Journal of Machine Learning Research 24 (2023) 1-36 Submitted 2/18; Revised 5/20; Published 1/23

Approximation Bounds for Hierarchical Clustering: Average Linkage, Bisecting K-means, and Local Search

Benjamin Moseley MOSELEYB@ANDREW.CMU.EDU
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Joshua R. Wang JOSHUAWANG@GOOGLE.COM
Google Research
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Editor: Vahab Mirrokni

Abstract

Hierarchical clustering is a data analysis method that has been used for decades. Despite its widespread use, the method has an underdeveloped analytical foundation. Having a well understood foundation would both support the currently used methods and help guide future improvements. The goal of this paper is to give an analytic framework to better understand observations seen in practice. This paper considers the dual of a problem framework for hierarchical clustering introduced by Dasgupta (2016). The main result is that one of the most popular algorithms used in practice, average linkage agglomerative clustering, has a small constant approximation ratio for this objective. To contrast, this paper establishes that using several other popular algorithms, including bisecting k-means divisive clustering, have a very poor lower bound on its approximation ratio for the same objective. However, we show that there are divisive algorithms that perform well with respect to this objective by giving two constant approximation algorithms. This paper is some of the first work to establish guarantees on widely used hierarchical algorithms for a natural objective function. This objective and analysis give insight into what these popular algorithms are optimizing and when they will perform well.

1. Introduction

Hierarchical clustering is a widely used method to analyze data. See Murtagh and Contreras (2012); Krishnamurthy et al. (2012); Heller and Ghahramani (2005) for an overview and pointers to relevant work. In a typical hierarchical clustering problem, one is given a set of n data points and a notion of similarity between the points. The output is a hierarchy of clusters on the input. Specifically, a dendrogram (tree) is constructed where the leaves correspond to the n input data points and the root corresponds to a cluster containing all data points. Each internal node of the tree corresponds to a cluster of the data points in its subtree. The clusters (internal nodes) become more refined as we move down the tree. The goal is to construct the tree so that these deeper clusters contain points that are relatively more similar.

There are many reasons for the popularity of hierarchical clustering, including that the number of clusters is not predetermined and that the clusters produced induce taxonomies that give meaningful ways to interpret data.

Methods used to perform hierarchical clustering are divided into two classes: agglomerative and divisive. **Agglomerative** algorithms take a bottom-up approach and are more commonly used than divisive approaches (Hastie et al., 2009). In an agglomerative method, each of the n input data points starts as its own cluster. Then iteratively, pairs of similar clusters are merged according to some appropriate notion of similarity. Perhaps the most popular definition of similarity is **average linkage** where the similarity between two clusters is defined as the average similarity between all pairs of data points in the two clusters. In average linkage agglomerative clustering the two clusters with the highest average similarity are merged at each step. Other variants are also popular. Related examples include: **single linkage**, where the similarity between two clusters is the maximum similarity between any pair of single data points in different clusters, and **complete linkage**, where the distance is the minimum similarity between any pair of single data points in different clusters.

Divisive algorithms take a top-down approach where initially all data points are placed into a single cluster. They recursively perform splits, dividing a cluster into smaller clusters that will be further subdivided. The process continues until each cluster consists of a single data point. In each step of the algorithm, the data points are partitioned such that points within each cluster are more similar than points across clusters. There are several approaches to perform divisive clustering. One example is bisecting k-means where k-means is used at each step with $k = 2$. For details on bisecting k-means, see Jain (2010).

Motivation: Hierarchical clustering has been used and studied for decades. There has been some work on theoretically quantifying the quality of the solutions produced by algorithms, such as Ackerman et al. (2012); Ackerman and Ben-David (2016); Zadeh and Ben-David (2009); Ben-David and Ackerman (2008); Dasgupta (2016). Much of this work focuses on deriving the structure of solutions created by algorithms or analytically describing desirable properties of a clustering algorithm. Though the area has been well-studied, there is no widely accepted formal problem framework. Hierarchical clustering describes a class of algorithmic methods rather than a problem with an objective function. Studying a formal objective for the problem could lead to the ability to objectively compare different methods; there is a desire for the community to investigate potential objectives, which would further support the use of current methods and guide the development of improvements.

This paper is concerned with investigating objectives for hierarchical clustering. It gives a natural objective and leverages its structural connection to average linkage agglomerative clustering to prove this algorithm obtains a constant approximation to the best possible clustering. In contrast to this positive result, single linkage, complete linkage, and bisecting k-means are shown to have superconstant (i.e. scaling with the number of data points) approximation ratios. This paper also provides some divisive algorithms that have comparable theoretical guarantees to average linkage.

Problem Formulation: Towards this paper's goal, we begin by trying to establish a formal problem framework for hierarchical clustering. Recently, Dasgupta (2016) introduced a new problem framework for hierarchical clustering. This work justified its proposed objective by establishing that for several sample problem instances, the resulting solution corresponds to what one might expect out of a desirable solution. This work spurred considerable interest and there have been several follow up papers (Charikar and Chatziafratis, 2017; Dasgupta, 2016; Roy and Pokutta, 2016).

In the problem introduced by Dasgupta (2016), the input is a set of n data points as input along with, for every pair of points i and j, a (nonnegative) weight $w_{i,j}$ denoting their similarity. The higher the weight, the more similar the edge endpoints. This can be represented as a weighted complete graph G. The output is a (full) *binary* tree where the leaves of the tree correspond to the input data points. For each pair of points i and j, let $T[i \vee j]$ denote the subtree rooted at i and j's least common ancestor. Let $\texttt{leaves}(T[i \vee j])$ denote the set of leaves in the tree $T[i \vee j]$. The goal is to construct the tree such that the cost $\text{cost}_G(T) := \sum_{i,j \in [n]} w_{ij} |\texttt{leaves}(T[i \vee j])|$ is minimized. Intuitively, this objective enforces that more similar points i and j should have a lower least common ancestor in the tree because the weight $w_{i,j}$ is large and so additions to $|\texttt{leaves}(T[i \vee j])|$ are severely penalized.

For this objective, several approximation algorithms have been given (Charikar and Chatziafratis, 2017; Dasgupta, 2016; Roy and Pokutta, 2016). There is a divisive clustering algorithm with an approximation ratio of $O(\sqrt{\log n})$ (Charikar and Chatziafratis, 2017). In particular, the algorithm gives an $O(\alpha_n)$-approximation where α_n is the approximation ratio of the sparsest cut subroutine (Charikar and Chatziafratis, 2017). Furthermore, assuming the Small-Set Expansion Hypothesis, every algorithm is a $\omega(1)$-approximation (Charikar and Chatziafratis, 2017). The current best known bound on α_n is $O(\sqrt{\log n})$ (Arora et al., 2009). Unfortunately, this conclusion misses one of our key goals in trying to establish an objective function. While the algorithms and analysis are ingenious, none of the algorithms with theoretical guarantees are from the class of algorithms used in practice. Furthermore, in this work we establish that several commonly used algorithms (namely, single linkage, average linkage, complete linkage, and bisecting k-means) have bad negative bounds on their solution quality with respect to Dasgupta's cost objective.

Hence this question still looms: are there strong theoretical guarantees for practical algorithms? It seems that Dasgupta's cost objective may not be ideal for this task; is there another natural objective that admits solutions that are provably close to optimal?

Results: In this paper, we consider an objective function motivated by the objective introduced by Dasgupta (2016). For a given tree T let $|\texttt{leaves-outside}(T[i \vee j])|$ be the total number of leaves that *are not* in the subtree rooted at the least common ancestor of i and j. The Dasgupta objective considers constructing a tree T to minimize the cost function $\text{cost}_G(T) := \sum_{i,j \in [n]} w_{ij} |\texttt{leaves}(T[i \vee j])|$. This paper considers the *dual* problem where T is constructed to maximize the reward function $\text{reward}_G(T) := \sum_{i,j \in [n]} w_{ij} |\texttt{leaves-outside}(T[i \vee j])| = (n \sum_{i,j \in [n]} w_{i,j}) - \text{cost}_G(T)$. It is important to observe that the optimal clustering *is the same for both objectives.* Due to this, all the examples given in Dasgupta (2016) motivating their objective by showing desirable structural properties of the optimal solution also apply to the objective considered in this paper. Our objective can be interpreted similarly to that in Dasgupta (2016). In particular, similar points i and j should be located lower in the tree as to maximize $|\texttt{leaves-outside}(T[i \vee j])|$.

This paper gives a thorough investigation of this new problem framework by analyzing several algorithms under this objective. The main result is establishing that average linkage clustering is a $\frac{1}{3}$-approximation. This result gives theoretical justification for the use of average linkage clustering and, additionally, this shows that the objective considered is tractable since it admits $\Omega(1)$-approximations. It also suggests that the objective captures a component of what average linkage optimizes for.

This paper then seeks to understand what other algorithms are good for this objective. We establish that single linkage, complete linkage, and bisecting k-means is no better than a polynomial-approximations. This establishes that these methods are *very* poor for the objective considered, and that these methods optimize for something distinct than what average linkage optimizes for.

Given these negative results, we question whether there are divisive algorithms that optimize for our objective. We answer this question affirmatively by giving a local search strategy that obtains a $\frac{1}{3}$-approximation as well as showing that randomly partitioning is exactly a $\frac{1}{3}$-approximation.

Related Work (This Cost Function): Charikar et al. (2019a) gave a tight lower bound construction for average linkage under this paper's reward objective, which matches our upper bound. They same work also gives an SDP relaxation technique which has a better approximation ratio. Charikar et al. (2019b) study our objective for Euclidean data and shows a better bound for average linkage under this data model. Menon et al. (2019) consider our objective in the setting where the data points arrive one by one.

Related Work (Other Cost Functions): Recently a contemporaneous paper (Cohen-Addad et al., 2017) done independently has been published. This paper considers a class of objectives motivated by the work of Dasgupta (2016). For their objective, they also derive positive results for average linkage clustering and additionally give axiomatic properties that are desirable in an objective for hierarchical clustering. Ma and Dhavala (2018) consider combining Dasgupta's objective function with prior knowledge about the data set. Wang and Wang (2018) suggests an alternate objective with the goal of comparing the clusterability across different input graphs, rather than just different clusterings for a single input graph. Both Chierchia and Perret (2019) and Monath et al. (2019) propose continuous objective functions with the goal of applying gradient descent. Lattanzi et al. (2019) and Abboud et al. (2019) both strive to make more efficient (parallelized and raw runtime, respectively) versions of classical clustering algorithms, and judge the resulting quality via comparing iteration-by-iteration against what the classical algorithm would have done. Wang and Moseley (2020) propose an objective function for which bisecting k-means achieves a constant approximation.

2. Preliminaries

In this section, we give preliminaries including a formal definition of the problem considered and basic building blocks for later algorithm analysis.

In the **Reward Hierarchical Clustering Problem** there are n input data points given as a set V. There is a weight $w_{i,j} \geq 0$ between each pair of points i and j denoting their similarity, represented as a complete graph G. The output of the problem is a rooted tree T where the leaves correspond to the data points and the internal nodes of the tree correspond to clusters of the points in the subtree. We will use the indices $1, 2, \dots n$ to denote the leaves of the tree. For two leaves i and j, let $T[i \vee j]$ denote the subtree rooted at the least common ancestor of i and j and let the set `leaves-outside`$(T[i \vee j])$ denote the number of leaves in T that are not in $T[i \vee j]$. The objective is to construct T to maximize the reward $\text{reward}_G(T) = \sum_{i \in [n]} \sum_{j \neq i \in [n]} w_{i,j} |\texttt{leaves-outside}(T[i \vee j])|$.

We make no assumptions on the structure of the optimal tree T. However, one optimal tree is a binary tree, so we may restrict the solution to binary trees without loss of generality. This can be seen as follows. It was shown in Dasgupta (2016) that the optimal solution for the $\text{cost}_G(T)$ is binary. Knowing that $\text{cost}_G(T) + \text{reward}_G(T) = n \sum_{i,j} w_{i,j}$, so the optimal solution to minimizing $\text{cost}_G(T)$ is the same as the optimal solution to maximizing $\text{reward}_G(T)$. Thus, there is an optimal solution for the $\text{reward}_G(T)$ objective that is binary.

To see this, let `leaves`$(T[i \vee j])$ be the set of leaves in $T[i \vee j]$ and $\text{cost}_G(T) := \sum_{i,j} w_{ij} |\texttt{leaves}(T[i \vee j])|$. The objective considered in Dasgupta (2016) focuses on minimizing $\text{cost}_G(T)$. We note than $\text{cost}_G(T) + \text{reward}_G(T) = n \sum_{i,j} w_{i,j}$, so the optimal solution to minimizing $\text{cost}_G(T)$ is the same as the optimal solution to maximizing $\text{reward}_G(T)$. In Dasgupta (2016) it was shown that the optimal solution for any input is a binary tree.

