

Fig. 2. What is the best hierarchical clustering for this data set?

based upon the *farthest-first traversal* of a set of points, used by González [7] as an approximation algorithm for the closely-related *k-center* problem. His use of this traversal for clustering is ingenious, and in fact just a cursory examination of its properties is necessary for his results. For hierarchical clustering, we examine it in greater detail and need to built upon it. Specifically, the farthest-first traversal of n data points yields a sequence of "centers" $\mu_1, \ldots, \mu_n$ such that for any k, the first k of these centers define a k-clustering which is within a factor two of optimal. However, the n clusterings created in this way are not hierarchical. Our main contribution is to demonstrate a simple and elegant way of using the information found by the traversal to create a hierarchical clustering.

Our algorithm also has a randomized variant with a tighter constant of approximation.

Theorem 2 *In the setting of the previous theorem, there is a randomized algorithm which produces a hierarchical clustering such that, for each k, the induced k-clustering has expected cost at most* $2e \approx 5.44$ *times that of the optimal k-clustering.*

Unlike our algorithm, the most common heuristics for hierarchical clustering work *bottom-up*, starting with a separate cluster for each point, and then progressively merging the two "closest" clusters until only a single cluster remains. The different schemes are distinguished by their notion of closeness. In *single-linkage* clustering, the distance between two clusters is the distance between their closest pair of points. In *complete-linkage* clustering, it is the distance between their farthest pair of points (and thus complete-linkage is explicitly trying to minimize the diameter, one of our cost functions). *Average-linkage* has many variants; in the one we consider, the distance between clusters is the distance between their means [5].

We analyze the worst-case behavior of these three heuristics, and find that their approximation ratios are unbounded.

Theorem 3 *For any k, single-linkage can produce induced k-clusterings which are a multiplicative factor k from optimal, while average- and complete-linkage can be off by a multiplicative factor of* $\log_2 k$.

The problems of single-linkage clustering are already well understood by statisticians; we give a lower bound on its performance mostly as further intuition about what different approximation factors mean. On the other hand, our bad cases for the other two heuristics yield insights into their particular failings. To be fair, the only algorithm which can really be judged in comparison to ours is complete-linkage, because it attempts to optimize the same cost function.

Since the publication of a preliminary abstract of this paper, we have learned that earlier work of Charikar, Chekuri, Feder and Motwani [3] uses similar techniques for a loosely related problem, and achieves the same bounds. These authors consider an online setting, in which an endless stream of data is arriving, and the goal is to maintain a k-clustering which is at all times within a constant factor of optimal. At first glance this problem is very different from ours, because it is focused on a particular value of k rather than optimizing all k simultaneously. However, the authors limit attention to a certain class of agglomerative algorithms, and this leads them to use many of the same techniques as ours. Like us, they rely upon a k-center algorithm, albeit a different one due to Hochbaum and Shmoys [9], and they also use a geometric binning of distances.

Ours is the first provably good approximation algorithm for hierarchical clustering under a radius-based cost function. We therefore start by reviewing the literature on approximation algorithms for k-clustering, to convey some sense of what these approximation factors mean, and what factors one might hope to achieve.

2 Approximation algorithms for clustering

The most widely-used clustering algorithms – k-means, EM, and the various hierarchical agglomerative procedures – have received almost no attention from theoretical computer scientists. Some exceptions include work by Kearns, Mansour, and Ng [10], and by Dasgupta and Schulman [4]. On the other hand, there has been a lot of theoretical work on the *k-center* and *k-median* problems. In each of these, the input consists of points in Euclidean space (or more generally, in a metric space) as well as a preordained number of clusters k, and the goal is to find a partition of the points into clusters $C_1, \ldots, C_k$, and also cluster centers $\mu_1, \ldots, \mu_k$ drawn from the metric space, so as to minimize some cost function which is related to the radius of the clusters.

(1) *k-center*: Maximum radius of clusters

$$\max_j \max_{x \in C_j} d(\mu_j, x)$$

(2) *k-median*: Average radius of clusters

$$\sum_j \sum_{x \in C_j} d(\mu_j, x)$$

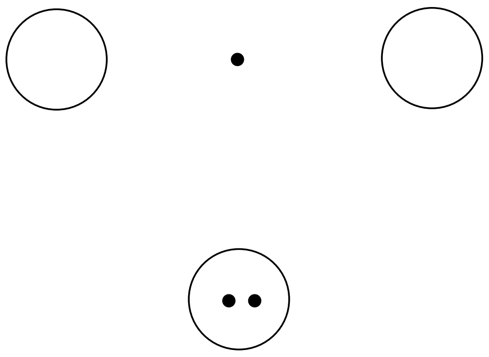

Fig. 3. The circles represent an optimal 3-clustering; all the data points lie within them. The dots are centers of a really bad clustering.

