New Bounds on the Cohesion of Complete-link and Other Linkage Methods for Agglomerative Clustering

Sanjoy Dasgupta [1] **Eduardo Laber** [2]

Abstract

Linkage methods are among the most popular algorithms for hierarchical clustering. Despite their relevance the current knowledge regarding the quality of the clustering produced by these methods is limited. Here, we improve the currently available bounds on the maximum diameter of the clustering obtained by `complete-linkage` for metric spaces. One of our new bounds, in contrast to the existing ones, allows us to separate `complete-linkage` from `single-linkage` in terms of approximation for the diameter, which corroborates the common perception that the former is more suitable than the latter when the goal is producing compact clusters. We also show that our techniques can be employed to derive upper bounds on the cohesion of a class of linkage methods that includes the quite popular `average-linkage`.

1. Introduction

Clustering is the problem of partitioning a set of items so that similar items are grouped together and dissimilar items are separated. It is a fundamental tool in machine learning that is commonly used for exploratory analysis and for reducing the computational resources required to handle large datasets. For comprehensive descriptions of different clustering methods and their applications, we refer to (Jain et al., 1999; Hennig et al., 2015).

One important type of clustering is hierarchical clustering. Given a set of n points, a hierarchical clustering is a sequence of clusterings $(\mathcal{C}^0, \mathcal{C}^1, \ldots, \mathcal{C}^{n-1})$, where $\mathcal{C}^0$ is a clustering with n unitary clusters, each of them corresponding to one of the n points, and $\mathcal{C}^i$, for $i \geq 1$, is obtained from $\mathcal{C}^{i-1}$ by replacing two clusters of $\mathcal{C}^{i-1}$ with their union. Hierarchical clustering algorithms are implemented in widely used machine learning libraries such as `scipy` and they have applications in many contexts such as in the study of evolution through phylogenetic trees (Eisen et al., 1998).

There is a significant literature on hierarchical clustering; for good surveys we refer to (Murtagh, 1983; Murtagh & Contreras, 2012). With regards to more theoretical work, one important line of research consists of designing algorithms for hierarchical clustering with provable guarantees for natural optimization criteria such as cluster diameter and the sum of quadratic errors (Dasgupta & Long, 2005; Charikar et al., 2004; Lin et al., 2010; Arutyunova & Röglin, 2022). Another relevant line aims to understand the theoretical properties (e.g. approximation guarantees) of algorithms widely used in practice, such as linkage methods (Dasgupta & Long, 2005; Ackermann et al., 2010; Großwendt & Röglin, 2015; Arutyunova et al., 2021; Großwendt et al., 2019).

Here, we contribute to this second line of research by giving new and improved analysis for the `complete-linkage` (Ackerman & Ben-David, 2016) and also for a class of linkage methods that includes `average-linkage` (Ackerman & Ben-David, 2016) and `minimax` (Bien & Tibshirani, 2011).

1.1. Our Results

Let $(\mathcal{X}, dist)$ be a metric space, where $\mathcal{X}$ is a set of n points. The diameter $\texttt{diam}(S)$ of a set of points S is given by $\texttt{diam}(S) = \max\{dist(x,y) | x, y \in S\}$. A *$k$-clustering* $\mathcal{C} = \{C_i | 1 \leq i \leq k\}$ is a partition of $\mathcal{X}$ into k groups. We define $\texttt{max-diam}(\mathcal{C}) := \max\{\texttt{diam}(C_i) | 1 \leq i \leq k\}$ and $\texttt{avg-diam}(\mathcal{C}) := \frac{1}{k}\sum_{i=1}^{k} \texttt{diam}(C_i)$. Moreover, let $\text{OPT}_{\texttt{DM}}(k)$ and $\text{OPT}_{\texttt{AV}}(k)$ be, respectively, the minimum possible `max-diam` and `avg-diam` of a k-clustering for $(\mathcal{X}, dist)$.

Arbitrary k. First, in Section 3, we prove that for all k the maximum diameter of the k-clustering produced by `complete-linkage` is at most $k^{1.59}\text{OPT}_{\texttt{AV}}(k)$. Since $\text{OPT}_{\texttt{AV}}(k) \leq \text{OPT}_{\texttt{DM}}(k)$, our result is an improvement over $O(k^{1.59}\text{OPT}_{\texttt{DM}}(k))$, the best known upper bound on the maximum diameter of `complete-linkage` (Aru-

[1] University of California San Diego, USA [2] Departamento de Informática, PUC-RIO, Brazil. Correspondence to: Eduardo Laber <eduardo.laber1@gmail.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

tyunova et al., 2023). Indeed, our bound can improve the previous one by up to a factor of k since there are instances in which $\text{OPT}_{\texttt{AV}}(k)$ is $\Theta(\frac{\text{OPT}_{\texttt{DM}}(k)}{k})$.

It is noteworthy that by using $\text{OPT}_{\texttt{AV}}$ rather than $\text{OPT}_{\texttt{DM}}$, we can corroborate with the intuition that `complete-linkage` produces clusters with smaller diameters than those produced by `single-linkage` since, in addition to the $k^{1.59}\text{OPT}_{\texttt{AV}}(k)$ upper bound for the former, we show an instance in which the maximum diameter of the latter is $\Omega(k^2\text{OPT}_{\texttt{AV}}(k))$. When $\text{OPT}_{\texttt{DM}}$ is employed, unexpectedly, as pointed out in (Arutyunova et al., 2023), this separation is not possible since the maximum diameter of `complete-linkage` is $\Omega(k\text{OPT}_{\texttt{DM}}(k))$ while that of `single-linkage` is $\Theta(k\text{OPT}_{\texttt{DM}}(k))$.

To obtain the aforementioned upper bound, our main technique consists of carefully defining a partition of the clusters built by `complete-linkage` along its execution and then bounding the diameter of the families in the partition. This technique yields an arguably simpler analysis than that of (Arutyunova et al., 2021; 2023).

Next, in Section 4, by using our technique in a significantly more involved way, we show that the maximum diameter of the k-clustering produced by `complete-linkage` is at most $(2k-2)\text{OPT}_{\texttt{DM}}(k)$ for $k \leq 4$ and at most $k^{1.30}\text{OPT}_{\texttt{DM}}(k)$, for $k > 4$. Thus, we considerably narrow the gap between the current upper bound and $\Omega(k\text{OPT}_{\texttt{DM}}(k))$, the best known lower bound.

Finally, in Section 5, we show that our techniques can be employed to obtain upper bounds on cohesion criteria of the clustering built by methods that belong to a class of linkage methods that includes `average-linkage` and `minimax`. In particular, we show that the average pairwise distance of every cluster in the k-clustering produced by `average-linkage` is at most $k^{1,59}\text{OPT}_{\texttt{AV}}(k)$. To the best of our knowledge, our analysis of the `average-linkage` is the first one regarding to a cohesion criterion.

Low values of k and practical applications. For large k, the upper bounds of `complete-linkage`, though close to the lower bound, are high and, thus, are not informative in the context of practical applications. However, as argued below, we have a different scenario for the very relevant case in which k is small. The relevance of small k is that, in general, people have difficulties in analyzing a partition containing many groups (large k).

(Charikar et al., 2004; Dasgupta & Long, 2005) propose algorithms that obtain a hierarchical clustering that guarantees an 8-approximation to the diameter for every k. The analysis from (Arutyunova et al., 2023) give, respectively, the following upper bounds on the approximation factor of `single-linkage` and `complete-linkage` regarding the diameter: 4 and 3 for $k = 2$; 6 and 5.71, for $k = 3$ and 8 and 9 for $k = 4$. Our analysis gives an approximation factor of $2k-2$ for $k \leq 4$, which improves these bounds. For an assessment of the quality of these bounds, one should take into account that, unless $P = NP$, the problem of finding the k-clustering that minimizes the maximum diameter, for $k \geq 3$, does not admit an approximation better than 2 in polytime (Megiddo, 1990).

For $k \geq 5$, the $k^{1.30}\text{OPT}_{\texttt{DM}}(k)$ upper bound does not improve the factor of 8 achieved by the algorithms proposed in (Charikar et al., 2004; Dasgupta & Long, 2005). However, our $k^{1.59}\text{OPT}_{\texttt{AV}}(k)$ upper bound improves it for instances in which $\text{OPT}_{\texttt{AV}}(k) \leq \frac{8}{k^{1.59}}\text{OPT}_{\texttt{DM}}(k)$. Since $\frac{\text{OPT}_{\texttt{DM}}(k)}{k} \leq \text{OPT}_{\texttt{AV}}(k) \leq \text{OPT}_{\texttt{DM}}(k)$, we can have improvements for $k \leq 34$.

An interesting aspect of our results is that they point in the opposite direction of the common intuition that bottom-up methods for hierarchical clustering do not work well for small k and, hence, are less preferable than top-down methods.