As mentioned, there are two common types of algorithms for hierarchical clustering: agglomerative (bottom-up) algorithms and divisive (top-down) algorithms. In an agglomerative algorithm, each vertex $v \in V$ begins in separate cluster, and each iteration of the algorithm chooses two clusters to merge into one. In a divisive algorithm, all vertices $v \in V$ begin in a single cluster, and each iteration of the algorithm selects a cluster with more than one vertex and partitions it into two small clusters.

In this section, we present some basic techniques which we later use to analyze the effect each iteration has on the reward. It will be convenient for us to think of the weight function as taking in two vertices instead of an edge, i.e. $w : V \times V \to \mathbb{R}^{\geq 0}$. This is without loss of generality, because we can always set the weight of any non-edge to zero (e.g. $w_{vv} = 0 \quad \forall v \in V$).

To bound the performance of an algorithm it suffices to bound $\text{reward}_G(T)$ and $\text{cost}_G(T)$ since $\text{reward}_G(T) + \text{cost}_G(T) = n \sum_{i,j} w_{i,j}$. Further, let T^* denote the optimal hierarchical clustering. Then its reward is at most $\text{reward}_G(T^*) \leq (n-2) \sum_{ij} w_{ij}$. This is because any edge ij can have at most $(n-2)$ non-leaves for its subtree $T[i \vee j]$; i and j are always leaves.

2.1 Analyzing Agglomerative Algorithms

In this section, we discuss a method for bounding the performance of an agglomerative algorithm. When an agglomerative algorithm merges two clusters A, B, this determines the least common ancestor for any pair of nodes i, j where $i \in A$ and $j \in B$. Given this, we define the reward gain *due to* merging A and B as, $\text{merge-rew}_G(A, B) := (n - |A| - |B|) \sum_{a \in A, b \in B} w_{ab}$.

Notice that the final reward $\text{reward}_G(T)$ is exactly the sum over iterations of the reward gains, since each edge is counted exactly once: when its endpoints are merged into a single cluster. Hence, $\text{reward}_G(T) = \sum_{\text{merges } A, B} \text{merge-rew}_G(A, B)$.

We next define the *cost* of merging A and B as the following. This is the potential reward lost by merging A and B; reward that can no longer be gained after A and B are merged, but was initially possible. Define, $\text{merge-cost}_G(A, B) := |B| \sum_{a \in A, c \in [n] \setminus (A \cup B)} w_{ac} + |A| \sum_{b \in B, c \in [n] \setminus (A \cup B)} w_{bc}$.

The total cost of the tree T, $\text{cost}_G(T)$, is exactly the sum over iterations of the cost increases, plus an additional $2\sum_{ij} w_{ij}$ term that accounts for each edge being counted towards its own endpoints. We can see why this is true if we consider a pair of vertices $i, j \in [n]$ in the final hierarchical clustering T. If at some point a cluster containing i is merged with a third cluster before it gets merged with the cluster containing j, then the number of leaves in $T[i \vee j]$ goes up by the size of the third cluster. This is exactly the quantity captured by our cost increase definition. Aggregated over all pairs i, j this is the following, $\text{cost}_G(T) = \sum_{i,j\in[n]} w_{ij}|\texttt{leaves}(T[i \vee j])| = 2\sum_{i,j\in[n]} w_{ij} + \sum_{\text{merges } A,\, B} \text{merge-cost}_G(A, B)$.

2.2 Analyzing Divisive Algorithms

Similar reasoning can be used for divisive algorithms. The following are reward gain and cost increase definitions for when a divisive algorithm partitions a cluster into two clusters A, B. Define, $\text{split-rew}_G(A, B) := |B|\sum_{a,a'\in A} w_{aa'} + |A|\sum_{b,b'\in B} w_{bb'}$ and $\text{split-cost}_G(A, B) := (|A| + |B|)\sum_{a\in A, b\in B} w_{ab}$.

Consider the reward gain. For $a, a' \in A$ we are now guaranteed that when the nodes in B are split from A then every node in B will not be a leaf in $T[a \vee a']$ (and a symmetric term for when they are both in B). On the cost side, the term counts the cost of any pairs $a \in A$ and $b \in B$ that are now separated since we now know their subtree $T[i \vee j]$ has exactly the nodes in $A \cup B$ as leaves.

3. A Theoretical Guarantee for Average Linkage Agglomerative Clustering

In this section, we present the main result, a theoretical guarantee on average linkage clustering. See Murtagh and Contreras (2012) for details and background on this widely used algorithm. The formal definition of the algorithm is given in the following pseudocode. The main idea is that initially all n input points are in their own cluster and the algorithm recursively merges clusters until there is one cluster. In each step, the algorithm merges the clusters A and B such that the pair maximizes the average distances of points between the two clusters, $\frac{1}{|A||B|}\sum_{a\in A, b\in B} w_{ab}$.

```
Data: Vertices $V$, weights $w : E \to \mathbb{R}^{\geq 0}$
Initialize clusters $\mathcal{C} \leftarrow \cup_{v\in V}\{v\}$;
while $|\mathcal{C}| \geq 2$ do
    Choose $A, B \in \mathcal{C}$ to maximize $\bar{w}(A, B) := \frac{1}{|A||B|}\sum_{a\in A, b\in B} w_{ab}$;
    Set $\mathcal{C} \leftarrow \mathcal{C} \cup \{A \cup B\} \setminus \{A, B\}$;
end
```

Algorithm 1: Average Linkage

The following theorem establishes that this algorithm is only a small constant factor away from optimal.

Theorem 1 *Consider a graph $G = (V, E)$ with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$. Let the hierarchical clustering T^* be an optimal solution maximizing of $\text{reward}_G(\cdot)$ and let T be the hierarchical clustering returned by Algorithm 1. Then, $\text{reward}_G(T) \geq \frac{1}{3}\text{reward}_G(T^*)$.*

Proof Consider an iteration of Algorithm 1. Let the current clusters be in the set $\mathcal{C}$, and the algorithm chooses to merge clusters A and B from $\mathcal{C}$. When doing so, the algorithm attains a reward gain of the following. Let $\bar{w}(A,B) = \frac{1}{|A||B|}\sum_{a\in A, b\in B} w_{ab}$ be the average weight of an edge between points in A and B.

$$\begin{aligned}\text{merge-rew}_G(A,B) = (n-|A|-|B|)\sum_{a\in A,b\in B} w_{ab} &= \sum_{C\in\mathcal{C}\setminus\{A,B\}} |C| \sum_{a\in A,b\in B} w_{ab}\\ &= \sum_{C\in\mathcal{C}\setminus\{A,B\}} |C||A||B|\bar{w}(A,B)\end{aligned}$$

while at the same time incurring a cost increase of:

$$\begin{aligned}\text{merge-cost}_G(A,B) &= |B|\sum_{a\in A, c\in[n]\setminus(A\cup B)} w_{ac} + |A|\sum_{b\in B, c\in[n]\setminus(A\cup B)} w_{bc}\\ &= |B|\sum_{C\in\mathcal{C}\setminus\{A,B\}}\sum_{a\in A,c\in C} w_{ac} + |A|\sum_{C\in\mathcal{C}\setminus\{A,B\}}\sum_{b\in B,c\in C} w_{bc}\\ &= \sum_{C\in\mathcal{C}\setminus\{A,B\}} |B||A||C|\bar{w}(A,C) + \sum_{C\in\mathcal{C}\setminus\{A,B\}} |A||B||C|\bar{w}(B,C)\\ &\le \sum_{C\in\mathcal{C}\setminus\{A,B\}} |B||A||C|\bar{w}(A,B) + \sum_{C\in\mathcal{C}\setminus\{A,B\}} |A||B||C|\bar{w}(A,B)\\ &= 2\cdot\text{merge-rew}_G(A,B)\end{aligned}$$

Intuitively, every time this algorithm loses two units of potential it cements the gain of one unit of potential, which is why it is a $\frac{1}{3}$-approximation. Formally:

$$\begin{aligned}\text{cost}_G(T) = 2\sum_{i,j} w_{ij} + \sum_{\text{merges } A,B}\text{merge-cost}_G(A,B) &\le 2\sum_{i,j} w_{ij} + 2\cdot\sum_{\text{merges } A,B}\text{merge-rew}_G(A,B)\\ &\le 2\sum_{i,j} w_{ij} + 2\cdot\text{reward}_G(T)\end{aligned}$$

Now the reward can be bounded as follows.

$$\begin{aligned}\text{reward}_G(T) &\ge n\sum_{ij} w_{ij} - \text{cost}_G(T) \ge n\sum_{ij} w_{ij} - 2\sum_{i,j} w_{ij} - 2\cdot\text{reward}_G(T)\\ \text{reward}_G(T) &\ge \frac{n-2}{3}\sum_{ij} w_{ij} \ge \frac{1}{3}\text{reward}_G(T^*)\end{aligned}$$

where the last step follows from the fact that it is impossible to have more than $n-2$ leaves-outside. ■

As an aside, we note that it is not possible for all inequalities in the proof to be simultaneously tight, and hence average linkage is actually slightly better (inverse polynomial in n) than a 1/3 approximation. The reasoning is as follows.

Consider the very first merge made by average-linkage, where it picks the highest weight edge of our graph G. Suppose this edge (u,v) has weight W. The proof above relates merge-cost$_G(\{u\},\{v\})$ and merge-rew$_G(\{u\},\{v\})$. We plug known quantities into the definition of merge reward.

$$\text{merge-rew}_G(\{u\},\{v\}) = (n-2)W$$

We can also unroll the definition of merge cost for this simple merge.

$$\text{merge-cost}_G(\{u\},\{v\}) = \sum_{z\in V\setminus\{u,v\}} (w_{uz} + w_{vz})$$

We know from our proof that the merge cost is between zero and $2(n-2)W$. Let's express the cost as $\alpha(n-2)W$, where $\alpha \in [0,2]$. We handle the case where α is high as follows. The average triangle containing (u,v) has $(\alpha+1)W$ total weight. Our reward objective has an alternate form in terms of triangles, as observed by Charikar et al. (2019b).

$$\text{reward}_G(T) = \sum_{i<j<k} \left(w_{ij}\mathbb{1}[\mathcal{E}_{ij}^k] + w_{jk}\mathbb{1}[\mathcal{E}_{jk}^i] + w_{ik}\mathbb{1}[\mathcal{E}_{ik}^j] \right)$$
$$\text{reward}_G(T) + \text{cost}_G(T) = \sum_{i<j<k} (w_{ij} + w_{jk} + w_{ik}) + 2\sum_{i<j} w_{ij}$$

Following the notation in that paper, $\mathcal{E}_{jk}^i$ denotes the event that among the three nodes $\{i,j,k\}$, i is split from the others first. For the average triangle containing (u,v), this means that even the optimal hierarchical clustering can only score W out of a possible $(\alpha+1)W$ total. We can combine this missing αW value for each of our $(n-2)$ triangles with the final approximation statement to get:

$$\begin{aligned}\text{reward}_G(T) &\geq \frac{1}{3}(\text{reward}_G(T^*) + \text{cost}_G(T^*) - 2\sum_{i<j} w_{ij}) \\ &\geq \frac{1}{3}(\text{reward}_G(T^*) + \alpha(n-2)W)\end{aligned}$$

Now, we handle the case where α is low. In this case, we gain $(n-2)W$ merge reward while only incurring $\alpha(n-2)W$ merge cost. This puts a little ahead in our telescoping inequality:

$$\text{cost}_G(T) \leq 2\sum_{i<j} w_{ij} + 2\text{reward}_G(T) + (2-\alpha)(n-2)W$$

Hence we arrive at a slightly better conclusion:

$$\text{reward}_G(T) \geq \frac{1}{3}\text{reward}_G(T^*) + \frac{2-\alpha}{3}(n-2)W$$

We conclude by noting that our two guarantees are balanced when $\alpha = 1$ and that $(n-2)W \geq \frac{2}{n(n-1)} \cdot \text{reward}_G(T^*)$.

$$\begin{aligned}\text{reward}_G(T) &\geq \frac{1}{3}\text{reward}_G(T^*) + \frac{1}{3}(n-2)W \\ &\geq \left(\frac{1}{3} + \frac{2}{n(n-1)}\right)\text{reward}_G(T^*)\end{aligned}$$

4. Divisive Local-Search

In this section, we develop a simple local search algorithm and bound its approximation ratio. The local search algorithm takes as input a cluster C and divides it into two clusters A and B to optimize a local objective: the split reward. In particular, initially $A = B = \emptyset$. Each node in C is added to A or B uniformly at random.