Both problems are NP-hard but have simple constant-factor approximation algorithms. For k-center, a 2-approximation was found by González [7] and by Hochbaum and Shmoys [9], and this is the best approximation factor possible [6]. For k-median there have been a series of results, of which the most recent [1] achieves an approximation ratio of $6 + \epsilon$, in time $n^{O(1/\epsilon)}$.

What does a constant-factor approximation mean for a clustering problem? Consider the scenario of Figure 3, set in the Euclidean plane. The solid lines show the real clusters, and the three dots represent the centers of a bad 3-clustering whose cost (in either measure) exceeds that of the true solution by a factor of at least ten. This clustering would therefore not be returned by the approximation algorithms we mentioned. However, EM and k-means regularly fall into local optima of this kind, and practitioners have to take great pains to try to avoid them. In this sense, constant-factor approximations avoid the worst: they are guaranteed to never do too badly. At the same time, the solutions they return can often use some fine-tuning, and local improvement procedures like EM might work well for this.

3 An approximation algorithm for hierarchical clustering

3.1 Farthest-first traversal

González [7] uses what might be called a *farthest-first traversal* of a data set as an approximation algorithm for the *k-center* problem, that of finding an optimal k-clustering under our cost function, maximum cluster radius. The idea is to pick any data point to start with, then choose the point furthest from it, then the point furthest from the first two (the distance of a point x from a set S is the usual $\min\{d(x, y) : y \in S\}$), and so on until k points are

```
Input: $n$ data points with a distance metric $d(\cdot,\cdot)$.

Pick a point and label it 1.

For $i = 2, 3, \ldots, n$
  Find the point furthest from $\{1, 2, \ldots, i-1\}$ and label it $i$.
  Let $\pi(i) = \arg\min_{j<i} d(i,j)$.
  Let $R_i = d(i, \pi(i))$.
```

Fig. 4. *Farthest-first traversal* of a data set. Take the distance from a point x to a set S to be $d(x,S) = \min_{y \in S} d(x,y)$.

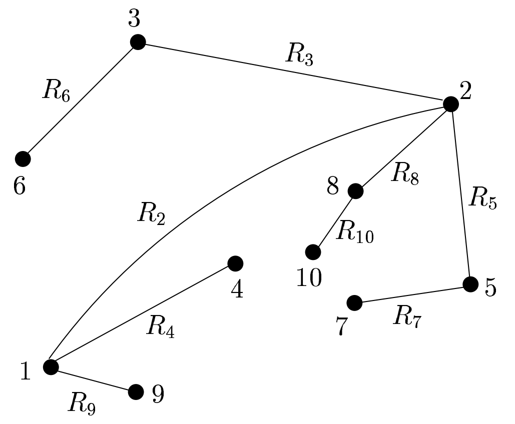

Fig. 5. A farthest-first traversal of ten data points in the plane, under Euclidean distance. The numbering is completely determined by the choice of point number one (and by the method of breaking any ties that arise).

obtained. These points are taken as cluster centers and each remaining point is assigned to the closest center. If the distance function is a metric, the resulting clustering is within a factor two of optimal. For hierarchical clustering, we will study the farthest-first traversal in detail, and will build upon it.

Starting with n points in a metric space, number *all* the points in it using a farthest-first traversal (Figure 4). For any point i, describe its closest neighbor among $1, 2, \ldots, i-1$ as its *parent*, $\pi(i)$. Let R_i be its distance to this parent,

$$R_i \;=\; d(i, \pi(i)) \;=\; d(i, \{1, 2, \ldots, i-1\}).$$

Figure 5 shows an example with a toy data set of ten points.

The algorithm of González uses points $1, 2, \ldots, k$ as centers for a k-clustering. Let $\mathbb{C}_k$ be this clustering. We begin by observing that its cost is exactly R_{k+1}.

Lemma 4 *Adopt the convention that $R_1 = \infty$ and $R_{n+1} = 0$.*

(1) $R_1 \geq R_2 \geq R_3 \geq \cdots \geq R_n$.

(2) For all k, cost$(\mathbb{C}_k) = R_{k+1}$.

PROOF. By the manner in which any point i is chosen, for all $j > i$

$$\begin{aligned} R_j &= d(j, \{1, 2, \ldots, j-1\}) \\ &\leq d(j, \{1, 2, \ldots, i-1\}) \\ &\leq d(i, \{1, 2, \ldots, i-1\}) \quad = \quad R_i, \end{aligned}$$

which immediately gives us (1). To see (2), notice that in the k-clustering, the distance from any point $i > k$ to its closest center is

$$d(i, \{1, 2, \ldots, k\}) \;\leq\; d(k+1, \{1, 2, \ldots, k\}) \;=\; R_{k+1}.$$

To illustrate our notation we repeat here the result of González [7].