1.2. Related Work

Linkage methods are discussed in a number of research papers and books on data mining and machine learning. Here, we discuss works that provide provable guarantees for some of the most popular linkage methods.

Complete-link and Variants. Several upper and lower bounds are known on the approximation factor for `complete-linkage` with respect to the maximum diameter. When $\mathcal{X} = \mathbb{R}^d$, d is constant and $dist$ is the Euclidean metric, (Ackermann et al., 2010) proved that `complete-linkage` is an $O(\log k \cdot \text{OPT}_{\texttt{DM}}(k))$ approximation. This was improved by (Großwendt & Röglin, 2015) to $O(\text{OPT}_{\texttt{DM}}(k))$. The dependence on d is doubly exponential.

For general metric spaces, (Dasgupta & Long, 2005) showed that there are instances for which the maximum diameter of the k-clustering built by `complete-linkage` is $\Omega(\log k \cdot \text{OPT}_{\texttt{DM}}(k))$. In (Arutyunova et al., 2021) this lower bound was improved to $\Omega(k \cdot \text{OPT}_{\texttt{DM}}(k))$. Moreover, the same paper showed that the maximum diameter of `complete-linkage`'s k-clustering is $O(k^2\text{OPT}_{\texttt{DM}}(k))$. This result was recently improved by the same authors to $O(k^{1.59}\text{OPT}_{\texttt{DM}}(k))$ (Arutyunova et al., 2023). We note that the version of `complete-linkage`, analyzed in (Arutyunova et al., 2021; 2023), merges at each iteration the two clusters A and B for which $\texttt{diam}(A \cup B)$ is minimum. A consequence of Proposition 2.1, presented here, is that this rule is equivalent to the classical definition of `complete-linkage` presented at the beginning of Section 2.

(Arutyunova et al., 2023) also analysed `minimax` (Bien & Tibshirani, 2011), a linkage method related to `complete-linkage`, that merges at each iteration the two clusters A and B for which $A \cup B$ has the minimum ratio. They show that the max-diam of the k-clustering built by `minimax` is $\Theta(k\text{OPT}_{\texttt{DM}}(k))$. In Section 5, we show that its max-diam is also $O(k^{1,59}\text{OPT}_{\texttt{AV}}(k))$. One disadvantage of `minimax` is its computational efficiency, while `complete-linkage` admits an $O(n^2)$ implementation (Defays, 1977), no sub-cubic time implementation for `minimax` is known (Bien & Tibshirani, 2011).

Single-link. Among linkage methods, `single-linkage` is likely the one with the most extensive theoretical analysis (Kleinberg & Tardos, 2006; Dasgupta & Long, 2005; Arutyunova et al., 2023; Laber & Murtinho, 2023).

The works of (Dasgupta & Long, 2005; Arutyunova et al., 2023) are those that are more related to ours. The former shows that $\Omega(k \cdot \text{OPT}_{\texttt{DM}}(k))$ is a lower bound on the maximum diameter of `Single-Link` while the latter proves that this bound is tight. We note that our $\Omega(k^2 \cdot \text{OPT}_{\texttt{AV}}(k))$ lower bound improves over that of (Dasgupta & Long, 2005) since $k\text{OPT}_{\texttt{AV}}(k) \geq \text{OPT}_{\texttt{DM}}(k)$.

Average-link. (Dasgupta, 2016) introduced a global cost function defined over the tree induced by a hierarchical clustering and proposed algorithms to optimize it. (Cohen-Addad et al., 2019; Moseley & Wang, 2023) show that `average-linkage` achieves constant approximation with respect to variants of the cost functions proposed by (Dasgupta, 2016). (Charikar et al., 2019) proved that these analyses are tight.

Ward. Another popular linkage method was proposed by (Ward, 1963). (Großwendt et al., 2019) shows that Ward's method gives a 2-approximation for k-means when the optimal clusters are well-separated.

2. Preliminaries

Pseudo-code for `complete-linkage` is shown in Algorithm 2. The function $\texttt{dist}_{CL}(A, B)$ that measures the distance between clusters A and B is given by

$$\texttt{dist}_{\texttt{CL}}(A, B) := \max\{dist(a, b) | (a, b) \in A \times B\}.$$

Algorithm 2 Complete Link

```
1: C^0 ← clustering with n unitary clusters, each one containing
   a point of X
2: For i = 1, ..., n − 1
3:    (A, B) ← clusters in C_{i−1} s.t. dist_CL(A, B) is minimum
4:    C^i ← C^{i−1} ∪ {A ∪ B} − {A, B}
```

The following property of `complete-linkage`, whose proof can be found in the Appendix A, will be useful for our analysis. In particular, it implies that the rule employed by `complete-linkage` is equivalent to the rule analysed in (Arutyunova et al., 2023) that merges at each iteration the two clusters A and B for which $\texttt{diam}(A \cup B)$ is minimum.

Proposition 2.1. *Let A_j and A'_j be the clusters merged at the jth iteration of* `complete-linkage`. *Then,* $\texttt{diam}(A_j \cup A'_j) = \max\{dist(x, y) | (x, y) \in A_j \times A'_j\}$, *for every $j \geq 1$.*

Moreover, $\texttt{diam}(A_j \cup A'_j) \geq \texttt{diam}(A_{j-1} \cup A'_{j-1})$, *for every $j \geq 2$.*

We conclude this section with some useful notation. The term *family* is used to denote a set of clusters. For a family F, we use $|F|$ and $\texttt{Pts}(F)$, respectively, to denote the number of clusters in F and the set of points that belong to some cluster in F, that is, $\texttt{Pts}(F) = \bigcup_{g \in F} g$. Moreover, we use $\texttt{diam}(F)$ to denote the maximum distance between points that belong to $\texttt{Pts}(F)$.

3. A First Bound on the Diameter of Complete-link

In this section, we prove that the maximum diameter of the k-clustering built by `complete-linkage` is at most $k^{1.59}\text{OPT}_{\texttt{AV}}(k)$.

Fix a target k-clustering $\mathcal{T} = (T_1, \ldots, T_k)$. Our proof maintains a dynamic partition of the clusters produced by `complete-linkage` into families, where the diameter of each such family F can be bounded in terms of the diameters of some of the T_i's that it touches. We note that our bounds will depend on the choice of $\mathcal{T}$ and we can take the best possible $\mathcal{T}$ according to our objective. In this section, we take $\mathcal{T}$ to be the k-clustering with minimum `avg-diam`.

In Algorithm 3, we define how the families evolve along the execution of `complete-linkage`. At the beginning, each of the $|\mathcal{X}|$ points is a cluster. We then define our first partition as $(F_1, \ldots, F_k)$, where F_i is a family that contains $|T_i|$ clusters, each one being a point from T_i. Along the algorithm's execution, the families are organized in a directed forest D. Initially, the forest D consists of k isolated nodes, where the ith node corresponds to family F_i.

When `complete-linkage` merges the clusters g and g' belonging to the families F and F', respectively, a new family F^{new} is created and, in case (a) of Algorithm 3, a second new family $F^{new'}$ is also created. These new families contain all the clusters in F and F', except for g and g' that are replaced by the cluster $g \cup g'$. Moreover, F^{new} and $F^{new'}$ (when it is created) become parents of F and F' in D. The precise definition of the new families and how the forest D is updated are given by cases (a) and (b) in Algorithm 3.

Algorithm 3 PARTITIONING THE CLUSTERS OF `complete-linkage`

Create a clustering $\mathcal{C}^0$ with n unitary clusters, each one containing a point of $\mathcal{X}$
$\mathcal{T} = \{T_i | 1 \leq i \leq k\} \leftarrow k$-clustering that satisfies $\texttt{avg-diam}(\mathcal{T}) = \text{OPT}_{\texttt{AV}}(k)$
$F_i \leftarrow \{\{x\} | x \in T_i\}, \forall i$
$D \leftarrow$ forest comprised of k isolated nodes $F_1, \ldots, F_k$.
For $t := 1, \ldots, n-k$
    $(g, g') \leftarrow$ next clusters to be merged by `complete-linkage`
    $\mathcal{C}^t \leftarrow \mathcal{C}^{t-1} \cup \{g \cup g'\} - \{g, g'\}$
    Let F and F' be the families associated with the roots of D that respectively contain g and g'. Assume w.l.o.g. $|F| \geq |F'|$.
    Proceed according to the following exclusive cases:
    (**case a**) $|F'| = 1$ and $|F| > 1$
        $F^{new} \leftarrow F - \{g\}$; $F^{new'} \leftarrow \{g \cup g'\}$
        F.parent $\leftarrow F^{new}$; F'.parent $\leftarrow F^{new'}$
    (**case b**) $|F'| > 1$ or $|F| = 1$
        $F^{new} \leftarrow (F \cup F' \cup \{g \cup g'\}) - g - g'$
        F.parent $\leftarrow F^{new}$; F'.parent $\leftarrow F^{new}$

To prove our bound, we first show (Proposition 3.1) that at the beginning of each iteration, there exists a family, among those associated with some root of D, that contains at least two clusters. Then, we show an upper bound (Proposition 3.2) on the diameter of every family, with at least two clusters, created by Algorithm 3. Finally, in Theorem 3.3, this last result is used to upper bound the diameter of every cluster created by `complete-linkage`, based on a simple idea: if a cluster $g \cup g'$ is created at iteration t and H is a family containing two clusters, say h and h', at the beginning of t, then `complete-linkage` rule guarantees that $\texttt{diam}(g \cup g') \leq \texttt{diam}(h \cup h') \leq \texttt{diam}(H)$.