Local search is run by moving individual nodes between A and B. In a step, any point $i \in A$ (resp. B) is added to B (resp. A) if $\sum_{j,l \in A; j,l \neq i} w_{j,l} + (|A| - 1) \sum_{j \in B} w_{i,j} > \sum_{j,l \in B} w_{j,l} + |B| \sum_{j \in A, j \neq i} w_{i,j}$ (resp. $\sum_{j,l \in B; j,l \neq i} w_{j,l} + (|B| - 1) \sum_{j \in A} w_{i,j} > \sum_{j,l \in A} w_{j,l} + |A| \sum_{j \in B, j \neq i} w_{i,j}$). This states that a point is moved to another set if the objective increases. The algorithm performs these local moves until there is no node that can be moved to improve the objective.

```
Data: Vertices $V$, weights $w : E \to \mathbb{R}^{\geq 0}$
Initialize clusters $\mathcal{C} \leftarrow \{V\}$;
while some cluster $C \in \mathcal{C}$ has more than one vertex do
    Let $A, B$ be a uniformly random 2-partition of $C$;
    Run local search on $A, B$ to maximize $|B| \sum_{a,a' \in A} w_{aa'} + |A| \sum_{b,b' \in B} w_{bb'}$,
      considering just moving a single node;
    Set $\mathcal{C} \leftarrow \mathcal{C} \cup \{A, B\} \setminus \{C\}$;
end
```

Algorithm 2: Divisive Local-Search

In the following theorem, we show that the algorithm is arbitrarily close to a $\frac{1}{3}$ approximation.

Theorem 2 *Consider a graph $G = (V, E)$ with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$. Let the hierarchical clustering T^* be the optimal solution of $\mathrm{reward}_G(\cdot)$ and let T be the hierarchical clustering returned by Algorithm 2. Then, $\mathrm{reward}_G(T) \geq \frac{(n-6)}{(n-2)} \frac{1}{3} \mathrm{reward}_G(T^*)$.*

Proof Since we know that $\mathrm{reward}_G(T^*) \leq (n-2) \sum_{ij} w_{ij}$, it suffices to show that $\mathrm{reward}_G(T) \geq \frac{1}{3}(n-2) \sum_{ij} w_{ij}$. We do this by considering possible local moves at every step.

Consider any step of the algorithm and suppose the algorithm decides to partition a cluster into A, B. As stated in the algorithm, its local search objective value is $OBJ = |B| \sum_{a,a' \in A} w_{aa'} + |A| \sum_{b,b' \in B} w_{bb'}$. Assume without loss of generality that $|B| \geq |A|$, and consider the expected local search objective OBJ' value for moving a random node from B to A. Note that the new local search objective value is at most what the algorithm obtained, i.e. $OBJ' \leq OBJ$:

$$\begin{aligned}
\mathrm{E}[OBJ'] &= (|B|-1)\left[\sum_{a,a'\in A} w_{aa'} + \frac{1}{|B|}\sum_{a\in A, b\in B} w_{ab}\right] + (|A|+1)\left[\frac{\binom{|B|-1}{2}}{\binom{|B|}{2}}\sum_{b,b'\in B} w_{bb'}\right] \\
&= (|B|-1)\left[\sum_{a,a'\in A} w_{aa'} + \frac{1}{|B|}\sum_{a\in A, b\in B} w_{ab}\right] + (|A|+1)\left[\frac{|B|-2}{|B|}\sum_{b,b'\in B} w_{bb'}\right] \\
&= (|B|-1)\sum_{a,a'\in A} w_{aa'} + \frac{|B|-1}{|B|}\sum_{a\in A, b\in B} w_{ab} + (|A|+1)\left[(1-\frac{2}{|B|})\sum_{b,b'\in B} w_{bb'}\right] \\
&= OBJ - \sum_{a,a'\in A} w_{aa'} + \frac{|B|-1}{|B|}\sum_{a\in A, b\in B} w_{ab} + (-\frac{2|A|}{|B|} + 1 - \frac{2}{|B|})\sum_{b,b'\in B} w_{bb'}
\end{aligned}$$

But since there are no improving moves we know the following.

$$0 \geq \mathrm{E}[OBJ'] - OBJ = -\sum_{a,a'\in A} w_{aa'} + \frac{|B|-1}{|B|}\sum_{a\in A, b\in B} w_{ab} - \frac{2|A|-|B|+2}{|B|}\sum_{b,b'\in B} w_{bb'}$$

Rearranging terms and multiplying by $|B|$ yields the following.

$$(|B|-1)\sum_{a\in A, b\in B} w_{ab} \leq |B|\sum_{a,a'\in A} w_{aa'} + (2|A|-|B|+2)\sum_{b,b'\in B} w_{bb'}$$

We now consider three cases. Either (i) $|B| \geq |A|+2$, (ii) $|B| = |A|+1$, or (iii) $|B| = |A|$. Case (i) is straightforward:

$$\begin{aligned}
\left(\frac{|B|-1}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B) \\
\frac{1}{2}\text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B)
\end{aligned}$$

In case (ii), we use the fact that $(x+2)(x-2) \leq (x+1)(x-1)$ to simplify:

$$\begin{aligned}
\left(\frac{|B|-1}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \left(\frac{|A|+1}{|A|}\right)\text{split-rew}_G(A,B) \\
\left(\frac{|B|-1}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \left(\frac{|B|+2}{|B|+1}\right)\text{split-rew}_G(A,B) \\
\left(\frac{|B|+1}{|B|+2}\right)\left(\frac{|B|-1}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B) \\
\left(\frac{|B|-2}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B) \\
\left(\frac{1}{2} - \frac{1.5}{|A|+|B|}\right)\text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B)
\end{aligned}$$

Case (iii) proceeds similarly; we now use the fact that $(x+2)(x-3) \leq (x)(x-1)$ to simplify:

$$\begin{aligned}
\left(\frac{|B|-1}{|A|+|B|}\right) \text{split-cost}_G(A,B) &\leq \left(\frac{|A|+2}{|A|}\right) \text{split-rew}_G(A,B) \\
\left(\frac{|B|-1}{|A|+|B|}\right) \text{split-cost}_G(A,B) &\leq \left(\frac{|B|+2}{|B|}\right) \text{split-rew}_G(A,B) \\
\left(\frac{|B|}{|B|+2}\right)\left(\frac{|B|-1}{|A|+|B|}\right) \text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B) \\
\left(\frac{|B|-3}{|A|+|B|}\right) \text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B) \\
\left(\frac{1}{2} - \frac{3}{|A|+|B|}\right) \text{split-cost}_G(A,B) &\leq \text{split-rew}_G(A,B)
\end{aligned}$$

Hence we have shown that for each step of our algorithm, the split reward is at least $(\frac{1}{2} - \frac{3}{|A|+|B|})$ times the split cost. We rewrite this inequality and then sum over all iterations:

$$\begin{aligned}
\text{split-rew}_G(A,B) &\geq \frac{1}{2}\text{split-cost}_G(A,B) - 3 \sum_{a\in A, b\in B} w_{ab} \\
\text{reward}_G(T) &\geq \frac{1}{2}\text{cost}_G(T) - 3 \sum_{i,j\in[n]} w_{ij} \\
&= \frac{1}{2}\left(n \sum_{i,j\in[n]} w_{ij} - \text{reward}_G(T)\right) - 3 \sum_{i,j\in[n]} w_{ij} \\
\frac{3}{2}\text{reward}_G(T) &\geq \frac{n-6}{2} \sum_{i,j\in[n]} w_{ij} \\
\text{reward}_G(T) &\geq \frac{n-6}{3} \sum_{i,j\in[n]} w_{ij}
\end{aligned}$$

This is what we wanted to prove. ■

We note that it is possible to improve the loss in terms of n to $\frac{n-4}{n-2}$ by instead considering the local search objective $(|B|-1)\sum_{a,a'\in A} w_{aa'} + (|A|-1)\sum_{b,b'\in B} w_{bb'}$.

5. Random Hierarchical Clustering

In this section, we bound the performance of a random divisive algorithm. In each step, the algorithm is given a cluster and divides the points into two clusters A and B where a point is added in each step uniformly at random. We show that this algorithm is a $\frac{1}{3}$-approximation to our reward function and further this is tight.

Theorem 3 *Consider a graph $G=(V,E)$ with nonnegative edge weights $w: E \to \mathbb{R}^{\geq 0}$. Let the hierarchical clustering T^* be a maximizer of $\text{reward}_G(\cdot)$ and let T be the hierarchical*

```
**Data:** Vertices $V$, weights $w : E \to \mathbb{R}^{\geq 0}$
Initialize clusters $\mathcal{C} \leftarrow \{V\}$;
while some cluster $C \in \mathcal{C}$ has more than one vertex do
    Let $A, B$ be a uniformly random 2-partition of $C$;
    Set $\mathcal{C} \leftarrow \mathcal{C} \cup \{A, B\} \setminus \{C\}$;
end
```

Algorithm 3: Random Hierarchical Clustering

clustering returned by Algorithm 3. Then:

$$\mathrm{E}[\mathit{reward}_G(T)] \geq \frac{1}{3}\mathit{reward}_G(T^*)$$

Proof We begin by pretending that A or B empty is a valid partition of C, and address this detail at the end of the proof. If so, we can generate A, B with the following random process: for each vertex $v \in C$, flip a fair coin to decide if it goes into A or into B.

Now, consider an edge $(i,j) \in E$. The algorithm will score a reward of $w_{ij}|\texttt{leaves-outside}(T[i \vee j])|$. Thus, we need to determine the expected value of $|\texttt{leaves-outside}(T[i \vee j])|$. How often does one of the $n-2$ other nodes besides i and j become a nonleaf of $T[i \vee j]$? Fix all all coin flips made for i and let $k \neq i, j$ be a point. The point k will become a nonleaf if j matches more coin flips than k does. The number of matched coin flips is a geometric random variable with parameter $1/2$. There is a $1/2$ chance of matching for zero coin flips, a $1/4$ chance of matching for one coin flip, and so on. Let h be the number of edges on the path from i to the root of the tree. The probability of equality is $\sum_{i=1}^{h} \frac{1}{2^{2i}} = 1/4 + 1/16 + 1/64 + \cdots = 1/3$ for sufficiently large h. By symmetry, the remaining $2/3$ probability is split bewteen j matching for more and k matching for more. Hence each of the other $n-2$ nodes k has exactly a $1/3$ chance of being a nonleaf. As a result,

$$\mathrm{E}[\mathrm{reward}_G(T)] = \frac{n-2}{3}\sum_{ij} w_{ij} \geq \frac{1}{3}\mathrm{reward}_G(T^*)$$

since it is impossible to have more than $n-2$ nonleaves.

Finally, we address the possibility of A or B being empty. This is equivalent to a node in T having a single child. In this case, $\mathrm{reward}_G(T)$ is unchanged if we merge the node with that child, since this does not change $\texttt{leaves}(T[i \vee j])$ for any edge (i,j). Hence if A or B is empty we can safely redraw. Hence our random process is equivalent to uniformly drawing over all partitions. This completes the proof. ■

We now establish that this is tight.

Lemma 4 *There exists a graph $G = (V, E)$ with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$, such that if the hierarchical clustering T^* is an optimal solution of $\mathit{reward}_G(\cdot)$ and T is the hierarchical clustering returned by Algorithm 3,*

$$\mathrm{E}[\mathit{reward}_G(T)] = \frac{1}{3}\mathit{reward}_G(T^*)$$

Algorithm	This paper's Reward Objective	Dasgupta's Cost Objective
Single Linkage	Flail: $\frac{1}{\Omega(n^{1/3})}$ [Thm. 10]	Line: $\Omega(\frac{n}{\log n})$ [Thm. 11]
Average Linkage	Clique Star: $(1/2+\delta)$ [Thm. 12]	Clique Star: $\Omega(n^{1/3})$ [Thm. 13]
Complete Linkage	Double Star: $\frac{1}{\Omega(n)}$ [Thm. 14]	Line: $\Omega(\frac{n}{\log n})$ [Thm. 15]
Bisecting k-Means	Double Star: $\frac{1}{\Omega(\sqrt{n})}$ [Thm. 16]	Cycle Star: $\Omega(\sqrt{n})$ [Thm. 17]

Table 1: The base counterexample graphs we use and the approximation bounds they yield.