Lemma 5 (González) *For any k, any k-clustering must have at least one cluster of diameter $\geq R_{k+1}$. Therefore,*

$$cost(\mathbb{C}_k) = R_{k+1} \leq 2 \cdot cost(\text{optimal } k\text{-clustering}).$$

PROOF. By construction, the points $1, 2, \ldots, k+1$ all have distance at least R_{k+1} from each other. Any k-clustering must put two of these points in the same cluster.

3.2 A first attempt at hierarchical clustering

A farthest-first traversal orders the points so that for any k, the first k points constitute the centers of a near-optimal k-clustering $\mathbb{C}_k$. Unfortunately, the n clusterings defined in this manner are not hierarchical. In Figure 5 for instance, the 2-clustering clearly puts point 6 in the cluster centered at 1, and point 3 in the cluster centered at 2. However, in the 3-clustering points 3 and 6 are grouped together.

We need a simple scheme for producing a hierarchical clustering starting with a numbering of the data points and an associated parent function π. The tree of Figure 5 is suggestive. Initially it consists of one connected component: one big cluster. Deleting an edge from the tree breaks this into two connected components, two clusters. Removing another edge will subdivide one of these two clusters, and so on.

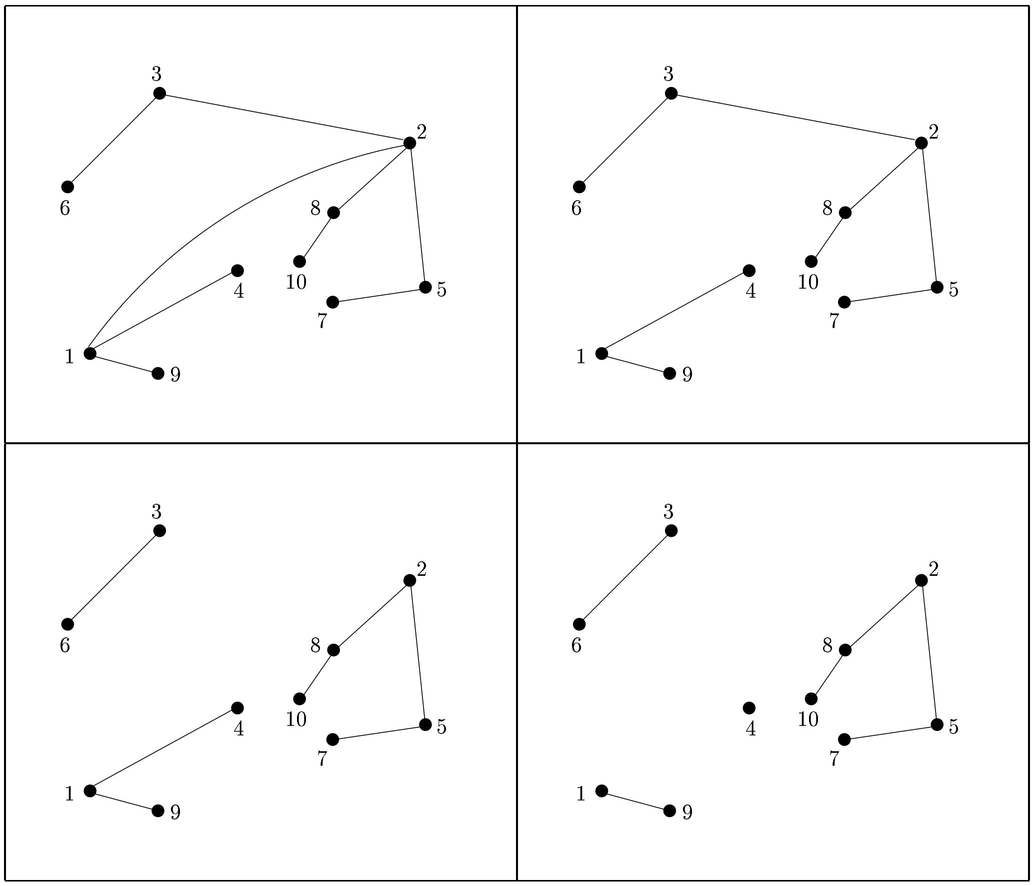

Fig. 6. A first try at hierarchically clustering the data of Figure 5. The 1-, 2-, 3-, and 4-clusterings are shown.

Definition A hierarchical clustering $\{\mathbb{C}_1^\rho, \ldots, \mathbb{C}_n^\rho\}$ based on a mapping ρ:

- Pick any function $\rho : \{2, \ldots, n\} \to \{1, 2, \ldots, n\}$ for which $\rho(i) < i$. This property is certainly satisfied by parent function π.
- The graph on nodes $\{1, 2, \ldots, n\}$, with edges $\{(i, \rho(i)) : 2 \leq i \leq n\}$, is a tree. Call it T^ρ.
- For any k, the k-clustering $\mathbb{C}_k^\rho$ is defined as follows.
  - · Remove the $k-1$ edges $(2, \rho(2)), \ldots, (k, \rho(k))$ from T^ρ.
  - · This leaves k connected components.
  - · Each cluster in $\mathbb{C}_k^\rho$ consists of the points in one of these components.