For our analysis, we need some extra terminology. Let $\texttt{leaves}(F)$ be the set of leaves of the subtree of D rooted at node/family F. We define $\phi(F) := |\texttt{leaves}(F)|$ and $\phi_\Sigma(F) := \sum_{H \in \texttt{leaves}(F)} \texttt{diam}(H)$. Note that if a family F^{new} is parent of both families F and F' in D then $\phi(F^{new}) = \phi(F) + \phi(F')$ and $\phi_\Sigma(F^{new}) = \phi_\Sigma(F) + \phi_\Sigma(F')$ Moreover, we say that a family F is *regular* if $|F| > 1$ and it is a *singleton* if $|F| = 1$.

Proposition 3.1. *At the beginning of each iteration of Algorithm 3, at least one of the roots of D corresponds to a regular family.*

Proof. Initially, the total number of roots of D is k. Since the number of roots either decreases or remains the same, the number of roots at the beginning of each iteration is at most k. At the beginning of iteration t, for $t \leq n-k$, the `complete-linkage` clustering $\mathcal{C}^t$ has more than k clusters, each of them belonging to one family that is a root of D. Since the number of roots is at most k, then there will be two different clusters associated with the same root, so that this root corresponds to a regular family. □

Proposition 3.2. *At the beginning of each iteration of Algorithm 3 the diameter of every regular family F satisfies* $\texttt{diam}(F) \leq \phi_\Sigma(F) \cdot \phi(F)^{(\log_2 3)-1} \leq k^{\log_2 3} OPT_{\texttt{AV}}(k)$.

Proof. We have that $\phi(F) \leq k$, Moreover, the choice of the target clustering $\mathcal{T}$ ensures that $\phi_\Sigma(F) \leq k\text{OPT}_{\texttt{AV}}(k)$. Hence, the inequality $\phi_\Sigma(F)\phi(F)^{(\log_2 3)-1} \leq k^{\log_2 3}\text{OPT}_{\texttt{AV}}(k)$ holds. Thus, we focus on the first inequality.

The proof is by induction on the iteration of `complete-linkage` (and, in parallel, of Algorithm 3). For every initial family F_i, $\phi(F_i) = 1$ and $\phi_\Sigma(F_i) = \texttt{diam}(F_i)$. Thus, for every F_i, $\texttt{diam}(F_i) \leq \phi_\Sigma(F_i)\phi(F_i)^{(\log_2 3)-1}$.

Let us assume by induction that the result at the beginning of iteration t. We consider what happens in iteration t according to the possible cases:

case (a). In this case, $F^{new'}$ is a singleton so we do not need to argue about it since the property is about regular families. Moreover, we have that

$$\texttt{diam}(F^{new}) = \texttt{diam}(F - \{g\}) \leq \texttt{diam}(F) \leq \phi_\Sigma(F)\phi(F)^{\log_2 3 - 1} = \phi_\Sigma(F^{new})\phi(F^{new})^{\log_2 3 - 1},$$

where the last inequality holds by induction and the last identity holds because $\phi_\Sigma(F^{new}) = \phi_\Sigma(F)$ and $\phi(F^{new}) = \phi(F)$.

case (b) We split the proof into 3 subcases:

subcase 1. $|F| = 1$ and $|F'| = 1$. In the case $F^{new} = \{g \cup g'\}$, so it is a singleton and, thus, there is nothing to argue since the property is about regular families.

subcase 2. $|F'| > 1$ and F=F'. In this case, we have

$$\texttt{diam}(F^{new}) = \texttt{diam}(F) \leq \phi_\Sigma(F)\phi(F)^{\log_2 3 - 1} = \phi_\Sigma(F^{new})\phi(F^{new})^{\log_2 3 - 1},$$

where the inequality holds by induction and the last identity holds because $\phi_\Sigma(F^{new}) = \phi_\Sigma(F)$ and $\phi(F^{new}) = \phi(F)$.

subcase 3. $|F'| > 1$ and $F \neq F'$. This case is the most interesting one. In this case, `complete-linkage` creates a new family F^{new} by merging two clusters g and g' from two distinct regular families F and F'. Let a and b be two farthest points in $\mathtt{Pts}(F^{new})$. If $a, b \in \mathtt{Pts}(F)$ or $a, b \in \mathtt{Pts}(F')$ the result holds for F^{new} since

$$\begin{aligned}\mathtt{diam}(F^{new}) \leq \max\{\mathtt{diam}(F), \mathtt{diam}(F')\} \leq \\ \max\{\phi_\Sigma(F) \cdot \phi(F)^{\log_2 3-1}, \phi_\Sigma(F') \cdot \phi(F')^{\log_2 3-1}\} \leq \\ \leq \phi_\Sigma(F^{new})\phi(F^{new})^{\log_2 3-1}\end{aligned}$$

Let $a \in \mathtt{Pts}(F)$, $b \in \mathtt{Pts}(F')$. We can assume w.l.o.g. that

$$\phi_\Sigma(F') \cdot \phi(F')^{(\log_2 3)-1} \leq \phi_\Sigma(F) \cdot \phi(F)^{(\log_2 3)-1}.$$

Note that this assumption will not conflict with the assumption $|F| \geq |F'|$ that was made to facilitate the presentation of Algorithm 3. Indeed, we do not use the assumption $|F| \geq |F'|$ in what follows.

Let $a' \in g$ and $b' \in g'$ be points that satisfy $dist(a', b') = \min\{dist(x, y)|(x, y) \in g \times g'\}$. Moreover, let h and h' be any two clusters in F. We have that

$$dist(a', b') \leq \max\{dist(x, y)|(x, y) \in g \times g'\} \leq \quad (1)$$
$$\max\{dist(x, y)|(x, y) \in h \times h'\} \leq \mathtt{diam}(h \cup h') \leq \quad (2)$$
$$\mathtt{diam}(F), \quad (3)$$

where the second inequality follows from `complete-linkage` rule.

By symmetry we also have $dist(a', b') \leq \mathtt{diam}(F')$ and, hence

$$dist(a', b') \leq \min\{\mathtt{diam}(F), \mathtt{diam}(F')\} \quad (4)$$

Consider the sequence of points a, a', b', b. It follows from the triangle inequality that

$$\mathtt{diam}(F^{new}) = dist(a, b) \leq \quad (5)$$
$$dist(a, a') + dist(a', b') + dist(b', b) \leq \quad (6)$$
$$\mathtt{diam}(F) + \mathtt{diam}(F') + \mathtt{diam}(F') \leq \quad (7)$$
$$\phi_\Sigma(F)\phi(F)^{\log_2 3-1} + 2\phi_\Sigma(F')\phi(F')^{\log_2 3-1} \leq \quad (8)$$
$$(\phi_\Sigma(F) + \phi_\Sigma(F'))(\phi(F') + \phi(F))^{\log_2 3-1} = \quad (9)$$
$$\phi_\Sigma(F^{new})\phi(F^{new})^{\log_2 3-1}, \quad (10)$$

where inequality (6) follows from (4), inequality (7) follows from the inductive hypothesis, inequality (8) follows from Proposition G.1 (with $a = \phi_\Sigma(F)$, $b = \phi_\Sigma(F')$, $x = \phi(F)$ and $y = \phi(F')$) and (9) holds because $\phi(F^{new}) = \phi(F) + \phi(F')$ and $\phi_\Sigma(F^{new}) = \phi_\Sigma(F) + \phi_\Sigma(F')$. □

Now, we state and prove the main result of this section.