Proof In the proof of Lemma 3, we showed that

$$\mathrm{E}[\mathrm{reward}_G(T)] = \frac{n-2}{3}\sum_{ij} w_{ij}.$$

This naturally suggests a tight example: any graph where the optimal hierarchical clustering T^* can capture all edges $(i,j) \in E$ with non-zero weight using only clusters of size 2. In other words, in any graph where the edges form a matching, the bound is tight. ■

6. Negative Results

This section gives negative lower bound proofs for the algorithms single linkage, average linkage, complete linkage, and bisecting k-means. All of the proofs in this section follow a similar structure:

1. Construct a base counterexample graph.
2. Increase several edge weights in the base graph by multiples of a constant $\epsilon > 0$ in order to ensure the algorithm breaks-ties in favor of the wrong decision.
3. Upper-bound the quality of the algorithm's hierarchical clustering, T.
4. The performance of the optimal solution is lower-bounded by giving a hierarchical clustering solution T'. This solution will be far superior to the algorithm's solution.

In steps (3) and (4), it will typically be the case that the analysis breaks edge weights into a base component and a tie-breaking component, which come from steps (1) and (2) respectively.

We begin by defining all of the base counterexample graphs that will be used. Then the negative results for each of the algorithms are given in the following subsections.

6.1 Base Counterexample Graphs

This subsection gives definitions for the base counterexample graphs that will be used in the lower bound proofs.

Definition 5 (Line Graph. See Figure 1.) *The line graph P_n, consists of n nodes: $v_1, \ldots, v_n$. The only edges are in the path $(v_1, v_2, \ldots, v_n)$. This graph is parameterized by n.*

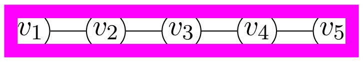

Figure 1: The unweighted line graph on 5 nodes.

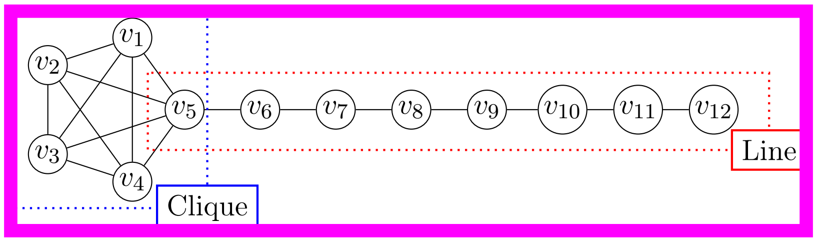

Figure 2: The unweighted $(n = 12, k = 5)$-flail graph.

Definition 6 (Flail Graph. See Figure 2) *The (n, k)-flail graph consists of n nodes: $v_1, \dots, v_n$. The first k nodes, $v_1, \dots, v_k$, form a clique. The only other edges are those in the path $v_k, v_{k+1}, \dots, v_n$. This graph is parameterized by n and k.*

Definition 7 (Double Star Graph. See Figure 3) *The k-double star graph consists of 2ℓ nodes: $v_1, \dots, v_{2k}$. There are edges from v_1 to all of $v_2, v_3, \dots, v_\ell$, forming a star. There are also edges from $v_{\ell+1}$ to all of $v_{\ell+2}, v_{\ell+3}, \dots, v_{2\ell}$, forming another star. Finally, there is an edge from v_1 to $v_{\ell+1}$, connecting the two stars. This graph is parameterized by ℓ, the size of each star.*

Definition 8 (Cycle Star Graph. See Figure 4) *The (k, ℓ)-cycle star graph consists of $k + k\ell$ nodes. Let $U = \{v_1, \dots, v_k\}$ and $U^i = \{v_{i,1}, \dots, v_{i,\ell}\}$ for $i \in [1, \dots, k]$. The vertex set $V = U \cup \bigcup_{i=1}^{k} U^i$. There are edges from v_i to v_{i+1} and v_k to v_1, forming a cycle. There are also edges from v_i to all of U^i, for each $i \in [1, \dots, k]$, forming a star. This graph is parameterized by k, the nodes on the cycle, and ℓ, the leaves in each of the k stars.*

Definition 9 (Clique Star Graph. See Figure 5) *The (k, ℓ)-clique star graph consists of $k + k\ell$ nodes. Let $U = \{v_1, \dots, v_k\}$ and $U^i = \{v_{i,1}, \dots, v_{i,\ell}\}$ for $i \in [1, \dots, k]$. The vertex set $V = U \cup \bigcup_{i=1}^{k} U^i$. The nodes in U form a clique. There are also edges from v_i to all of U^i, for each $i \in [1, \dots, k]$, forming a star. This graph is parameterized by k, the nodes int he clique, and ℓ, the leaves in each of the k stars.*

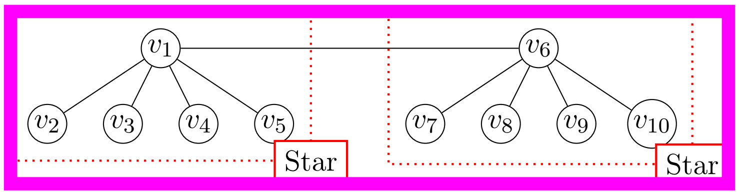

Figure 3: The unweighted $(\ell = 5)$-double star graph.

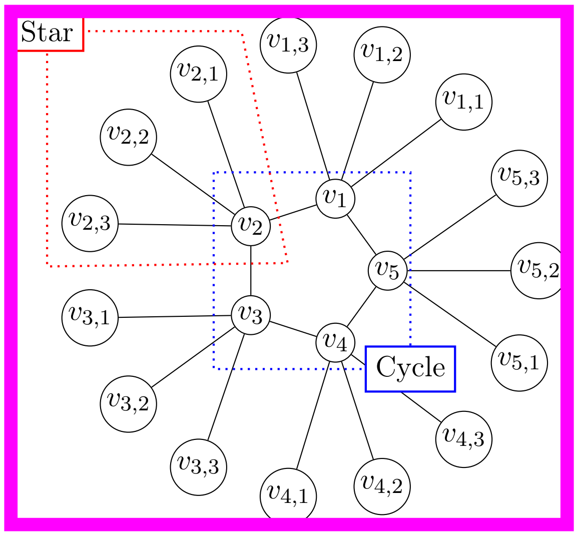

Figure 4: The unweighted $(k = 5, \ell = 3)$-cycle star graph.

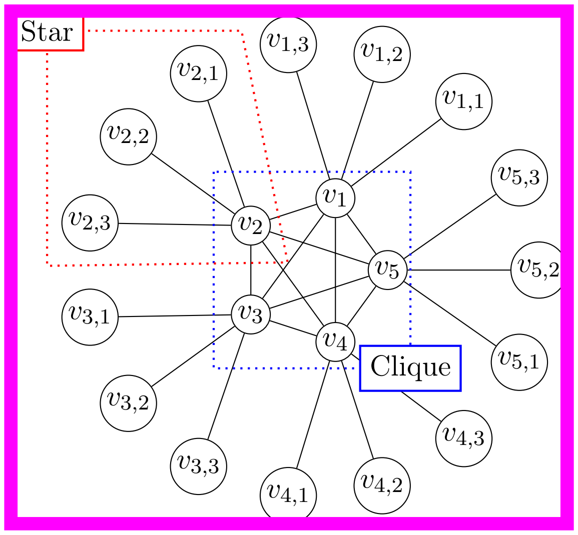

Figure 5: The unweighted $(k = 5, \ell = 3)$-clique star graph.

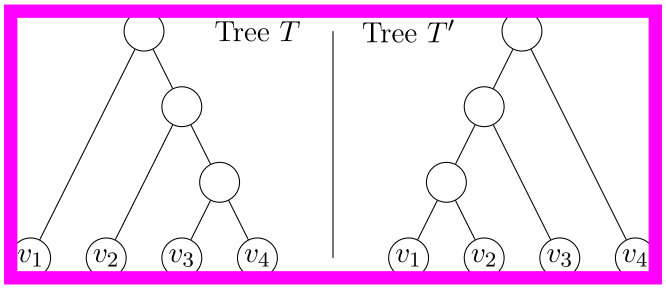

Figure 6: Hierarchical clusterings from the proof of Theorem 10, for $n = 4$. Single-linkage produces the left clustering, while the right clustering achieves a better objective value.

6.2 Single Linkage and the Reward Objective

This subsection gives a lower-bound proof showing that single linkage does not produce a good approximation to the reward objective introduced in this paper.

Theorem 10 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a maximizer of $\mathrm{reward}_G(\cdot)$ and T is the hierarchical clustering returned by single linkage, $\mathrm{reward}_G(T) \leq \frac{1}{\Omega(n^{1/3})} \cdot \mathrm{reward}_G(T^*)$.*

Proof For convenience, it is assumed that n is a perfect cube. If n is not, then the graph can be padded with unconnected vertices.

The base counterexample graph family is the flail graph; see Figure 2 for an example. For the actual counterexample graph, several weights for tie-breaking purposes will be increased. Let $\epsilon > 0$ be a constant that will be fixed later. Choose G_n to be the $(n, n^{2/3})$-flail graph. The weights w are as follows. All edges are unit weight, except edges between consecutive vertices: the edge (v_i, v_{i+1}) has weight $(1 + i\epsilon)$. In other words, the most valuable edge is (v_{n-1}, v_n), followed by (v_{n-2}, v_{n-1}), and so on all the way down to (v_1, v_2). The least valuable edges are the clique edges between nonconsecutive vertices.

Consider the clustering created by the single linkage algorithm on G_n. Recall that single linkage merges clusters in each step based on the most valuable edge between two clusters. By construction, it must take the cluster $\{v_n\}$ is merged with v_{n-1}. The resulting cluster is merged with v_{n-2}, and so on, until the cluster merges with v_1. This clustering is depicted in Figure 6. Let T be the tree corresponding to this hierarchical clustering. The reward given in this tree is bounded as follows. To compute the reward, we first split each edge into a unit weight component and a tie-breaking component. Then the contribution from clique edges and path edge are accounted for separately.

$$\begin{aligned}
\text{reward}_G(T) &= \sum_{(i,j)\in E} w_{ij}|\texttt{leaves-outside}(T[i \vee j])| \\
&= \underbrace{\sum_{(i,j)\in E} |\texttt{leaves-outside}(T[i \vee j])|}_{\text{Base Weight Component}} + \underbrace{\sum_{i=1}^{n-1} i\epsilon \cdot |\texttt{leaves-outside}(T[v_i \vee v_{i+1}])|}_{\text{Tie-Breaking Weight Component}} \\
&= \underbrace{\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} |\texttt{leaves-outside}(T[v_i \vee v_j])|}_{\text{Base Weight Component, Clique Edges}} + \underbrace{\sum_{i=k}^{n-1} |\texttt{leaves-outside}(T[v_i \vee v_{i+1}])|}_{\text{Base Weight Component, Path Edges}} \\
&+ \qquad \underbrace{\epsilon \sum_{i=1}^{n-1} i(i-1)}_{\text{Tie-Breaking Weight Component}} \\
&= \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} (i-1) + \sum_{i=k}^{n-1} (i-1) + \epsilon \cdot O(n^3) \\
&= O(k^3) + O(n^2) + \epsilon \cdot O(n^3)
\end{aligned}$$

Fix ϵ to be $1/n$. Since $k = n^{2/3}$, this entire expression is $O(n^2)$.

To finish the proof, it suffices to show that some hierarchical clustering obtains at least $\Omega(n^{7/3})$ reward on this graph. Consider the hierarchical clustering which does things in the opposite direction; T' takes the cluster $\{v_1\}$ and merges in v_2, then v_3, and so on, until it merges in v_n. This clustering is also depicted in Figure 6. How much reward does T' accumulate? A similar calculation as before bounds this as follows:

$$\begin{aligned}
\text{reward}_G(T') &= \sum_{(i,j)\in E} w_{ij}|\texttt{leaves-outside}(T'[v_i \vee v_j])| \\
&\geq \underbrace{\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} |\texttt{leaves-outside}(T'[v_i \vee v_j])|}_{\text{Base Weight Component, Clique Edges}} \\
&= \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} (n-j) \\
&\geq \binom{k}{2}(n-k)
\end{aligned}$$

Since $k = n^{2/3}$, this is at least $\Omega((n^{2/3})^2 \cdot n) = \Omega(n^{7/3})$, as desired. This establishes that some hierarchical clustering, and hence also the optimal hierarchical clustering, beats single linkage in terms of reward by a factor of at least $\Omega(n^{1/3})$. This completes the proof. ■

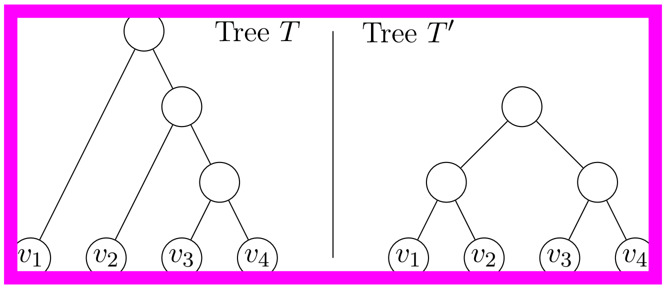

Figure 7: Hierarchical clusterings from the proof of Theorem 11, for $n = 4$. Single-linkage produces the left clustering, while the right clustering achieves a better objective value.