Figure 6 illustrates this for T^π. Witness that the clusterings $\{\mathbb{C}_1^\pi, \mathbb{C}_2^\pi, \ldots, \mathbb{C}_n^\pi\}$ *are* hierarchical.

Lemma 6 *Create clusterings* $\mathbb{C}_1^\rho, \ldots, \mathbb{C}_n^\rho$ *as above. Then*

(1) these clusterings are hierarchical, and

(2) for all k*, the points* $1, 2, \ldots, k$ *lie in different clusters of* $\mathbb{C}_k^\rho$*.*

PROOF. This can be seen inductively. The k-clustering is produced by removing edges $(2, \rho(2))$ through $(k, \rho(k))$ from T^ρ. The $(k+1)$-clustering is produced by further removing the edge $(k+1, \rho(k+1))$. This last operation splits the cluster (of $\mathbb{C}_k^\rho$) containing $k+1$ and $\rho(k+1)$ into two pieces. One piece contains $\rho(k+1) \leq k$; the other contains $k+1$.

3.3 Levels of granularity

The hierarchical clustering generated by tree T^π might be very poor. To get a sense of what's lacking, look again at Figure 5. Pick any node k in this tree, remove the edge $(k, \pi(k))$, and consider the connected component containing k. The nodes in this component are grouped together in the k-clustering. The immediate neighbors of k are very close to it – at most R_{k+1} away, and this in turn is at most twice the cost of the optimal k-clustering (recall Lemma 5). But other nodes in this cluster could potentially be much further away.

We will therefore construct an alternative parent function π' whose tree $T^{\pi'}$ has the following property: as you move along any path with increasing node numbers, the edge lengths are bounded by a geometrically decreasing sequence. This immediately rules out the bad effect mentioned above, and as a consequence $\text{cost}(\mathbb{C}_k^{\pi'}) \leq O(1) \cdot \text{cost}(\mathbb{C}_k)$.

We will build π' by viewing the data at certain specific levels of granularity. Let $R = R_2$; this is some rough measure of the span of the data. If we do not care about distances smaller than R, the entire data set can be summarized by the single point $\{1\}$. This is our coarsest view, and we will call it L_0, granularity level zero. Suppose we want a little more detail, but we still don't care about distances less than $R/2$. Then the data can be summarized by L_0 augmented with $L_1 = \{i : R/2 < R_i \leq R\}$. Continuing in this manner, we construct levels $L_0, L_1, L_2, \ldots$ such that every data point is within distance $R/2^j$ of $L_0 \cup L_1 \cup \cdots \cup L_j$.

Earlier we set the parent of i to be its closest neighbor amongst $\{1, 2, \ldots, i-1\}$. We now choose parents from a more restricted set: the closest point *at a lower level of granularity.* (Approximating a metric space at geometrically converging levels of granularity is key to *chaining*, a proof technique often used in empirical process theory and machine learning, e.g. [15,11].) This can be generalized to allow the granularity to be refined by a factor $\beta > 1$ possibly different from 2 between levels, and to start with granularity αR_2, for $\alpha \in [1, \beta)$. The resulting hierarchical clustering algorithm is shown in Figure 7, and its effect on our earlier example in the case $\beta = 2$ can be seen in Figure 8. For the time being, think of α as 1; it will come in handy later.

```
Input: $n$ data points with a distance metric $d(\cdot,\cdot)$.

Fix constants $\beta > 1$ and $\alpha \in [1, \beta)$.

Numbering the points
  Number the points by farthest-first traversal (Figure 4).
  For $i = 2, 3, \ldots, n$, let $R_i = d(i, \{1, 2, \ldots, i-1\})$.
  Let $R = \alpha R_2$.

Levels of granularity
  Lowest level: $L_0 = \{1\}$.
  For $j > 1$, $L_j = \{i : R/\beta^j < R_i \leq R/\beta^{j-1}\}$.

Hierarchical clustering
  Parent function: $\pi'(i)$ = closest point to $i$ at lower level of granularity.
  Return the hierarchical clustering corresponding to tree $T^{\pi'}$.
```

Fig. 7. A hierarchical clustering procedure. The main result uses $\alpha = 1, \beta = 2$.

3.4 A performance guarantee

Each data point x is assigned to a particular level of granularity; denote this level by $\text{lev}(x)$. In the notation of the algorithm, we would say $x \in L_{\text{lev}(x)}$.