Theorem 3.3. *For every k, the maximum diameter of the k-clustering built by* `complete-linkage` *is at most $k^{\log_2 3} OPT_{\mathtt{AV}}(k)$.*

Proof. We prove by induction on the iteration of `complete-linkage` (and, in parallel, of Algorithm 3) that the diameter of each cluster created by `complete-linkage` is at most $k^{\log_2 3}\mathrm{OPT}_{\mathtt{AV}}(k)$. At the beginning, we have n clusters, each of them corresponding to a point, so that for every initial cluster A, $\mathtt{diam}(A) = 0 \leq k^{\log_2 3}\mathrm{OPT}_{\mathtt{AV}}(k)$. We assume by induction that at the beginning of iteration t every cluster satisfies the desired property,

Let g and g' be two clusters merged at iteration t. By Proposition 3.1 there is a regular family F at the beginning of the t-th iteration. Let h and h' be two clusters in F. Therefore,

$$\mathtt{diam}(g \cup g') = \quad (11)$$
$$\max\{\mathtt{diam}(g), \mathtt{diam}(g'), \mathtt{dist}_{CL}(g, g')\} \leq \quad (12)$$
$$\max\{\mathtt{diam}(g), \mathtt{diam}(g'), \mathtt{dist}_{CL}(h, h')\} \leq \quad (13)$$
$$\max\{\mathtt{diam}(g), \mathtt{diam}(g'), \mathtt{diam}(h \cup h')\} \leq \quad (14)$$
$$\max\{\mathtt{diam}(g), \mathtt{diam}(g'), \mathtt{diam}(F)\} \leq \quad (15)$$
$$k^{1.59}\mathrm{OPT}_{\mathtt{AV}}(k), \quad (16)$$

where the first inequality holds due to the choice of `complete-linkage` and the last one from the induction hypothesis and Proposition 3.2. □

`Single-linkage` is a popular linkage method whose pseudo-code is obtained by replacing $\mathtt{dist_{CL}}$ with $\mathtt{dist_{SL}}$ in Algorithm 2, where $\mathtt{dist_{SL}}(A, B) := \min\{dist(a, b)|(x, y) \in A \times B\}$.

The rule employed by `single-linkage`, in contrast to that of `complete-linkage`, is not greedy with respect to the minimization of the diameter. Thus, it is expected that the latter presents better bounds than the former. However, perhaps surprisingly, this is not the case when we consider approximation regarding to $\mathrm{OPT}_{\mathtt{DM}}$ since the maximum diameter of the latter is $\Omega(\mathrm{OPT}_{\mathtt{DM}}(k))$ while that of the former is $\Theta(\mathrm{OPT}_{\mathtt{DM}}(k))$ (Arutyunova et al., 2023).

The use of $\mathrm{OPT}_{\mathtt{AV}}$, instead of $\mathrm{OPT}_{\mathtt{DM}}$, allows a separation between `complete-linkage` and `single-linkage` in terms of worst-case approximation. In fact, Theorem 3.3 shows that the maximum diameter of `complete-linkage` is at most $k^{1.59}\mathrm{OPT}_{\mathtt{AV}}(k)$ while the next result shows that the maximum diameter of `single-linkage` is $\Omega(k^2\mathrm{OPT}_{\mathtt{AV}}(k))$.

Theorem 3.4. *There is an instance in which the k-clustering produced by* `single-linkage` *includes a cluster of diameter $\Omega(k^2 OPT_{\mathtt{AV}}(k))$.*

Proof. We present a simple instance for which the k-clustering produced by `single-linkage` has a cluster of diameter $\Omega(k^2 \text{OPT}_{\texttt{AV}}(k))$. Let B be a large positive number and let us consider k groups $G_1, \ldots, G_k$: G_1 consists of 2 points a and b, with $dist(a, b) = B$; the group G_i, for $1 < i < k$ is a singleton, containing only the point x_i; and the group G_k consists of $k-1$ points $y_1, \ldots, y_{k-1}$, with $dist(y_i, y_j) = B + \epsilon$ for all i and j.

Moreover, we have that $dist(x_i, a) = dist(x_i, b) = (i-1) \times (B - \epsilon)$ for $i = 2, \ldots, k-1$ and $dist(x_i, x_j) = (j-i)(B-\epsilon)$, for $1 < i < j < k$. Finally, the distance of any point in G_k to a point outside G_k is $2B$.

Note that $(G_1, \ldots, G_k)$ is a k-clustering and the average diameter of its clusters is $(2B+\epsilon)/k$. On the other hand, `single-linkage` builds the $k-$clustering $(G_1 \cup \cdots \cup G_{k-1}, \{y_1\}, \ldots, \{y_{k-1}\})$ and the cluster $(G_1 \cup \cdots \cup G_{k-1})$ has diameter $(k-1)(B-\epsilon)$. □

4. A Better Bound for Complete-Link

One of the key ideas of the approach presented in the previous section is to use the diameter of a regular family to bound the diameter of any cluster that is created. Indeed, if at the beginning of an iteration, there is a family F with two clusters, then the diameter of the cluster created at this iteration is at most $\texttt{diam}(F)$. However, Algorithm 3 and its analysis do not take full advantage of this idea. As an example, let us assume that at the beginning of some iteration there are 3 regular families, say F, F' and F'', with $\texttt{diam}(F) \geq \texttt{diam}(F') \geq \texttt{diam}(F'')$, all of them corresponding to roots of D. If a cluster in F is merged with one in F' (case (b) of Algorithm 3) then a new family is created and its diameter is used as a bound, which is not desirable since it is larger than that of F''.

To obtain a better bound, instead of creating a new family whenever clusters from different families, say F and F', are merged, we create an edge between F and F' in a dynamic graph G that keeps track of the merges among different families. When a connected component of G has at most one family that can still be used as a bounding tool, we replace all families in the component with a new family. The motivation for doing so is to use a better bound as much as possible, which contrasts with the approach taken by Algorithm 3.

This new approach is presented in Algorithm 4. The algorithm maintains a set of excluded clusters $\mathcal{E}$; clusters in this set are never included in the families that the algorithm creates (line 4). Moreover, it maintains both a direct forest D and a graph G. Each node of G as well as each node of D is associated with a family; an edge is created in G between nodes/families F and F' if `complete-linkage` merges two clusters $g \notin \mathcal{E}$ and $g' \notin \mathcal{E}$ that, respectively, contain points from $\texttt{Pts}(F)$ and $\texttt{Pts}(F')$. The graph and the forest may be updated at each iteration of Algorithm 4.

Before giving extra details regarding Algorithm 4, we need to explain the concept of a pure cluster. A family F is created (lines 4 and 4) by specifying the clusters that it contains. If a cluster g is one of them, we say that g is *pure* w.r.t. F or, alternatively, F has the pure cluster g. Moreover, if a cluster g is obtained by merging two clusters that are pure with respect to some family F then g is also pure w.r.t. F. If a cluster is not pure w.r.t. any family, we say that it is *non-pure*. We use $\texttt{pure}_t(F)$ to denote the number of pure clusters w.r.t. F that belong to $\mathcal{C}^t$. Note that if $pure_{t-1}(F) \geq 2$ then $\texttt{diam}(F)$ is an upper bound on the diameter of the cluster that is created at iteration t, so that families with at least two pure clusters play a role similar to that of regular families in the analysis of Algorithm 3.

In contrast to Algorithm 3, where each cluster belongs to one family, in Algorithm 4 every cluster that does not belong to $\mathcal{E}$ is either pure w.r.t. some family F in G (this would be equivalent of belonging to F) or it is contained in $\bigcup_{H \in C} \texttt{Pts}(H)$ for some connected component C in G. For our analysis, we note that $\texttt{Pts}(F)$, $\texttt{diam}(F)$ and $|F|$ refer, respectively, to the set of points of F, the diameter of F and the number of clusters in F **at the moment** that F is created.

The algorithm starts (lines 4-4) with the initialization of the set $\mathcal{E}$, the forest D and the graph G. Then, in the loop, two clusters are merged following the `complete-linkage` rule. In terms of the graph, each merge may lead to the addition of new edges and also to the union of two connected components. In terms of the families, a merge can reduce by one unit the number of pure clusters of one or two families. If this happens pure clusters may be added to set $\mathcal{E}$ (lines 4 and 4) and this may also trigger one of the cases (a), (b) or (c). If either (a) or (b) occurs a new family F_C, associated with the component C that satisfies one of these cases, is created to replace all families in C (line 4). If case (c) occurs the component C is removed from G.

The main loop was carefully designed to guarantee that (i) at the beginning of each iteration there exists a family that has at least two pure clusters associated with it and (ii) the diameter of family F_C is slightly smaller than twice the sum of the diameters of the families in the underlying connected component C.