6.3 Single Linkage and the Cost Objective

In this subsection, it is shown that the algorithm single linkage does not produce a good approximation to Dasgupta's cost objective.

Theorem 11 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a minimizer of $cost_G(\cdot)$ and T is the hierarchical clustering returned by single linkage, $cost_G(T) \geq \Omega(\frac{n}{\log n}) \cdot cost_G(T^*)$.*

Proof The proof uses the line graph as the base counterexample graph family; see Figure 1 for an example. For the actual counterexample graph, several weights in the line graph are increased for tie-breaking purposes. Let $\epsilon > 0$ be a constant that will be set later. Choose G_n to be P_n. The weights of G_n are as follows. The edge (v_i, v_{i+1}) has weight $(1 + i\epsilon)$. In other words, the most valuable edge is (v_{n-1}, v_n), followed by (v_{n-2}, v_{n-1}), and so on all the way down to (v_1, v_2).

Consider what the single linkage algorithm does on G_n. Recall that single linkage only considers the most valuable edge between two clusters. By construction, it must take the cluster $\{v_n\}$ and merge in v_{n-1}, then v_{n-2}, and so on, until it merges in v_1. This clustering is depicted in Figure 7. Given that this is how the algorithm behaves, the question is how much cost does this hierarchical clustering T accumulate. This is bounded as follows.

$$\begin{aligned}
\mathrm{cost}_G(T) &= \sum_{i=1}^{n-1} (1 + i\epsilon)|\texttt{leaves}(T[i \vee i+1])| \\
&= \sum_{i=1}^{n-1} (1 + i\epsilon)(n - i + 1) \\
&= \underbrace{\sum_{i=1}^{n-1} (n - i + 1)}_{\text{Base Weight Component}} \quad + \quad \underbrace{\epsilon \sum_{i=1}^{n-1} i(n - i + 1)}_{\text{Tie-Breaking Weight Component}} \\
&\geq \Omega(n^2)
\end{aligned}$$

To finish the proof, it suffices to show that some hierarchical clustering obtains at most $O(n \log n)$ cost on this graph. This was essentially shown in Dasgupta's original paper, but this is presented here as well for the sake of completeness Dasgupta (2016).

Consider the following hierarchical clustering T'. The construction begins with each node in its own cluster: $C_1 = \{v_1\}, C_2 = \{v_2\}, \ldots, C_n = \{v_n\}$. Then perform a merge step by pairing up each odd numbered cluster with the next largest numbered cluster. This yields new clusters $C_1 = \{v_1, v_2\}, C_2 = \{v_3, v_4\}, \ldots$. The edges between the resulting clusters from a line graph. The process is repeated until there is a single cluster left. This clustering is also depicted in Figure 7. The total cost of this clustering T' is bounded as follows.

$$\begin{aligned}
\mathrm{cost}_G(T') &= \sum_{i=1}^{n-1} (1 + i\epsilon)|\mathtt{leaves}(T'[v_i \vee v_{i+1}])| \\
&= \underbrace{\sum_{i=1}^{n-1} |\mathtt{leaves}(T'[v_i \vee v_{i+1}])|}_{\text{Base Weight Component}} + \underbrace{\epsilon \sum_{i=1}^{n-1} i|\mathtt{leaves}(T'[v_i \vee v_{i+1}])|}_{\text{Tie-Breaking Weight Component}} \\
&\leq \sum_{i=1}^{n-1} |\mathtt{leaves}(T'[v_i \vee v_{i+1}])| + \epsilon \cdot O(n^3)
\end{aligned}$$

Choose $\epsilon = 1/n^2$ so that the second term is $O(n)$. To bound the first term, notice that every time a merge step is performed, at most n cost is incurred; there is a single edge between every pair of merged clusters and all the clusters being merged are disjoint. There are at most $O(\log n)$ merge steps performed, since each step reduces the number of clusters by a factor of 2. Hence $\mathrm{cost}_G(T') \leq O(n \log n)$, as desired. Together this shows that some hierarchical clustering, and hence also the optimal hierarchical clustering, beats single linkage in terms of cost by a factor of at least $\Omega(n/\log n)$. This completes the proof. ■

6.4 Average Linkage and the Reward Objective

In this subsection, it is shown that the average linkage algorithm is at best a constant-approximation to the reward objective.

Theorem 12 *Let $\delta > 0$ be any constant. There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a maximizer of reward$_G(\cdot)$ and T is the hierarchical clustering returned by average linkage, reward$_G(T) \leq (\frac{1}{2} + \delta) \cdot$ reward$_G(T^*)$, as long as n is sufficiently large.*

Proof Let k be a constant integer (that depends only on the constant δ) that we will select later. For convenience, assume that the number of nodes n is a multiple of k. If n is not, then the graph can be padded with unconnected vertices.

The base counterexample graph family is the clique star graph; see Figure 5 for an example. For the actual counterexample graph, several weights will be increased for tie-breaking purposes. Let $\epsilon > 0$ be a constant that will be fixed later. Choose G_n to be the

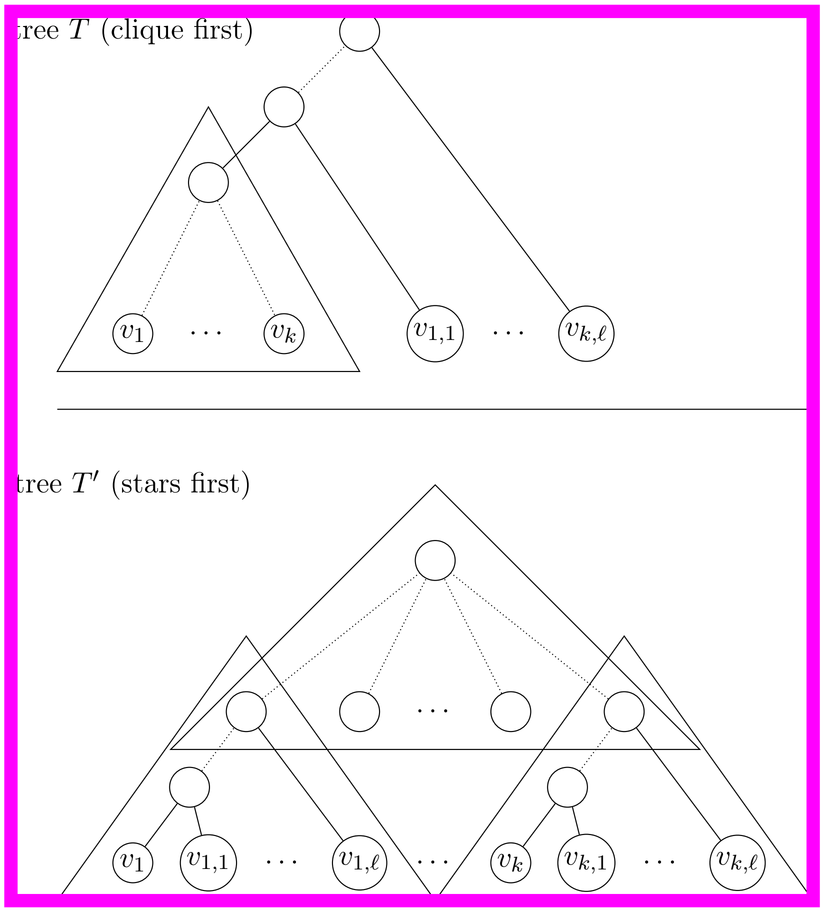

Figure 8: Hierarchical clusterings from the proofs of Theorem 12 and Theorem 13. Average-linkage produces the top clustering, while the bottom clustering achieves a better objective value. Solid edges denote direct children, while dotted edges denote indirect children.

$(k, (n/k) - 1)$-clique star graph. The weights w are as follows. All edges are unit weight, except the clique edges which have weight $(1 + \epsilon)$ instead.

Consider what average linkage does on G_n. Average linkage begins by merging together the clique of weight $(1 + \epsilon)$. After this, the remaining graph is a large star, so average linkage finishes by merging in one node at a time into the center. This clustering is depicted in Figure 8. How much reward does this hierarchical clustering T accumulate?

$$
\begin{aligned}
\text{reward}_G(T) &= \left[\underbrace{\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} |\texttt{leaves-outside}(T[v_i \vee v_j])| + \sum_{i=1}^{k} \sum_{j=1}^{\ell} |\texttt{leaves-outside}(T[v_i \vee v_{i,j}])|}_{\text{Base Weight Component}} \right] + \underbrace{\epsilon \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} |\texttt{leaves-outside}(T[v_i \vee v_j])|}_{\text{Tie-Breaking Weight Component}} \\
&\leq \underbrace{\sum_{i=1}^{k-1} \sum_{j=i+1}^{k} (n-j)}_{\text{Base Weight Component, Clique Edges}} + \underbrace{\sum_{i=k+1}^{n} (n-i)}_{\text{Base Weight Component, Star Edges}} \\
&\quad + \underbrace{\epsilon \cdot \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} (n-j)}_{\text{Tie-Breaking Weight Component}} \\
&\leq O(k^2 n) + \left(\frac{1}{2} n^2\right) + \epsilon \cdot O(k^2 n)
\end{aligned}
$$

We now choose $\epsilon = 1$ so that the RHS expression is $\frac{1}{2}n^2 + O(k^2 n)$. To finish the proof, it will suffice to show that some hierarchical clustering obtains nearly n^2 reward on this graph.

Consider the hierarchical clustering which first handles the small stars (by merging in one node at a time into the center), and then merges together the remaining clique. This clustering is also depicted in Figure 8. How much reward does T' accumulate?

$$\begin{aligned}
\text{reward}_G(T') &= \left[\underbrace{\begin{aligned}&\sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\texttt{leaves-outside}(T'[v_i \vee v_j])| \\ &+ \sum_{i=1}^{k}\sum_{j=1}^{\ell} |\texttt{leaves-outside}(T'[v_i \vee v_{i,j}])|\end{aligned}}_{\text{Base Weight Component}}\right] + \underbrace{\epsilon \sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\texttt{leaves-outside}(T'[v_i \vee v_j])|}_{\text{Tie-Breaking Weight Component}} \\
&\geq \underbrace{k\sum_{j=1}^{\ell}(n-j-1)}_{\text{Base Weight Component, Star Edges}} \\
&\geq k\ell(n-1-\frac{n}{2k}) \\
&= (n-k)(n-1-\frac{n}{2k}) \\
&= n^2 - kn - n + k - \frac{n^2}{2k} + n/2 \\
&\geq (1-\frac{1}{2k})n^2 - O(kn)
\end{aligned}$$

The remainder of the proof is relatively straightforward. We have an upper bound on $\text{reward}_G(T)$ and a lower bound on $\text{reward}_G(T')$. We have yet to pick the constant k and we have yet to define what it means for n to be sufficiently large. We want to use our remaining choices to guarantee a $(\frac{1}{2}+\delta)$ gap between $\text{reward}_G(T)$ and $\text{reward}_G(T')$.

Our first action is to let n be sufficiently large so that $O(k^2 n)$ and $O(kn)$ are both at most $\frac{1}{4}\delta n^2$. Our bounds simplify to the following:

$$\begin{aligned}
\text{reward}_G(T) &\leq \left(\frac{1}{2}+\frac{1}{4}\delta\right)n^2 \\
\text{reward}_G(T') &\geq \left(1-\frac{1}{2k}-\frac{1}{4}\delta\right)n^2
\end{aligned}$$

We know the final gap we want, so our choice of k is now forced. We would like:

$$\begin{aligned}
\left(\frac{1}{2}+\frac{1}{4}\delta\right)n^2 &\leq \left(\frac{1}{2}+\delta\right)\left(1-\frac{1}{2k}-\frac{1}{4}\delta\right)n^2 \\
\frac{\frac{1}{2}+\frac{1}{4}\delta}{\frac{1}{2}+\delta} &\leq \left(1-\frac{1}{2k}-\frac{1}{4}\delta\right) \\
\frac{1}{2k} &\leq \frac{\frac{3}{4}\delta}{\frac{1}{2}+\delta} - \frac{1}{4}\delta
\end{aligned}$$

Note that for $\delta \geq \frac{1}{2}$, the theorem statement is trivially true. However, if $\delta < \frac{1}{2}$, then $\frac{\frac{3}{4}\delta}{\frac{1}{2}+\delta} > \frac{3}{4}\delta$ and so the RHS is positive. Hence it is possible to pick an integer k large enough to create the final gap we want. In particular, choosing k to be $\lceil \delta^{-1} \rceil$ suffices. Thus, some hierarchical clustering, and hence also the optimal clustering, beats average linkage in terms of cost by a factor of at least $(\frac{1}{2} + \delta)$. This completes the proof. ■

6.5 Average Linkage and the Cost Objective

This subsection shows that the average linkage algorithm does not produce a good approximation to Dasgupta's cost objective.