Lemma 7 *Pick any point x in the data set. For all j, x lies at distance $\leq R/\beta^j$ from $L_0 \cup L_1 \cup \cdots \cup L_j$.*

PROOF. Let l be the highest-numbered point on level j. Then all points have distance $\leq R_{l+1}$ from $L_0 \cup L_1 \cup \cdots \cup L_j = \{1, 2, \ldots, l\}$, and $R_{l+1} \leq R/\beta^j$ (since $l + 1 \notin L_j$).

Corollary 8 *For any data point x,*

$$d(x, \pi'(x)) \leq \frac{R}{\beta^{lev(x)-1}}.$$

Lemma 9 *Pick any $k < n$. The k-clustering $\mathbb{C}_k^{\pi'}$ has cost at most*

$$\frac{R}{(\beta - 1)\beta^{lev(k+1)-2}} \quad \leq \quad \frac{\beta^2}{\beta - 1} \cdot R_{k+1}.$$

PROOF. We have already seen that the k-clustering puts points $1, 2, \ldots, k$ in distinct clusters; call these the centers of the clusters. Pick any $i > k$. To

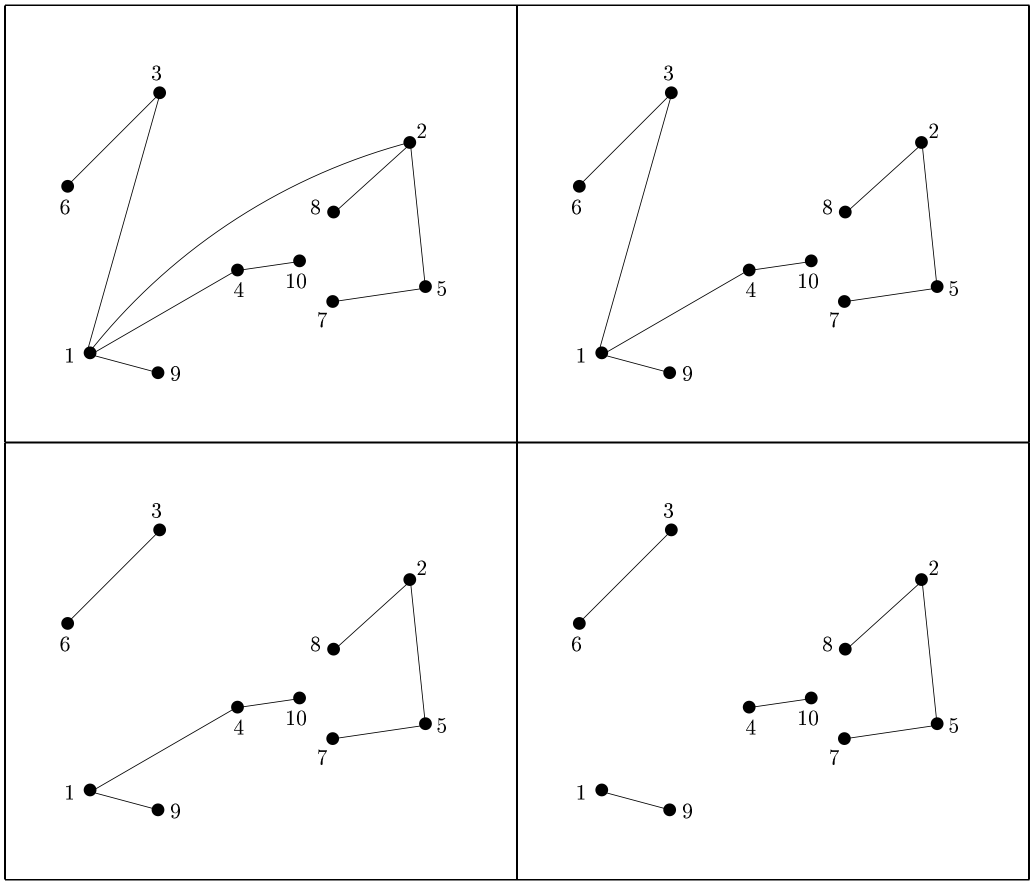

Fig. 8. A continuation of the example of Figure 6. Shown are the 1-, 2-, 3-, and 4-clusterings obtained from the modified parent function π'.

determine which cluster it belongs to, follow the parent links

$$(i_0 = i) \rightarrow (i_1 = \pi'(i_0)) \rightarrow (i_2 = \pi'(i_1)) \rightarrow \cdots$$

until some point $i_l \in \{1, 2, \ldots, k\}$ is reached. This sequence $i_0, i_1, \ldots, i_l$ is decreasing, and these points belong in the same cluster because they lie in the same connected component of $T^{\pi'}$ when edges $(2, \pi'(2)), \ldots, (k, \pi'(k))$ are removed.