The roadmap to establish our improved bound (Theorem 4.5) consists of first showing that (i) holds (Lemma 4.1) and, then, showing an upper bound on the diameters of the families F_C that are created in line 4. This upper bound will be used to bound the diameter of every cluster that is created by `complete-linkage`. Note that our strategy is similar to that employed to prove Theorem 3.3. However,

Algorithm 4 TIGHTER BOUND FOR `complete-linkage`

1: $\mathcal{C}^0 \leftarrow$ clustering with n unitary clusters, each one containing a point of $\mathcal{X}$
2: $(T_1^*, \ldots, T_k^*) \leftarrow$ a k-clustering with maximum diameter equal to $\text{OPT}_{\texttt{DM}}(k)$
3: For each i, with $|T_i^*| > 1$, $F_i \leftarrow \{\{x\} | x \in T_i^*\}$
4: Create a forest D with no edges and vertex set $\{F_i | T_i^*$ has at least two points$\}$
5: Create a graph G with no edges and vertex set $\{F_i | T_i^*$ has at least two points$\}$
6: $\mathcal{E} \leftarrow$ set of clusters T_i^* with exactly one point
7: **For** $t := 1 \ldots n-k$
8: $\quad (g, g') \leftarrow$ next clusters to be merged by `Complete-Link`
9: $\quad \mathcal{C}^t \leftarrow \mathcal{C}^{t-1} \cup \{g \cup g'\} - \{g, g'\}$
10: $\quad$ **If** g or g' is a cluster in $\mathcal{E}$
11: $\quad\quad$ Add $g \cup g'$ to $\mathcal{E}$ and remove from $\mathcal{E}$ the clusters in $\{g, g'\}$ that belong to $\mathcal{E}$
12: $\quad$ **Else**
13: $\quad\quad$ Create edges between all families F and F' such that $\texttt{Pts}(F)$ has a point in g and $\texttt{Pts}(F')$ has a point in g'
14: $\quad$ Consider the following exclusive cases:
15: $\quad$ (a) $\exists$ connected component C in G, with $|C| > 1$, that has exactly one family F such that $\texttt{pure}_t(F) > 1$
16: $\quad$ (b) $\exists$ connected component C in G, with $|C| > 1$, such that every family F in C satisfies $\texttt{pure}_t(F) \leq 1$
17: $\quad$ (c) $\exists$ connected component C in G, with $|C| = 1$, and its only family F satisfies $\texttt{pure}_t(F) \leq 1$
18: $\quad$ **If** (b) does not occur
19: $\quad\quad$ For each family H in G that satisifies $\texttt{pure}_{t-1}(H) > 1$ and $\texttt{pure}_t(H) = 1$
20: $\quad\quad\quad$ Add the pure cluster in H to $\mathcal{E}$
21: $\quad$ **If** (b) occurs
22: $\quad\quad$ $H \leftarrow$ some family in C such that $\texttt{pure}_{t-1}(H) > 1$ and $\texttt{pure}_t(H) = 1$
23: $\quad\quad$ Add the pure cluster in H to $\mathcal{E}$
24: $\quad$ **If** either (a) or (b) occurs
25: $\quad\quad$ Create family $F_C := \{h | h \in \mathcal{C}^t$ and $h \subseteq \bigcup_{H \in C} \texttt{Pts}(H)\} \setminus \mathcal{E}$
26: $\quad\quad$ Set F_C as the parent, in the forest D, of every family of C
27: $\quad\quad$ Add to G a node corresponding to F_C
28: $\quad\quad$ Remove all families in the connected component C from G
29: $\quad$ **If** (c) occurs
30: $\quad\quad$ Remove all families in the connected component C from G

the proofs here are significantly more involved.

We start with Lemma 4.1. We present a sketch of the proof and we refer to Appendix B for the full proof.

Lemma 4.1. *For $t \leq n-k$, at the beginning of iteration t of Algorithm 4, each connected component C of G satisfies one of the following properties: (i) $|C| = 1$ and the only family of C has at least two pure clusters or (ii) $|C| > 1$ and there exist two families in C such that each of them has at least two pure clusters.*

Proof Sketch. We first argue that if all components of G satisfy the desired properties at the beginning of iteration t then all components of G also satisfy them at the beginning of iteration $t+1$. Next, we argue that G does not have all its nodes removed at some iteration.

At the beginning of Algorithm 4, all the components in G satisfy property (i) because, by line 4, all the families F_i in G have at least two clusters and all their clusters are pure.

When two clusters are merged at some iteration t, then at most two distinct families have their number of pure clusters decreased by one unit (Proposition B.1). As a result, one connected component, say C, where these families lie in the updated graph may not respect the conditions of the lemma anymore. However, in this case, we can show that either (a), (b) or (c) occurs. In the case (c), the component C is removed from G, so we do not have a problem with C at the next iteration. If either (a) or (b) occurs, C is replaced with a new component that only has the family F_C. Proposition B.2 shows that there are two pure clusters w.r.t. F_C, so this new component satisfies the condition (i).

Now, assume that G has all its nodes removed at some iteration t'. It is possible to conclude that at the beginning of t', G has just one component, this component has just one family and this family has exactly 2 pure clusters. This together with the fact that at most k clusters are added to $\mathcal{E}$ (Proposition B.4) allows the conclusion that there are at most $k+1$ clusters at the beginning of t'. But this is not a problem since $t' \geq n-k$ in this case. $\square$

Now, we bound the diameter of the families F_C at the moment they are created by Algorithm 4. To this end, we define a spanning tree T_C for C and use its paths to bound the diameter of F_C. Consider the sequence of merges $m_1, \ldots, m_{|C|-1}$, between clusters, that builds the connected component C, that is, right after each merge at least two families in C that were not connected become connected. Moreover, let g_i be cluster produced by merging m_i. The nodes of T_C are the families in C and the edges of T_C are defined as follows: for each merge m_i we create an edge e_i between two arbitrarily chosen families, say F^1 and F^2, among those that were not connected before merge m_i and also have points in g_i, that is, $\texttt{Pts}(F^1) \cap g_i \neq \emptyset$ and $\texttt{Pts}(F^2) \cap g_i \neq \emptyset$. The weight of e_i is given by the diameter of g_i.

For the following results, let $\texttt{DM}_i$ be the ith smallest diameter among the families that belong to C.

Proposition 4.2. *The weight of the cheapest edge of T_C is at most $\texttt{DM}_1$ and, for $i > 1$, the weight of its ith cheapest edge is at most $\texttt{DM}_{i-1}$.*

The proof of Proposition 4.2 can be found in the Section C. The key observation is that the weight of e_i is not larger than the diameter of families that have at least two pure clusters right before merge m_i and it is also not larger than the diameter of families in C that have not been created when the merge m_i occurs. By arguing that there at least $|C| - i + 2$ families that satisfy one of these conditions, we establish the proof.

The next proposition gives an upper bound on the diameter of F_C as a function of the diameters of the families in the component C associated with F_C. In high-level, its proof considers the path P in T_C between the families where the two farthest points in $\texttt{Pts}(F_C)$ lie and then use the triangle inequality to show that the distance between these points is upper bounded by the sum of the weights of the edges in P plus the sum of the diameters of the nodes/families in P. This sum, however, is upper bounded by the sum of the diameters of all the families in C plus the sum of the weights of the edges in T_C, so that

$$\texttt{diam}(F_C) \leq \sum_{i=1}^{|C|} \texttt{DM}_i + \left(\texttt{DM}_1 + \sum_{i=1}^{|C|-2} \texttt{DM}_i \right).$$

The proposition, in fact, shaves $\texttt{DM}_1$ from the above upper bound via a more careful analysis. Its proof can be found in Section D.

Proposition 4.3. *Let F_C be a family associated with the connected component C of G in line 4 of Algorithm 4. Then, when F_C is created, we have*

$$\texttt{diam}(F_C) \leq \sum_{i=1}^{|C|} \texttt{DM}_i + \sum_{i=1}^{|C|-2} \texttt{DM}_i$$

For the next lemma recall that $\phi(F) = |\texttt{leaves}(F)|$, where $\texttt{leaves}(F)$ is the set of leaves in the subtree of D rooted at node/family F.

Let $\alpha = \max\{\frac{\log(2i-2)}{\log i} | i$ is a natural number larger than $1\}$. Proposition G.2 shows that $\alpha = \frac{\log 6}{\log 4} < 1.30$. Moreover, we define $\alpha_k = \log_k(2k-2)$, if $k \leq 4$, and $\alpha_k = \alpha$ for $k > 4$.

Lemma 4.4. *Every family F created by Algorithm 4 satisfies $\texttt{diam}(F) \leq OPT_{\texttt{DM}}(k)\phi(F)^{\alpha_k}$.*

Proof. The initial families F_i satisfies the property because $\texttt{diam}(F_i) = \texttt{diam}(T_i^*) \leq \text{OPT}_{\texttt{DM}}(k) \leq \text{OPT}_{\texttt{DM}}(k)\phi(F_i)^{\alpha_k}$ since $\phi(F_i) = 1$.