Theorem 13 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a minimizer of $\mathrm{cost}_G(\cdot)$ and T is the hierarchical clustering returned by average linkage, $\mathrm{cost}_G(T) \geq \Omega(n^{1/3}) \cdot \mathrm{cost}_G(T^*)$.*

Proof For convenience, assume that n is a perfect cube. If n is not, then the graph can be padded with unconnected vertices.

The base counterexample graph family is the clique star graph; see Figure 5 for an example. For the actual counterexample graph, several weights will be increased for tie-breaking purposes. Let $\epsilon > 0$ be a constant that will be fixed later. Choose G_n to be the $(n^{1/3}, n^{2/3} - 1)$-clique star graph. The weights w are as follows. All edges are unit weight, except the clique edges which have weight $(1 + \epsilon)$ instead.

Consider what average linkage does on G_n. Average linkage begins by merging together the clique of weight $(1+\epsilon)$. After this, the remaining graph is a large star, so average linkage finishes by merging in one node at a time into the center. This clustering is depicted in Figure 8. How much cost does this hierarchical clustering T accumulate?

$$\begin{aligned}
\mathrm{cost}_G(T) &= \underbrace{\sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\mathtt{leaves}(T[v_i \vee v_j])| + \sum_{i=1}^{k}\sum_{j=1}^{\ell} |\mathtt{leaves}(T[v_i \vee v_{i,j}])|}_{\text{Base Weight Component}} \\
&\quad + \underbrace{\epsilon \sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\mathtt{leaves}(T[v_i \vee v_j])|}_{\text{Tie-Breaking Weight Component}} \\
&\geq \underbrace{\sum_{i=1}^{k-1} i \cdot (i+1)}_{\text{Base Weight Component, Clique Edges}} + \underbrace{\sum_{i=k+1}^{n} i}_{\text{Base Weight Component, Star Edges}} \\
&\geq \Omega(k^3) + (n-k)\frac{n+k+1}{2}
\end{aligned}$$

Since $k = n^{1/3}$, this entire expression is $\Omega(n^2)$. To finish the proof, it suffices to show that some hierarchical clustering obtains at most $O(n^{5/3})$ cost on this graph.

Consider the hierarchical clustering which first handles the small stars (by merging in one node at a time into the center), and then merges together the remaining clique. This clustering is also depicted in Figure 8. How much cost does T' accumulate?

$$
\begin{aligned}
\mathrm{cost}_G(T') &= \underbrace{\sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\texttt{leaves}(T'[v_i \vee v_j])| + \sum_{i=1}^{k}\sum_{j=1}^{\ell} |\texttt{leaves}(T'[v_i \vee v_{i,j}])|}_{\text{Base Weight Component}} \\
&\quad + \underbrace{\epsilon \sum_{i=1}^{k-1}\sum_{j=i+1}^{k} |\texttt{leaves}(T'[v_i \vee v_j])|}_{\text{Tie-Breaking Weight Component}} \\
&\leq \underbrace{\sum_{i=1}^{k-1} i\cdot(i+1)\cdot(\ell+1)}_{\text{Base Weight Component, Clique Edges}} + \underbrace{k\sum_{j=1}^{\ell}(j+1)}_{\text{Base Weight Component, Star Edges}} + \epsilon\cdot O(k^3\ell) \\
&\leq O(k^3\ell) + O(k\ell^2) + \epsilon\cdot O(k^3\ell)
\end{aligned}
$$

Since $k = n^{1/3}$ and $\ell = n^{2/3} - 1$, and we can choose $\epsilon = 1$, this entire expression is $O(n^{5/3})$. Thus, some hierarchical clustering, and hence also the optimal clustering, beats average linkage in terms of cost by a factor of at least $\Omega(n^{1/3})$. This completes the proof.

■

6.6 Complete Linkage and the Reward Objective

In this subsection, it is shown that the complete linkage algorithm does not produce a good approximation to the reward objective.

Theorem 14 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a maximizer of $\mathrm{reward}_G(\cdot)$ and T is the hierarchical clustering returned by complete linkage, $\mathrm{reward}_G(T) \leq \frac{1}{\Omega(n)} \cdot \mathrm{reward}_G(T^*)$.*

Proof For convenience, assume that n is even. If n is not, then the graph can be padded with unconnected vertices.

The base counterexample graph family is the double star graph; see Figure 3 for an example. Here n denotes the number of nodes and $\ell = (n-2)/2$. For the actual counterexample graph, several weights will be increased for tie-breaking purposes. Let $\epsilon > 0$ be a constant that will be fixed later. Choose G_n to be the $n/2$-double star graph, although our tie-breaking additions will cause some additional edges to appear. The edge $(v_1, v_{\ell+1})$ has weight $(1+\epsilon)$, while all other edges originally in the base graph have unit weight. Let $S = [n] \setminus \{1, \ell+1\}$. For all pairs $(i,j) \in S \times S$ where $i < j$, add an edge of weight ϵ between v_i and v_j (that is, add a clique of weight ϵ on S).

Consider what complete linkage does on G_n. Complete linkage begins by merging the edge of weight $(1+\epsilon)$, creating clusters $\{v_1, v_{\ell+1}\}, \{v_2\}, \ldots, \{v_n\}$. At this point, the $\{v_1, v_{\ell+1}\}$ cluster has zero complete linkage with every other cluster, but the complete linkage between any other pair of clusters is ϵ. Hence complete linkage will merge all other clusters (in some

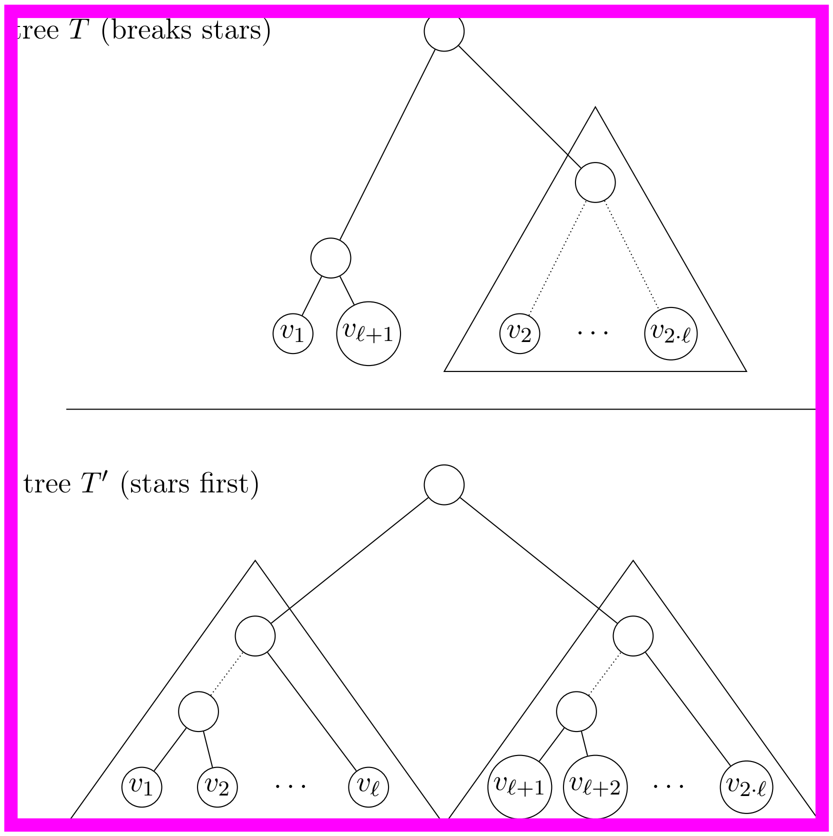

Figure 9: Hierarchical clusterings from the proofs of Theorem 14 and Theorem 16. Complete-linkage and Bisecting k-Means produce the top clustering, while the bottom clustering achieves a better objective value. Solid edges denote direct children, while dotted edges denote indirect children.

order) and only then merge in $\{v_1, v_{\ell+1}\}$. This clustering is depicted in Figure 9. How much reward does this hierarchical clustering T accumulate?

$$\begin{aligned}
\text{reward}_G(T) &= \left[\underbrace{\begin{array}{l} |\texttt{leaves-outside}(T[v_1 \vee v_{\ell+1}])| \\ + \displaystyle\sum_{i=2}^{\ell} |\texttt{leaves-outside}(T[v_1 \vee v_i])| \\ + \displaystyle\sum_{i=\ell+2}^{2\ell} |\texttt{leaves-outside}(T[v_{\ell+1} \vee v_i])| \end{array}}_{\text{Base Weight Component}}\right] + \left[\underbrace{\begin{array}{l} \epsilon|\texttt{leaves-outside}(T[v_1 \vee v_{\ell+1}])| \\ + \epsilon \displaystyle\sum_{\substack{(i,j)\in S\times S \\ i<j}} |\texttt{leaves-outside}(T[v_i \vee v_j])| \end{array}}_{\text{Tie-Breaking Weight Component}}\right] \\
&= [1 \cdot (n-2) + (\ell-1)\cdot 0 + (\ell-1)\cdot 0] + \left[\epsilon \cdot O(n^3)\right] \\
&= O(n) + \epsilon \cdot O(n^3)
\end{aligned}$$

Fix ϵ to be $1/n^2$, so this entire expression is $O(n)$. To finish the proof, it suffices to show that some hierarchical clustering obtains at least $\Omega(n^2)$ reward on this graph.

Consider the following hierarchical clustering T'. Cluster the stars separately; $\{v_1\}$ is merged with v_2, then v_3, and so on, until v_ℓ is merged. At the same time, $\{v_{k+1}\}$ is merged with $v_{\ell+2}$, then v_{k+3}, and so on, until $v_{2\ell}$ is merged. This clustering is also depicted in Figure 9. How much reward does T' accumulate?

$$\begin{aligned}
\text{reward}_G(T') &\geq \underbrace{\sum_{i=2}^{\ell} |\texttt{leaves-outside}(T'[v_1 \vee v_i])| + \sum_{i=\ell+2}^{2\ell} |\texttt{leaves-outside}(T'[v_{\ell+1} \vee v_i])|}_{\text{Base Weight Component, Star Edges}} \\
&\geq 2 \cdot \sum_{j=\ell}^{n-2} j \\
&\geq \Omega(n^2)
\end{aligned}$$

We have shown that some hierarchical clustering, and hence also the optimal hierarchical clustering, beats complete linkage in terms of reward by a factor of at least $\Omega(n)$. This completes the proof. ■

6.7 Complete Linkage and the Cost Objective

In this subsection, it is shown that the complete linkage algorithm does not produce a good approximation to Dasgupta's cost objective.

Theorem 15 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a minimizer of $cost_G(\cdot)$ and T is the hierarchical clustering returned by complete linkage, $cost_G(T) \geq \Omega(\frac{n}{\log n}) \cdot cost_G(T^*)$.*

Proof For convenience, it will be assumed that n is even. If n is odd, then the graph can be padded with unconnected vertices.