To bound $d(i, i_l)$ we use the triangle inequality,

$$d(i, i_l) \ \leq \ d(i_0, i_1) + d(i_1, i_2) + \cdots + d(i_{l-1}, i_l).$$

By Corollary 8,

$$d(i, i_l) \ \leq \ \frac{R}{\beta^{\mathrm{lev}(i_0)-1}} + \frac{R}{\beta^{\mathrm{lev}(i_1)-1}} + \cdots + \frac{R}{\beta^{\mathrm{lev}(i_{l-1})-1}}.$$

Each point in the sequence $i_0, \ldots, i_l$ lies in a strictly lower level of granularity than its predecessor, so

$$d(i, i_l) \;\le\; \frac{R}{\beta^{\mathrm{lev}(i_{l-1})-1}} \left(1 + \frac{1}{\beta} + \frac{1}{\beta^2} + \cdots\right).$$

Since the path terminated once it reached $i_l \in \{1, ..., k\}$, we have $i_{l-1} > k$, which implies

$$d(i, i_l) \le \frac{R}{(\beta-1)\beta^{\mathrm{lev}(k+1)-2}} \;\le\; \frac{\beta^2}{\beta - 1} \cdot R_{k+1},$$

since $R_{k+1} > R/\beta^{\mathrm{lev}(k+1)}$.

Theorem 1 follows immediately, by setting $\beta = 2$ and applying Lemma 5. We next consider a variant in which α is chosen randomly for a fixed value of β. This trick has been used in scheduling [13] and, later, in other algorithms, and we thank Rajeev Motwani for suggesting it to us.

Lemma 10 *Choose $\alpha \overset{d}{=} \beta^{U[0,1]}$ (that is, pick a real number uniformly at random from $[0, 1]$ and then raise β to this power). For all $k < n$, the induced k-clustering has expected cost at most $\frac{\beta}{\ln \beta} \cdot R_{k+1}$.*

PROOF. Suppose $k + 1$ lies in level l; then $R/\beta^{l-1} \ge R_{k+1} > R/\beta^l$. Write $R_{k+1} = R/\beta^{((l-1)+\epsilon)}$, where $\epsilon \in [0, 1)$. This ϵ is the fractional part of

$$\log_\beta \frac{R}{R_{k+1}} \;=\; \log_\beta \frac{\alpha R_2}{R_{k+1}} \;=\; \log_\beta \frac{R_2}{R_{k+1}} + \log_\beta \alpha.$$

Since $\log_\beta \alpha$ is distributed uniformly over $[0, 1)$, so is ϵ.

By Lemma 9, the k-clustering has cost at most

$$\frac{R}{\beta^{l-1}} \cdot \frac{\beta}{\beta - 1} \;=\; \frac{\beta^{1+\epsilon}}{\beta - 1} \cdot R_{k+1}.$$

This has expected value

$$\int_0^1 \frac{\beta^{1+\epsilon}}{\beta - 1} \cdot R_{k+1}\, d\epsilon \;=\; \frac{\beta}{\ln \beta} \cdot R_{k+1}.$$

Choosing $\beta = e$ and applying Lemma 5 then gives an (expected) approximation factor of $2e \approx 5.44$. Choosing $\beta = 2$ gives a factor of about 5.77.

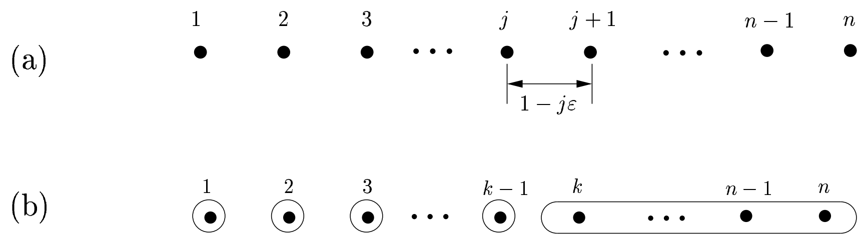

Fig. 9. (a) Data points on a line. (b) The k-clustering induced by a single-linkage heuristic.

4 Worst-case performance of standard agglomerative clustering heuristics

4.1 Single linkage

Single-linkage clustering has a performance ratio of at least k for intermediate k-clusterings, even in the simple case when the data points lie on a line. This particular kind of bad behavior is known to statisticians as *chaining* [8] (not to be confused with the aforementioned use of the same term in empirical process theory).

Consider the set of points shown in Figure 9(a). Let the distance between any two adjacent points j and $j+1$ be $1-j\varepsilon$, for some tiny $\varepsilon > 0$. Then the intermediate k-clustering found by single linkage is as shown in the bottom half of the figure. It consists of one large cluster containing $n-k+1$ points, and $k-1$ singleton clusters. The diameter of the big cluster can be made arbitrarily close to $n-k$ by setting ε small enough. On the other hand, the optimal k-clustering has clusters of diameter at most $\lceil \frac{n}{k} \rceil - 1$, for vanishingly small ε. Therefore the approximation ratio is at least k.