Let us assume that the result holds at the beginning of iteration t. If no family is created at iteration t the result holds at the beginning of iteration $t+1$. Otherwise, a family F_C, associated with a connected component C, is created. Let $\{F_C^i | i = 1, \ldots, |C|\}$ be the nodes/families in C right before the creation of F_C. Moreover, assume that $\phi(F_C^i) \leq \phi(F_C^{i+1})$. We have that

$$\texttt{diam}(F_c) \leq \tag{17}$$

$$\sum_{i=1}^{|C|} \texttt{DM}_i + \sum_{i=1}^{|C|-2} \texttt{DM}_i \leq \tag{18}$$

$$\sum_{i=1}^{|C|} \texttt{diam}(F_C^i) + \sum_{i=1}^{|C|-2} \texttt{diam}(F_C^i) \leq \tag{19}$$

$$\text{OPT}_{\texttt{DM}}(k) \cdot \left(\sum_{i=1}^{|C|} \phi(F_C^i)^{\alpha_k} + \sum_{i=1}^{|C|-2} \phi(F_C^i)^{\alpha_k} \right) \leq \tag{20}$$

$$\text{OPT}_{\texttt{DM}}(k) \left(\sum_{i=1}^{|C|} \phi(F_C^i) \right)^{\alpha_k} = \text{OPT}_{\texttt{DM}}(k)\phi(F_C)^{\alpha_k}, \tag{21}$$

where (17) follows from Proposition 4.3; (18) holds because $\texttt{DM}_1, \ldots, \texttt{DM}_{|C|-2}$ are the $|C| - 2$ smallest diameters among the diameters of the families in C; (19) follows from the inductive hypothesis and (20) follows from Proposition G.3, using $a_i = \phi(F_C^i)$, $\ell = |C|$ and $p = \alpha_k$. $\square$

Theorem 4.5 is the main result of this section. Its proof is similar to that of Theorem 3.3 and can be found in Appendix E.

Theorem 4.5. *The maximum diameter among the clusters of the k-clustering produced by `complete-linkage` is at most $(2k-2)OPT_{\texttt{DM}}(k)$, if $k \leq 4$, and at most $k^{1.30}OPT_{\texttt{DM}}(k)$, if $k > 4$.*

We note that $k = 2$, $k = 3$ and $k = 4$, we get approximation factors of 2, 4 and 6, respectively. For $k > 4$ the approximation factor is $k^{\log_4 6} \leq k^{1.30}$.

5. Other Linkage Methods

In this last section, we show that Theorem 3.3 generalizes to a class of linkage methods that includes `minimax` and the quite popular `average-linkage`.

Let f be a distance function that maps a pair of clusters into a non-negative real number and let $\texttt{Link}_f$ be a linkage method that follows the pseudo-code of Algorithm 2, with the exception that it uses the function f, rather than $\texttt{dist}_{CL}$, to measure the distance between two clusters. Moreover, for a cluster A, let $\texttt{cost}(A)$ be a cohesion criterion (e.g. diameter). We say that f and $\texttt{cost}$ *align* if they satisfy the following conditions for every pair of disjoint clusters A and B:

(i) $\min\{\texttt{dist}(a,b)|(a,b) \in A \times B\} \leq f(A,B) \leq \texttt{diam}(A \cup B)$;

(ii) $\texttt{cost}(A) = 0$ if $|A| = 1$;

(iii) $\texttt{cost}(A \cup B) \leq \min\{\texttt{cost}(A), \texttt{cost}(B), f(A,B)\}$

Theorem 5.2 presented below is a generalization of Theorem 3.3. In fact, from the former, we can recover the latter by setting $\texttt{cost} = \texttt{diam}$ and $f = \texttt{dist}_{CL}$. The proof of Theorem 5.2 is essentially the same as that of Theorem 3.3, but for a few differences that we explain in what follows.

The proof of Theorem 5.2 is based on the analysis of the families generated by the variation of Algorithm 3 that uses a distance function f that satisfies (i), rather than $\texttt{dist}_{CL}$, to decide which clusters are merged at each iteration. We use $\texttt{Algo}_f$ to denote this modified version of Algorithm 3.

Proposition 3.1 does not depend on the distance function employed to decide which clusters shall be merged at each iteration, so it is still valid for $\texttt{Algo}_f$.

The following proposition generalizes Proposition 3.2 for linkage methods whose underlying distances satisfy condition (i).

Proposition 5.1. *If f satisfies condition (i), then at the beginning of each iteration of $\texttt{Algo}_f$ the diameter of every regular family F satisfies* $\texttt{diam}(F) \leq \phi_\Sigma(F) \cdot \phi(F)^{(\log_2 3)-1} \leq k^{\log_2 3} OPT_{\texttt{AV}}(k)$.

Proof. In Proposition 3.2, the complete-linkage's rule is just used to prove inequalities (1)-(3). However, these inequalities are valid if the function f satisfies condition (i). In fact, we have

$$dist(a', b') \leq f(g, g) \leq f(h, h') \leq \texttt{diam}(h \cup h') \leq \texttt{diam}(F),$$

where the first and the third inequality hold due to condition (i) while the second holds due to the choice of $\texttt{Link}_f$. $\square$

Now, we have the elements required for the proof of Theorem 5.2. This proof can be found in Section F.

Theorem 5.2. *If f and $\texttt{cost}$ align, then the k-clustering $\mathcal{C}$ built by $\texttt{Link}_f$ satisfies* $\max\{\texttt{cost}(C)|C \in \mathcal{C}\} \leq k^{1.59} OPT_{\texttt{AV}}(k)$.

Now, we specialize Theorem 5.2 for average-linkage and minimax. average-linkage employs the distance function

$$\texttt{dist}_{AL}(A,B) = \frac{1}{|A| \cdot |B|} \sum_{a \in A} \sum_{b \in B} dist(a,b)$$

to measure the distance between clusters A and B. Clearly, $\texttt{dist}_{AL}$ satisfies condition (i). For a cluster A, we define $\texttt{avg}(A)$ as 0 if $|A| = 1$ and as the average pairwise distance of the points in A if $|A| > 1$, that is,

$$\texttt{avg}(A) := \frac{2}{|A|(|A|-1)} \sum_{x,y \in A} dist(x,y).$$

Since $\texttt{avg}(A \cup B)$ is a convex combination of $\texttt{avg}(A)$, $\texttt{avg}(B)$ and $\texttt{dist}_{AL}(A,B)$, condition (iii) is also satisfied and, therefore, $\texttt{dist}_{AL}$ and $\texttt{avg}$ align. We have the following result.

Theorem 5.3. *For every k, the k-clustering $\mathcal{C}$ built by average-linkage satisfies* $\max\{\texttt{avg}(C)|C \in \mathcal{C}\} \leq k^{1.59} OPT_{\texttt{AV}}(k)$.

Now we consider the minimax linkage method. This method employs the function $\texttt{dist}_{MM}(A,B) := \min_{x \in A \cup B} \max_{y \in A \cup B} dist(x,y)$ to measure the distance between clusters.

We have that $\texttt{dist}_{MM}$ satisfies (i). Consider the cohesion criterion $\texttt{radius}(A)$ that has value 0 if $|A| = 1$ and when $|A| > 1$, $\texttt{radius}(A) := \min_{x \in A} \max_{y \in A} dist(x,y)$. Since $\texttt{radius}(A \cup B) = \texttt{dist}_{MM}(A,B)$ the condition (iii) is also satisfied and, hence, $\texttt{dist}_{MM}$ and $\texttt{radius}$ align. We have that

Theorem 5.4. *For every k, the k-clustering $\mathcal{C}$ built by minimax satisfies* $\max\{\texttt{radius}(C)|C \in \mathcal{C}\} \leq k^{1.59} OPT_{\texttt{AV}}(k)$.

Acknowledgements

The first author thanks the National Science Foundation for support under grant IIS-2211386. The work of the second author is partially supported by the Air Force Office of Scientific Research (award number FA9550-22-1-0475) and by CNPq (grant 310741/2021-1)

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ackerman, M. and Ben-David, S. A characterization of linkage-based hierarchical clustering. *J. Mach. Learn.*

Res., 17:232:1–232:17, 2016. URL http://jmlr.org/papers/v17/11-198.html.

Ackermann, M. R., Blömer, J., Kuntze, D., and Sohler, C. Analysis of agglomerative clustering. *CoRR*, abs/1012.3697, 2010. URL http://arxiv.org/abs/1012.3697.

Arutyunova, A. and Röglin, H. The price of hierarchical clustering. In Chechik, S., Navarro, G., Rotenberg, E., and Herman, G. (eds.), *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*, volume 244 of *LIPIcs*, pp. 10:1–10:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPICS.ESA.2022.10. URL https://doi.org/10.4230/LIPIcs.ESA.2022.10.

Arutyunova, A., Großwendt, A., Röglin, H., Schmidt, M., and Wargalla, J. Upper and Lower Bounds for Complete Linkage in General Metric Spaces. In Wootters, M. and Sanità, L. (eds.), *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 18:1–18:22, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-207-5. doi: 10.4230/LIPIcs.APPROX/RANDOM.2021.18. URL https://drops.dagstuhl.de/opus/volltexte/2021/14711.

Arutyunova, A., Großwendt, A., Röglin, H., Schmidt, M., and Wargalla, J. Upper and lower bounds for complete linkage in general metric spaces. *Machine Learning*, pp. 1–30, 2023.