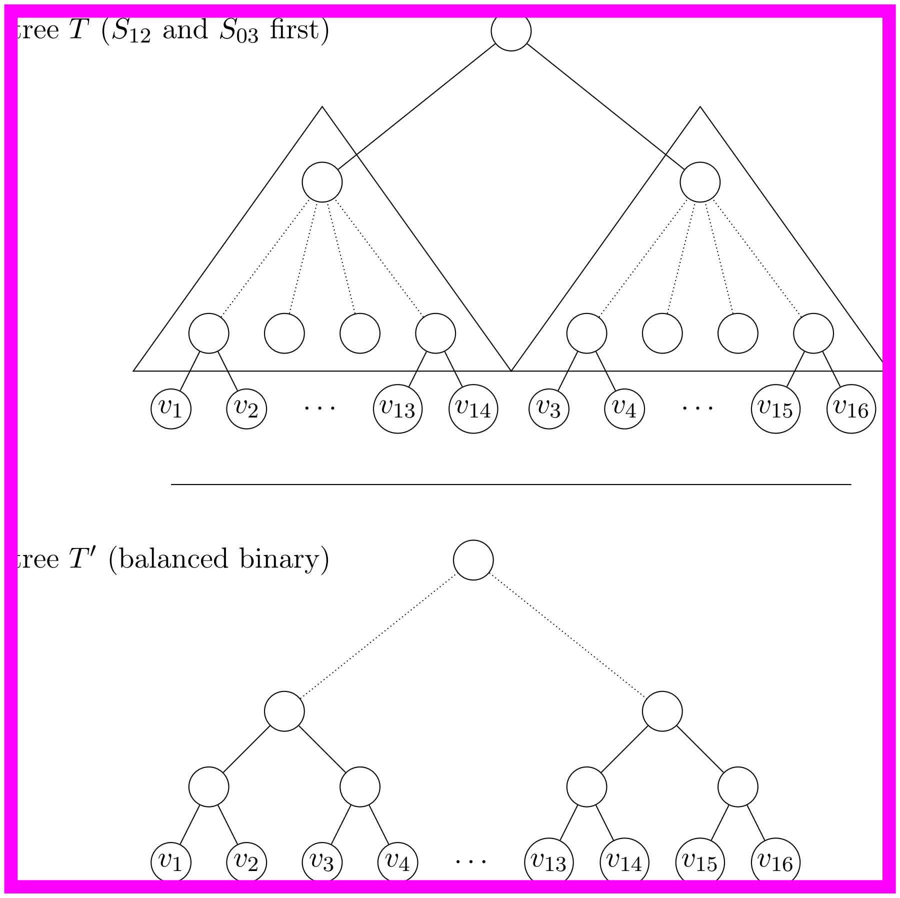

Figure 10: Hierarchical clusterings from the proof of Theorem 15, for $n = 16$. Complete-linkage produces the top clustering, while the bottom clustering achieves a better objective value. Solid edges denote direct children, while dotted edges denote indirect children.

Like with single linkage, the base counterexample graph family is the line graph; see Figure 1 for an example. For the actual counterexample graph, several weights are increased for tie-breaking purposes. Let $\epsilon > 0$ be a constant that will be fixed later. Choose G_n to be P_n, although the tie-breaking additions will cause some additional edges to appear. Let $S_{12} = \{i \in [n] \mid i \equiv 1 \pmod 4 \text{ or } i \equiv 2 \pmod 4\}$ and $S_{03} = \{i \in [n] \mid i \equiv 0 \pmod 4 \text{ or } i \equiv 3 \pmod 4\}$. For all pairs $(i, j) \in S_{12} \times S_{12}$ where $i < j$, add ϵ to the weight of edge (v_i, v_j). If such an edge did not exist before, pretend it existed as a zero-weight edge, which now has ϵ-weight (that is, we add an clique of weight ϵ on S_{12}). This process is repeated for all pairs $(i, j) \in S_{03} \times S_{03}$ where $i < j$ (that is, we also add an clique of weight ϵ on S_{03}).

Consider what the complete linkage algorithm does on G_n. Complete linkage begins by merging all the edges of weight $(1 + \epsilon)$, resulting with the clusters $\{v_1, v_2\}, \{v_3, v_4\}, \ldots, \{v_{n-1}, v_n\}$. At this point, there are two types of clusters. A cluster either (i) only includes v_i where i is congruent to one or two modulo four, or (ii) only includes v_i where i is congruent to zero or three modulo four. Furthermore, the complete linkage between two clusters of the former type is ϵ, the complete linkage between two clusters of the latter type is ϵ, and the complete linkage between two clusters of different types is zero. Hence, complete linkage will merge all clusters of the former type (in some order) and all clusters of the latter type (in some order). Finally, it will merge these two large clusters together. This clustering is depicted in Figure 10. The cost this hierarchical clustering T accumulates is as follows.

$$\begin{aligned}\text{cost}_G(T) &= \underbrace{\sum_{i=1}^{n-1} |\texttt{leaves}(T[v_i \vee v_{i+1}])|}_{\text{Base Weight Component}} \\ &\quad \underbrace{+\,\epsilon \sum_{\substack{(i,j)\in S_{12}\times S_{12}\\ i<j}} |\texttt{leaves}(T[v_i \vee v_j])| + \epsilon \sum_{\substack{(i,j)\in S_{03}\times S_{03}\\ i<j}} |\texttt{leaves}(T[v_i \vee v_j])|}_{\text{Tie-Breaking Weight Component}} \\ &\geq (n/2\cdot 2 + (n/2-1)\cdot n) \\ &\geq \Omega(n^2)\end{aligned}$$

Note that $n/2$ of the base edges are involved immediately in a merge into a cluster of size two, while the remainder are involved in the final merge into a cluster of size n. To finish the proof, it suffices to show that some hierarchical clustering obtains at most $O(n\log n)$ cost on this graph. The proof of this is identical to that in Theorem 11, except that the tie-breaking term is slightly different. This clustering is also depicted in Figure 10. Choosing $\epsilon = 1/n^2$ still suffices to make the $O(n\log n)$ term dominate, since for the current graph added at most $O(n^2)$ edges, each of which involves at most $O(n)$ leaves.

Together this shows that some hierarchical clustering, and hence also the optimal clustering, beats complete linkage in terms of cost by a factor of at least $\Omega(n\log n)$. This completes the proof.

■

6.8 Bisecting k-Means and the Reward Objective

In this subsection, we consider the divisive algorithm which uses the k-means objective (with $k = 2$) when choosing how to split clusters. Normally, the k-means objective concerns the distances between points and their cluster center: $\min \sum_{i=1}^{k}\sum_{x\in S_i} ||x-\mu_i||^2$. However, it is known that this can be rewritten as a sum over intra-cluster distances: $\min \sum_{i=1}^{k} \frac{1}{|S_i|}\sum_{x,y\in S_i} ||x-y||^2$ Awasthi et al. (2015). In other words, when splitting a cluster into two sets A and B, the algorithm minimizes $\frac{1}{|A|}\sum_{a,a'\in A} ||a-a'||^2 + \frac{1}{B}\sum_{b,b'\in B} ||b-b'||^2$. At first glance, this appears to almost capture split-rew$_G(A,B)$; the key difference is that the summation has been scaled down by a factor of $|A||B|$. Of course, it also involves minimization over squared distances instead of maximization over similarity weights. We show that the divisive algorithm which splits clusters by the natural k-means similarity

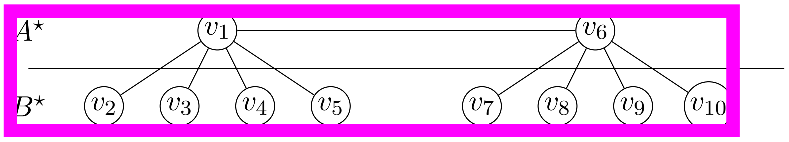

Figure 11: The initial split $(A^\star, B^\star)$ of bisecting 2-means from the proof of Theorem 16, for $\ell = 5$.

objective, namely $\max \frac{1}{|A|} \sum_{a,a' \in A} w_{aa'} + \frac{1}{|B|} \sum_{b,b' \in B} w_{bb'}$, is not a good approximation to the optimal hierarchical clustering.

Theorem 16 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a minimizer of $cost_G(\cdot)$ and T is the hierarchical clustering returned by bisecting k-Means, $reward_G(T) \leq \frac{1}{\Omega(\sqrt{n})} \cdot reward_G(T^*)$.*

Proof For convenience, we will assume that n is even and a perfect square. If n is not, then we can always pad the graph with unconnected vertices.

The base counterexample graph family is the double star graph; see Figure 3 for an example. In the actual counterexample graph, several weights are increased. Choose G_n to be the $(\ell = n/2)$-cycle star graph. The weights w are as follows. All edges are unit weight, except for the $(v_1, v_{\ell+1})$ edge (which connects the two stars), which has $\sqrt{n}$ weight instead.

Consider what bisecting k-means does on G_n. We want to show that the first step of bisecting k-means results in the clusters $A^* = \{v_1, v_{\ell+1}\}$, $B^* = \{v_2, v_3, \ldots, v_\ell, v_{\ell+2}, v_{\ell+3}, \ldots, v_{2\ell}\}$. This split is depicted in Figure 11.

Recall that the k-means (similarity) objective is $\max \frac{w(A)}{|A|} + \frac{w(B)}{|B|}$. Note that (A^*, B^*) achieves a (k-means) objective value of $\frac{\sqrt{n}}{2}$. Can any other choice of clusters (A, B) match this objective value? We begin by observing that G_n is a tree, so the graphs induced by A and B must be forests. A forest always has more vertices than edges, so the contribution of a cluster to the objective value is at most its average edge weight. Hence it cannot be the case that $(v_1, v_{\ell+1})$ does not lie completely inside A or B; this would result in objective value at most two.

Without loss of generality, then, we assume that $v_1, v_{\ell+1} \in A$. At this point, B cannot contain edges since all edges have v_1 or $v_{\ell+1}$ as one of their endpoints. Furthermore, adding any other vertices to A only decreases its contribution, since any other vertex contributes one to $w(A)$ and one to $|A|$, which only serves to drag down the average $\frac{w(A)}{|A|}$. Hence we have shown that k-means begins by choosing clusters (A^*, B^*). The upshot of this is that the initial split into (A^*, B^*) cuts all $n-2$ star edges, which greatly reduces the reward of T:

$$\begin{aligned}\text{reward}_G(T) &= \left[\underbrace{\begin{array}{l}\displaystyle\sum_{i=2}^{\ell} |\texttt{leaves-outside}(T[v_1 \vee v_i])| \\ \displaystyle + \sum_{i=\ell+2}^{2\ell} |\texttt{leaves-outside}(T[v_{\ell+1} \vee v_i])|\end{array}}_{\text{Star Edges}}\right] + \underbrace{\sqrt{n}|\texttt{leaves-outside}(T[v_1 \vee v_{\ell+1}])|}_{\text{Edge Between Stars}} \\ &= 0 + \sqrt{n}(n-2)\end{aligned}$$

To finish the proof, it suffices to show that some hierarchical clustering obtains at least $\Omega(n^2)$ reward on this graph.

Consider the following hierarchical clustering T'. Cluster the stars separately: $\{v_1\}$ is merged with v_2, then v_3, and so on, until we merge in v_ℓ. At the same time, $\{v_{\ell+1}\}$ is merged with $v_{\ell+2}$, then $v_{\ell+3}$, and so on, until we merge in $v_{2\ell}$. Finally, merge these two clusters together. This clustering is also depicted in Figure 9. How much reward does T' accumulate?

$$\begin{aligned}\text{reward}_G(T') &\geq \underbrace{\sum_{i=2}^{\ell} |\texttt{leaves-outside}(T'[v_1 \vee v_i])| + \sum_{i=\ell+2}^{2\ell} |\texttt{leaves-outside}(T'[v_{\ell+1} \vee v_i])|}_{\text{Star Edges}} \\ &\geq 2 \cdot \sum_{i=2}^{\ell} (n-i) \\ &\geq \Omega(\ell n)\end{aligned}$$

Since $\ell = n/2$, this is $\Omega(n^2)$, as desired. This shows that some hierarchical clustering, and hence also the optimal hierarchical clustering, beats bisecting k-means in terms of reward by a factor of at least $\frac{1}{\Omega(\sqrt{n})}$. This completes the proof. ■

6.9 Bisecting k-Means and the Cost Objective

This subsection gives a lower bound showing that bisecting k-means does not produce a good approximation to Dasgupta's cost objective.

Theorem 17 *There exists a family of graphs G_n with nonnegative edge weights $w : E \to \mathbb{R}^{\geq 0}$ such that if the hierarchical clustering T^* is a minimizer of $\text{cost}_G(\cdot)$ and T is the hierarchical clustering returned by bisecting k-Means, $\text{cost}_G(T) \geq \Omega(\sqrt{n}) \cdot \text{cost}_G(T^*)$.*

Proof For convenience, we will assume that n is a perfect square. If n is not, then we can always pad the graph with unconnected vertices.

The base counterexample graph family is the cycle star graph; see Figure 4 for an example. In the actual counterexample graph, several weights are increased. Choose G_n to be the

$(n^{1/2}, n^{1/2}-1)$-cycle star graph. The weights w are as follows. All edges are unit weight, except the clique edges which have weight $\sqrt{n}$ instead.