4.2 Average linkage

The average-linkage heuristic can create intermediate k-clusterings of cost $\log k$ times optimal. Fix any k which is a power of two. Our bad example in this case involves points in $(\log k)$-dimensional space under an L_1 metric. Again we will make use of a tiny constant $\varepsilon > 0$.

- For $j = 1, 2, \ldots, \log k$, define

$$B_j = (1-\varepsilon j) \cdot \{-1, +1\} = \{-(1-\varepsilon j), +(1-\varepsilon j)\}.$$

- Let $S = B_1 \times B_2 \times \cdots \times B_{\log k}$ be the set of vertices of a $(\log k)$-dimensional

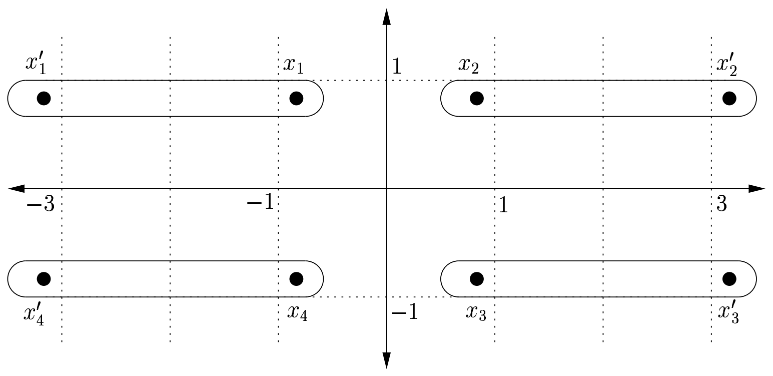

Fig. 10. A two-dimensional bad case for average-linkage clustering. The optimal 4-clustering is shown, and has clusters of diameter ≈ 2. Average-linkage will put the central four points together; these have diameter ≈ 4. By increasing the dimension to $\log k$, we find that average-linkage can return k-clusterings whose component clusters have diameter $\log k$ times optimal.

cube whose side lengths are just slightly less than two.

- Arbitrarily label the points of S as $x_1, \ldots, x_k$. Now define points $S' = \{x'_1, \ldots, x'_k\}$ as follows. For $\ell = 1, 2, \ldots, k$,
  - if x_ℓ has a positive first coordinate, then let x'_ℓ be identical to x_ℓ, but with first coordinate $+3 + \ell \cdot 2\varepsilon \log k$;
  - if x_ℓ has a negative first coordinate, then let x'_ℓ be identical to x_ℓ, but with first coordinate $-3 - \ell \cdot 2\varepsilon \log k$.
- The data set so far consists of $S \cup S'$, a total of $2k$ points. Duplicate points as necessary to get the count up to n. Figure 10 shows this data set for $k = 4$.

Lemma 11 *For the data set just defined, under the L_1 metric,*

(1) The distance between any two distinct points of S' is at least two.

(2) The distance from any point in S' to $[-1, +1]^{\log k}$ is more than two.

(3) Any two points in S which disagree on the j^{th} coordinate have distance at least $2(1 - \varepsilon j)$ between them.

PROOF. (1) Pick distinct $x'_a, x'_b \in S'$. If x_a, x_b disagree on the first coordinate then x'_a, x'_b differ by at least six on the first coordinate. If x_a, x_b differ on the j^{th} coordinate, then x'_a, x'_b differ by at least $2\varepsilon \log k$ on the first coordinate, and by $2(1 - \varepsilon j)$ on the j^{th} coordinate, giving a total of at least two.

(2) This can be seen by considering the first coordinate alone.

The closest pairs of points (once duplicates get merged) are therefore those pairs in S which disagree only on the last coordinate. The distance between such pairs is $2(1 - \varepsilon \log k)$. They get merged, and in this way S is reduced to just $k/2$ clusters, with means $B_1 \times B_2 \times \cdots \times B_{\log k - 1} \times \{0\}$. Continuing in this way, eventually all the points in S get merged into one cluster centered at the origin, while the points of S' remain untouched. During this phase, clusters getting merged always have means which are within distance two of each other, and which lie in $[-1, +1]^{\log k}$.

The k-clustering therefore includes a large cluster containing all of S. The diameter of the cluster is at least the diameter of S, namely $2 \log k - \varepsilon \log k - \varepsilon \log^2 k$.

There is a better k-clustering: $\{x_1, x_1'\}, \{x_2, x_2'\}, \ldots, \{x_k, x_k'\}$. These clusters have diameter at most $2 + 2\varepsilon k \log k + \varepsilon \log k$. Letting ε go to zero, we get an approximation ratio of $\log k$.

4.3 Complete linkage

In our counterexample for complete linkage, the data lie in $\mathbf{R} \times \mathbf{R}$, and the distance between two points (x, y) and (x', y') is defined as

$$d((x, y), (x', y')) \;=\; \mathbf{1}(x \neq x') + \log_2(1 + |y - y'|).$$

It can quickly be confirmed that this is a metric.