Bien, J. and Tibshirani, R. Hierarchical clustering with prototypes via minimax linkage". *Journal of the American Statistical Association*, 106:1075–1084, 2011.

Charikar, M., Chekuri, C., Feder, T., and Motwani, R. Incremental clustering and dynamic information retrieval. *SIAM J. Comput.*, 33(6):1417–1440, 2004. doi: 10.1137/S0097539702418498. URL https://doi.org/10.1137/S0097539702418498.

Charikar, M., Chatziafratis, V., and Niazadeh, R. Hierarchical clustering better than average-linkage. In Chan, T. M. (ed.), *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pp. 2291–2304. SIAM, 2019. doi: 10.1137/1.9781611975482.139. URL https://doi.org/10.1137/1.9781611975482.139.

Cohen-Addad, V., Kanade, V., Mallmann-Trenn, F., and Mathieu, C. Hierarchical clustering: Objective functions and algorithms. *J. ACM*, 66(4):26:1–26:42, 2019. doi: 10.1145/3321386. URL https://doi.org/10.1145/3321386.

Dasgupta, S. A cost function for similarity-based hierarchical clustering. In Wichs, D. and Mansour, Y. (eds.), *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pp. 118–127. ACM, 2016. doi: 10.1145/2897518.2897527. URL https://doi.org/10.1145/2897518.2897527.

Dasgupta, S. and Long, P. M. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569, 2005. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2004.10.006. URL https://www.sciencedirect.com/science/article/pii/S0022000004001321. Special Issue on COLT 2002.

Defays, D. An efficient algorithm for a complete link method. *Comput. J.*, 20(4):364–366, 1977. doi: 10.1093/COMJNL/20.4.364. URL https://doi.org/10.1093/comjnl/20.4.364.

Eisen, M., Spellman, P., Brown, P., and Botstein, D. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, 95(25): 14863–14868, December 1998. ISSN 0027-8424. doi: 10.1073/pnas.95.25.14863.

Großwendt, A. and Röglin, H. Improved analysis of complete-linkage clustering. In Bansal, N. and Finocchi, I. (eds.), *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pp. 656–667. Springer, 2015. doi: 10.1007/978-3-662-48350-3_55. URL https://doi.org/10.1007/978-3-662-48350-3_55.

Großwendt, A., Röglin, H., and Schmidt, M. Analysis of ward's method. In Chan, T. M. (ed.), *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pp. 2939–2957. SIAM, 2019. doi: 10.1137/1.9781611975482.182. URL https://doi.org/10.1137/1.9781611975482.182.

Hennig, C., Meila, M., Murtagh, F., and Rocci, R. *Handbook of Cluster Analysis*. Chapman and Hall/CRC, 2015.

Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, September 1999. ISSN 0360-0300.

Kleinberg, J. M. and Tardos, É. *Algorithm design*. Addison-Wesley, 2006. ISBN 978-0-321-37291-8.

Laber, E. S. and Murtinho, L. Optimization of inter-group criteria for clustering with minimum size constraints. In *NeurIPS*, 2023.

Lin, G., Nagarajan, C., Rajaraman, R., and Williamson, D. P. A general approach for incremental approximation and hierarchical clustering. *SIAM J. Comput.*, 39(8):3633–3669, 2010. doi: 10.1137/070698257. URL https://doi.org/10.1137/070698257.

Megiddo, N. On the complexity of some geometric problems in unbounded dimension. *J. Symb. Comput.*, 10(3/4):327–334, 1990. doi: 10.1016/S0747-7171(08)80067-3. URL https://doi.org/10.1016/S0747-7171(08)80067-3.

Moseley, B. and Wang, J. R. Approximation bounds for hierarchical clustering: Average linkage, bisecting k-means, and local search. *J. Mach. Learn. Res.*, 24:1:1–1:36, 2023. URL http://jmlr.org/papers/v24/18-080.html.

Murtagh, F. A Survey of Recent Advances in Hierarchical Clustering Algorithms. *The Computer Journal*, 26(4):354–359, 11 1983. ISSN 0010-4620. doi: 10.1093/comjnl/26.4.354. URL https://doi.org/10.1093/comjnl/26.4.354.

Murtagh, F. and Contreras, P. Algorithms for hierarchical clustering: an overview. *WIREs Data Mining Knowl. Discov.*, 2(1):86–97, 2012. doi: 10.1002/WIDM.53. URL https://doi.org/10.1002/widm.53.

Ward, J. H. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 1963.

A. Proof of Proposition 2.1

In this section, we present the proof of Proposition 2.1 and then we argue that it implies that the rule employed by `complete-linkage` is equivalent to the rule that chooses at each iteration the two clusters A and B for which $\texttt{diam}(A \cup B)$ is minimum. This rule was analyzed in (Arutyunova et al., 2023).

Proof. The proof is by induction. For $j = 1$, A_j and A'_j are singletons so that $\texttt{diam}(A_1 \cup A'_1) = dist(x, y)$, where x and y are the only points in A_1 and A'_1, respectively.

We assume by induction that the result holds for every $i < j$. First we prove that $\texttt{diam}(A_j \cup A'_j) \geq \texttt{diam}(A_{j-1} \cup A'_{j-1})$. Note that

$$\texttt{diam}(A_j \cup A'_j) = \max\{\texttt{diam}(A_j), \texttt{diam}(A'_j), \max\{dist(x,y)|(x,y) \in A_j \times A'_j\}\} \tag{22}$$

If $A_j = A_{j-1} \cup A'_{j-1}$ or $A'_j = A_{j-1} \cup A'_{j-1}$ we conclude that $\texttt{diam}(A_j \cup A'_j) \geq \texttt{diam}(A_{j-1} \cup A'_{j-1})$. Otherwise, we have that

$$\texttt{diam}(A_j \cup A'_j) \geq \max\{dist(x,y)|(x,y) \in A_j \times A'_j\} \geq$$
$$\max\{dist(x,y)|(x,y) \in A_{j-1} \times A'_{j-1}\} = \texttt{diam}(A_{j-1} \cup A'_{j-1}),$$

where the second inequality follows from the `complete-linkage` rule and the last identity follows by induction.

It remains to show that

$$\texttt{diam}(A_j \cup A'_j) = \max\{dist(x,y)|(x,y) \in A_j \times A'_j\} \tag{23}$$

First, we consider the case where either $A_j = A_{j-1} \cup A'_{j-1}$ or $A'_j = A_{j-1} \cup A'_{j-1}$. We assume w.l.o.g. that $A_j = A_{j-1} \cup A'_{j-1}$. Thus, $\texttt{diam}(A_j) = \texttt{diam}(A_{j-1} \cup A'_{j-1}) \geq \texttt{diam}(A'_j)$, where the inequality holds because either A'_j is a singleton or it was obtained by merging two clusters before the iteration $j-1$ and, in this case, the induction hypothesis guarantees the inequality. Moreover, in this case, cluster A'_j is available to be merged right before the $(j-1)$th merge, so it follows from the `complete-linkage` rule that

$$\max\{dist(x,y)|(x,y) \in A'_j \times A_{j-1}\} \geq \max\{dist(x,y)|(x,y) \in A_{j-1} \times A'_{j-1}\} \tag{24}$$

and

$$\max\{dist(x,y)|(x,y) \in A'_j \times A'_{j-1}\} \geq \max\{dist(x,y)|(x,y) \in A_{j-1} \times A'_{j-1}\} \tag{25}$$

Thus,

$$\texttt{diam}(A_j \cup A'_j) \geq \tag{26}$$
$$\max\{dist(x,y)|(x,y) \in A_j \times A'_j\} = \tag{27}$$
$$\max\{dist(x,y)|(x,y) \in (A_{j-1} \cup A'_{j-1}) \times A'_j\} \geq \tag{28}$$
$$\max\{dist(x,y)|(x,y) \in A_{j-1} \times A'_{j-1}\} = \tag{29}$$
$$\texttt{diam}(A_{j-1} \cup A'_{j-1}) = \tag{30}$$
$$\texttt{diam}(A_j) \geq \texttt{diam}(A'_j), \tag{31}$$

where (28) follows from (24) and (25), while (29) follows by induction.

Therefore, (23) must hold, otherwise the inequalities (26)-(31) would contradict (22).

If neither $A_j = A_{j-1} \cup A'_{j-1}$ nor $A'_j = A_{j-1} \cup A'_{j-1}$ we have that

$$\texttt{diam}(A_j \cup A'_j) \geq \tag{32}$$
$$\max\{dist(x,y)|(x,y) \in A_j \times A'_j\} \geq \tag{33}$$
$$\max\{dist(x,y)|(x,y) \in A_{j-1} \times A'_{j-1}\} = \tag{34}$$
$$\texttt{diam}(A_{j-1} \cup A'_{j-1}) \geq \tag{35}$$
$$\max\{\texttt{diam}(A_j), \texttt{diam}(A'_j)\} \tag{36}$$

Again, (23) must hold, otherwise we contradict (22). □

Now we show that the rule employed by `complete-linkage` is equivalent to the rule that chooses at each iteration the two clusters A and B for which $\texttt{diam}(A \cup B)$ is minimum.