Consider what bisecting k-means does on G_n. The first step of bisecting k-means is the most important, and the proof requires some analysis to prove this step is an irrevocable mistake.

Recall that the k-means (similarity) objective is $\max \frac{w(A)}{|A|} + \frac{w(B)}{|B|}$. One choice of initial clusters is $A = U$ and $B = \bigcup_{i=1}^{k} U^i$. This choice has a ($k$-means similarity) objective score of $\sqrt{n}$. The optimal choice of initial clusters (A^*, B^*) must hence score at least $\sqrt{n}$. This implies that either $\frac{w(A^*)}{|A^*|}$ or $\frac{w(B^*)}{|B^*|}$ must be at least $\frac{\sqrt{n}}{2}$. Without loss of generality, assume it is $\frac{w(A^*)}{|A^*|}$. Since the total weight in the graph is at most $2n$, this implies that A^* is small: $|A^*| \le 4\sqrt{n}$. B^* must be correspondingly large: $|B^*| \ge n - 4\sqrt{n}$. This means that (assuming n is large enough) $\frac{w(B^*)}{|B^*|}$ is a constant!

We have come to the conclusion that $\frac{w(A^*)}{|A^*|}$ must be contributing at least $\sqrt{n} - O(1)$. Notice that if A^* contains $x < \sqrt{n}$ nodes from U, its contribution can be at most $\frac{\sqrt{n}(x-1)}{x}$ (nodes in $V \setminus U$ only make matters worse, since they contribute one to the numerator and denominator at best). This implies that A^* must contain at least a constant fraction of the nodes in U:

$$\begin{aligned}\frac{\sqrt{n}(x-1)}{x} &\ge \sqrt{n} - O(1)\\ O(1) &\ge \frac{\sqrt{n}}{x}\\ x &\ge \Omega(\sqrt{n})\end{aligned}$$

However, these nodes in $A^* \cap U$ are the centers of several stars. Together, these stars represent $\Omega(n)$ nodes, of which A^* cannot even cover a constant fraction, since we established that A^* is too small. These facts about the initial split are depicted in Figure 12. The upshot of this is that this initial split into (A^*, B^*) cuts $\Omega(n)$ star edges, which will contribute a large factor to the cost of T:

$$\begin{aligned}\text{cost}_G(T) &\ge \underbrace{\sum_{i=1}^{k}\sum_{j=1}^{\ell} |\texttt{leaves}(T[v_i \vee v_{i,j}])|}_{\text{Star Edges}}\\ &\ge \Omega(n)\cdot n\\ &= \Omega(n^2)\end{aligned}$$

To finish the proof, it suffices to show that some hierarchical clustering obtains at most $O(n^{1.5})$ cost on this graph.

Consider the following hierarchical clustering T'. Cluster the stars separately: $\{v_i\}$ is merged with $v_{i,1}$, then $v_{i,2}$, and so on, until we merge in $v_{i,\ell}$. This leaves us with a cycle of k clusters, which we handle as we have handled line graphs; pair up each odd cluster with the next largest cluster, repeatedly, until we are left with a single cluster. This clustering is

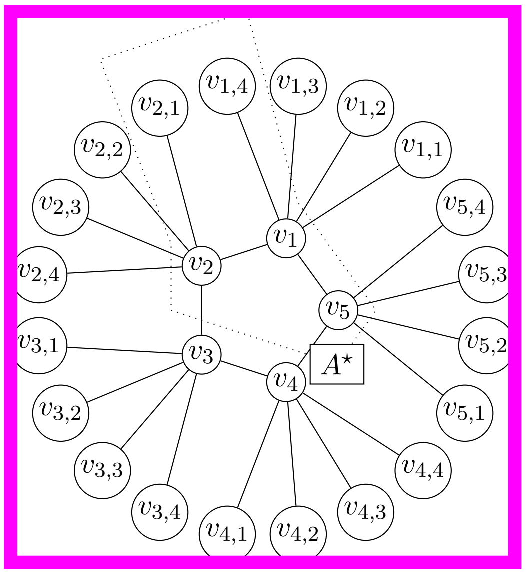

Figure 12: A depiction of the initial split $(A^\star, B^\star)$ of bisecting 2-means from the proof of Theorem 17, for $k = 5$, $\ell = 4$. Our analysis shows that $A^\star$ must contain a constant fraction of the central cycle nodes. Additionally, since it is of $O(\sqrt{n})$ size it cannot contain many star nodes.

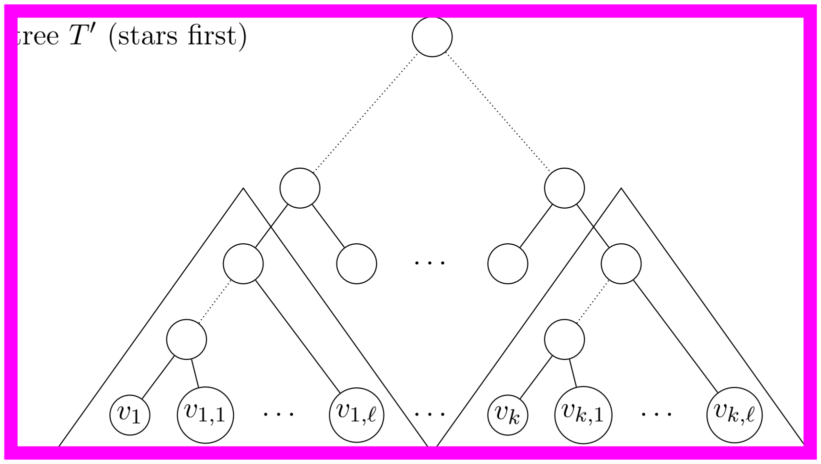

Figure 13: Hierarchical clustering from the proof of Theorem 17. The depicted clustering achieves a better objective value than bisecting 2-means (which is illustrated in Figure 12). Solid edges denote direct children, while dotted edges denote indirect children.

depicted in Figure 13.How much cost does T' accumulate?

$$\begin{aligned}\text{cost}_G(T') = &\underbrace{\sqrt{n}|\texttt{leaves}(T'[v_k \vee v_1])| + \sqrt{n}\sum_{i=1}^{k-1}|\texttt{leaves}(T'[v_i \vee v_{i+1}])|}_{\text{Cycle Edges}} \\ &+ \underbrace{\sum_{i=1}^{k}\sum_{j=1}^{\ell}|\texttt{leaves}(T'[v_i \vee v_{i,j}])|}_{\text{Star Edges}} \\ \leq &(n \log n) + (k\ell^2)\end{aligned}$$

Since $k = \sqrt{n}$ and $\ell = \sqrt{n} - 1$, this is at most $O(n^{1.5})$, as desired. This shows that some hierarchical clustering, and hence also the optimal hierarchical clustering, beats bisecting k-means in terms of reward by a factor of at least $\Omega(\sqrt{n})$. This completes the proof. ■

7. Conclusion

One motive for developing an analytic framework is that it may help clarify and explain our observations from practice. In this case, we have shown that average linkage is a $\frac{1}{3}$-approximation to a particular objective function (Theorem 1), and the analysis that does so helps to explain what average linkage is optimizing. We have also shown that average-linkage can be no better than a $\frac{1}{2}$-approximation (Theorem 12)[1]. One open problem is to devise new algorithms and determine the best approximation ratio possible for the problem. The current state of the art is a (≈ 0.336)-approximation based on semi-definite programming (Charikar et al., 2019a). Can this be improved further? Another open problem is to find a characterization of graphs that excludes some of the worst-case ones used to prove negative results. Is there a formal way to restrict inputs that allows for better objective guarantees?

We mention that similar results to ours for average-linkage have been shown by Cohen-Addad et al. (2017). In this work, it is shown that average-linkage is a $\frac{1}{2}$-approximation for a related objective function when there are dissimilarity scores between the points.

Another open direction is the possibility of other objective functions. What are single linkage, complete linkage, and bisecting k-means optimizing for? One quirk shared by both Dasgupta's cost objective and our reward objective is that the optimal tree is always binary. This is not appropriate for all applications; for example, in the classical application of biological taxonomy, groups typically contain much more than two subgroups. Can we devise an objective function which incentivizes non-binary trees?

8. Acknowledgments

Benjamin Moseley was supported in part by a Google Research Award, a Yahoo Research Award and NSF Grants CCF-1617724, CCF-1733873, CCF-1725661, CCF-1824303, CCF-

1. A follow-up work to ours shows that average-linkage cannot achieve a $\frac{1}{3} + \epsilon$ approximation for any $\epsilon > 0$ (Charikar et al., 2019a).

1845146 and CMMI-1938909. This work was partially done while the author was working at Washington University in St. Louis.

Joshua R. Wang was supported in part by NSF Grant CCF-1524062. This work was partially done while the author was at Stanford University.

References

Amir Abboud, Vincent Cohen-Addad, and Hussein Houdrougé. Subquadratic high-dimensional hierarchical clustering. In *Advances in Neural Information Processing Systems*, pages 11576–11586, 2019.

Margareta Ackerman and Shai Ben-David. A characterization of linkage-based hierarchical clustering. *Journal of Machine Learning Research*, 17:232:1–232:17, 2016.

Margareta Ackerman, Shai Ben-David, Simina Brânzei, and David Loker. Weighted clustering. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*, 2012.

Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *J. ACM*, 56(2):5:1–5:37, 2009.

Pranjal Awasthi, Afonso S Bandeira, Moses Charikar, Ravishankar Krishnaswamy, Soledad Villar, and Rachel Ward. Relax, no need to round: Integrality of clustering formulations. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 191–200. ACM, 2015.

Shai Ben-David and Margareta Ackerman. Measures of clustering quality: A working set of axioms for clustering. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 121–128, 2008. URL `http://papers.nips.cc/paper/3491-measures-of-clustering-quality-a-working-set-of-axioms-for-clustering`.

Moses Charikar and Vaggos Chatziafratis. Approximate hierarchical clustering via sparsest cut and spreading metrics. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 841–854, 2017.

Moses Charikar, Vaggos Chatziafratis, and Rad Niazadeh. Hierarchical clustering better than average-linkage. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2291–2304, 2019a.

Moses Charikar, Vaggos Chatziafratis, Rad Niazadeh, and Grigory Yaroslavtsev. Hierarchical clustering for euclidean data. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 2721–2730, 2019b.

Giovanni Chierchia and Benjamin Perret. Ultrametric fitting by gradient descent. In *Advances in neural information processing systems*, pages 3175–3186, 2019.

Vincent Cohen-Addad, Varun Kanade, Frederik Mallmann-Trenn, and Claire Mathieu. Hierarchical clustering: Objective functions and algorithms. *CoRR*, abs/1704.02147, 2017.

Sanjoy Dasgupta. A cost function for similarity-based hierarchical clustering. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 118–127, 2016.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Unsupervised Learning*, pages 485–585. Springer New York, New York, NY, 2009.

Katherine A. Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, pages 297–304, 2005.

Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8): 651 – 666, 2010. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2009.09.011. URL `http://www.sciencedirect.com/science/article/pii/S0167865509002323`.

Akshay Krishnamurthy, Sivaraman Balakrishnan, Min Xu, and Aarti Singh. Efficient active algorithms for hierarchical clustering. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

Silvio Lattanzi, Thomas Lavastida, Kefu Lu, and Benjamin Moseley. A framework for parallelizing hierarchical clustering methods. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 73–89. Springer, 2019.

Xiaofei Ma and Satya Dhavala. Hierarchical clustering with prior knowledge. *arXiv preprint arXiv:1806.03432*, 2018.

Aditya Krishna Menon, Anand Rajagopalan, Baris Sumengen, Gui Citovsky, Qin Cao, and Sanjiv Kumar. Online hierarchical clustering approximations. *arXiv preprint arXiv:1909.09667*, 2019.

Nicholas Monath, Manzil Zaheer, Daniel Silva, Andrew McCallum, and Amr Ahmed. Gradient-based hierarchical clustering using continuous representations of trees in hyperbolic space. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 714–722, 2019.

Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

Aurko Roy and Sebastian Pokutta. Hierarchical clustering via spreading metrics. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2316–2324, 2016.

Dingkang Wang and Yusu Wang. An improved cost function for hierarchical cluster trees. *arXiv preprint arXiv:1812.02715*, 2018.

Yuyan Wang and Ben Moseley. An objective for hierarchical clustering in euclidean space and its connection tobisecting k-means. In *Proceedings of the 34th Conference on Artificial Intelligence (AAAI 2020)*, 2020.

Reza Zadeh and Shai Ben-David. A uniqueness theorem for clustering. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 639–646, 2009.