Assume k is a power of two for convenience. The data set consists of k clusters $S_1, S_2, \ldots, S_k$, each with k points (points can be duplicated to get the total up to n). Within each cluster, all points have the same y-coordinate, and have x-coordinates (say) $\{1, 2, \ldots, k\}$ (it doesn't matter what they are, as long as they are the same across clusters). Therefore the clusters all have diameter one. The y-spacing between clusters is shown in Figure 11 for the case $k = 4$. The y-distance between S_j and S_{j+1} is $1 - \varepsilon(\log_2 k - q)$, where 2^q is the largest power of two dividing j (which might be $2^0 = 1$).

This example is set up so that the k-clustering found by complete linkage will have clusters which (each) touch every S_j, and which therefore have diameter $\log_2 k$, as ε goes to zero.

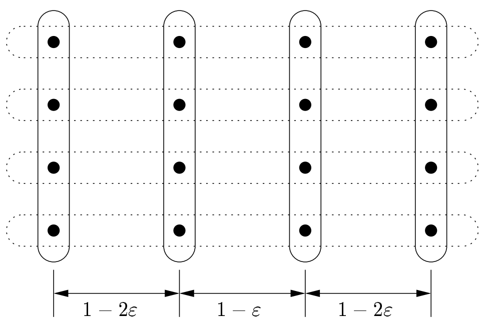

Fig. 11. A bad case for complete-linkage clustering. The norm here is unusual; see the definition. The optimal 4-clustering is shown in bold and the 4-clustering found by complete-linkage is delineated by dots.

5 Practical issues and open questions

5.1 Small values of k

It is often sufficient to guarantee good k-clusterings just for small values of k, say in the hundreds or so, or in some cases even smaller. Therefore it would be quite heartening if it turned out that our $\Omega(\log k)$ lower bound on the approximation ratio of average- and complete-linkage were actually an upper bound as well. At present no nontrivial upper bounds are known.

5.2 Other cost functions

We could reasonably have chosen the cost function of k-median (*average* distance to nearest cluster center), but picked ours instead because it is easier from a technical point of view. Plaxton has since obtained similar performance guarantees for this other cost function [14]. Can the constants in either of these analyses be improved?

5.3 Efficiency

Can our algorithm, or any of the standard heuristics we have considered, be implemented in $o(n^2)$ time for data sets of size n? Results of Borodin *et al.* [2] and Thorup [16] offer some hope here. At the same time, results of Mettu [12] indicate that at least for the k-median cost function, we cannot hope for a subquadratic algorithm which guarantees a constant-factor approximation for all k.

5.4 Hill Climbing

Is there a simple procedure for hill climbing the space of hierarchical clusterings with respect to our cost function? This would be a useful postprocessing step to improve the quality of the solutions we obtain.

Acknowledgements

The authors thank Rajeev Motwani, Werner Stuetzle, and the anonymous reviewers for their help.

References

[1] V. Arya, N. Garg, V. Khandekar, V. Pandit, A. Meyerson, and K. Munagala. Local search heuristics for k-median and facility location problems. *Proceedings of the 33rd ACM Symposium on the Theory of Computing*, 2001.

[2] A. Borodin, R. Ostrovsky, and Y. Rabani. Subquadratic approximation algorithms for clustering problems in high dimensional spaces. *Proceedings of the 31st ACM Symposium on the Theory of Computing*, 1999.

[3] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. *Proceedings of the 29th ACM Symposium on the Theory of Computing*, 1997.

[4] S. Dasgupta and L. J. Schulman. A two-round variant of em for gaussian mixtures. *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000.

[5] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95:14863–14868, 1998.

[6] T. Feder and D. Greene. Optimal algorithms for approximate clustering. *Proceedings of the 20th ACM Symposium on the Theory of Computing*, 1988.

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

[8] J. A. Hartigan. Statistical theory in clustering. *Journal of Classification*, (2):63–76, 1985.

[9] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

[10] M. J. Kearns, Y. Mansour, and A. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, 1997.

[11] P. M. Long. The complexity of learning according to two models of a drifting environment. *Machine Learning*, 37(3):337–354, 1999.

[12] R. R. Mettu. *Approximation Algorithms for NP-Hard Clustering Problems.* PhD thesis, Department of Computer Science, University of Texas at Austin, August 2002.

[13] R. Motwani, S. Phillips, and E. Torng. Non-clairvoyant scheduling. *Theoretical Computer Science*, 130:17–47, 1994.

[14] G. Plaxton. Approximation algorithms for hierarchical location problems. *Proceedings of the 35th ACM Symposium on the Theory of Computing*, 2003.

[15] D. Pollard. *Convergence of stochastic processes.* Springer, 1984.

[16] M. Thorup. Quick k-median, k-center, and facility location for sparse graphs. *International Colloquium on Automata, Languages, and Programming*, 2001.