We assume for the sake of reaching a contradiction that at some iteration `complete-linkage` merges clusters A and B while there were clusters A' and B', with $\texttt{diam}(A' \cup B') < \texttt{diam}(A \cup B)$, that could be merged. In this case, we conclude that $\texttt{dist}_{CL}(A', B') \leq \texttt{diam}(A' \cup B') < \texttt{diam}(A \cup B) = \texttt{dist}_{CL}(A, B)$, where the last identity follows from Proposition 2.1. However, this contradicts the choice of `complete-linkage`.

B. Proof of Lemma 4.1

The following propositions are helpful to prove Lemma 4.1. The first one characterizes how $\texttt{pure}_t$ evolves when two clusters are merged.

Proposition B.1. *Let g and g' be the clusters merged at the iteration t of Algorithm 4. Then, exactly one of the following cases happen:*

1. *Both g and g' are non-pure. We have that $\texttt{pure}_t(H) = \texttt{pure}_{t-1}(H)$ for every family H in G.*
2. *g is a pure cluster w.r.t. family F and g' is non-pure. We have that $\texttt{pure}_t(F) = \texttt{pure}_{t-1}(F) - 1$ and $\texttt{pure}_t(H) = \texttt{pure}_{t-1}(H)$ for every family $H \neq F$.*
3. *g' is a pure cluster w.r.t. family F' and g is non-pure. We have that $\texttt{pure}_t(F') = \texttt{pure}_{t-1}(F') - 1$ and $\texttt{pure}_t(H) = \texttt{pure}_{t-1}(H)$ for every family $H \neq F'$.*
4. *g and g' are pure clusters w.r.t. families F and F', respectively.*

   Then, $\texttt{pure}_t(F) = \texttt{pure}_{t-1}(F) - 1$, $\texttt{pure}_t(F') = \texttt{pure}_{t-1}(F') - 1$ and $\texttt{pure}_t(H) = \texttt{pure}_{t-1}(H)$ for every family $H \notin \{F, F'\}$.

   Moreover, If $\texttt{pure}_{t-1}(F) \geq 2$ and $\texttt{pure}_{t-1}(F') \geq 2$ then $g \cup g'$ is not added to $\mathcal{E}$ by line 4 of Algorithm 4.

Proof. We argue for each of the cases of the statement:

1. This case holds because no pure cluster is affected when g and g' are merged.
2. In this case, g does not count for $\texttt{pure}_t(F)$, because g is merged, and $g \cup g'$ is not a pure cluster. Thus, $\texttt{pure}_t(F) = \texttt{pure}_{t-1}(F) - 1$.
3. The proof of this case is analogous to that of item 2.
4. If $F = F'$ then $g \cup g'$ is pure w.r.t. F. Thus, $\texttt{pure}_t(F) = \texttt{pure}_{t-1}(F) - 1$ because g and g' counts only for $\texttt{pure}_{t-1}(F)$ while $g \cup g'$ just count for $\texttt{pure}_t(F)$.

   If $F \neq F'$ then $g \cup g'$ is not pure. Since g counts for $\texttt{pure}_{t-1}(F)$ but not for $\texttt{pure}_t(F)$ we have $\texttt{pure}_t(F) = \texttt{pure}_{t-1}(F) - 1$. By using the same reasoning we conclude that $\texttt{pure}_t(F') = \texttt{pure}_{t-1}(F') - 1$.

   If $\texttt{pure}_{t-1}(F) \geq 2$ we cannot have $g \in \mathcal{E}$ because g is pure w.r.t. to F and no cluster in $\mathcal{E}$ is pure with respect to F. In fact, any cluster $h \in \mathcal{E}$ is added to $\mathcal{E}$ by either line 4 or line 4, or $h = h' \cup (h - h')$, where h' is a cluster that was added to $\mathcal{E}$ by either line 4 or line 4. Since h' is not pure w.r.t. F then h is not pure w.r.t. F. The same reasoning shows that if $\texttt{pure}_{t-1}(F') \geq 2$, then $g' \notin \mathcal{E}$. Therefore, $g \cup g'$ is not added to $\mathcal{E}$ by line 4.

□

Proposition B.2. *Every family is created by Algorithm 4 containing at least two clusters. Moreover, if F_C is created by case (b) of Algorithm 4 at iteration t, then at the beginning of this iteration exactly two families that belong to C have exactly two pure clusters and all the other families in C have at most one pure cluster.*

Proof. All the families created at line 4 have at least two pure clusters. We use induction on the number of iterations.

At the beginning of iteration 1 all connected components of G have only one family. Let g and g' be the two clusters merged when $t = 1$. If a family F_C is created (line 4) at this iteration, then the merging of g and g' must produce a connected component with two families. Thus, we conclude that g is pure with respect to a family F and g' is pure w.r.t. to a family F', with $F' \neq F$. Assume w.l.o.g. that $|F'| \geq |F|$. If (a) occurs we must have $|F| = 2$ and $|F'| > 2$ and if (b) occurs we must have $|F| = 2$ and $|F'| = 2$. Furthermore, If (a) occurs F_C will have at least two clusters, those in $F' \setminus \{g'\}$. If (b) occurs F_C will also have at least two clusters, $g \cup g'$ and the pure cluster in $(F \cup F') \setminus \{g, g'\}$ that is not added to $\mathcal{E}$ by line 4. Thus, the result holds at the first iteration.

Let $t > 1$. We assume by induction that the result holds for iteration $t - 1$. We analyze iteration t. We split the proof into two cases:

Case 1) F_C is created due to case (a) at iteration t.

The definition of the case (a) assures that there is a family F in the connected component C with $\texttt{pure}_t(F) \geq 2$. Thus, the pure clusters w.r.t. F are added to F_C, so that F_C is created with at least two clusters.

Case 2) F_C is created due to case (b) at iteration t.

The definition of case (b) assures that $|C| \geq 2$ and all families in C have at most one pure cluster after the merge of iteration t. Moreover, Proposition B.1 assures that at most two families have their number of pure clusters decreased when two clusters are merged. Thus, at least $|C| - 2$ families that lie in C have 0 or 1 pure cluster at the beginning of iteration t, otherwise, case (b) cannot occur.

We argue that exactly two families that lie in C have at least 2 pure clusters at the beginning of iteration t. For the sake of a contradiction, assume that either 0 or 1 family that lies in C has at least 2 pure clusters at the beginning of iteration t. We consider two scenarios:

- The component C does exist at the beginning of t, that is, it was not produced by the union of two components at iteration t. In this scenario, C would satisfy either case (a) or case (b) at iteration $t - 1$, and C would be removed from G. Thus, this scenario cannot occur.

- The component C does not exist at the beginning of t. In this scenario, C is the union of two connected components of G at the beginning of iteration t, that is, $C = C' \cup C''$. Let us assume $|C'| \geq |C''|$. If $|C'| \geq 2$, then $|C'|$ would satisfy either case (a) or case (b) at iteration $t - 1$ and, as a consequence, would be removed from G; this is not possible. If $|C'| = |C''| = 1$, then by our assumption one of these components, say C', has a family with at most one pure cluster at the beginning of t. Then by induction, C' does not correspond to a family created at iteration $t - 1$. Thus, C' satisfies case (c) at iteration $t - 1$, so that C' would be removed from G. Thus, this scenario cannot occur as well.

Let H and H' be the families in C that have at least 2 pure clusters at the beginning of iteration t. If one of them, say H, has more than 2 pure clusters, then C does not satisfy case (b) because we would have $\texttt{pure}_t(H) \geq 2$. Thus, $\texttt{pure}_{t-1}(H) = \texttt{pure}_{t-1}(H') = 2$ and $\texttt{pure}_{t-1}(H'') < 2$ for every family H'' in $C \setminus \{H, H'\}$.

It remains to show that F_C is created with at least two clusters. Since (b) occurs, we must have that g is pure w.r.t H and g' is pure w.r.t. H'. Moreover, item 4 of Proposition B.1 guarantees that $g \cup g' \notin \mathcal{E}$. Thus, $g \cup g'$ is added to F_C. Moreover, either H or H' will have a pure cluster that is not added to $\mathcal{E}$ by line 4 and, thus, this cluster will be added to F_C. $\square$

Proposition B.3. *Let C be the connected component associated with family F_C. If F_C is created by case (a) of Algorithm 4, then $|C| - 1$ families in C have a pure cluster added to $\mathcal{E}$ by line 4. If F_C is created by case (b) of Algorithm 4, then $|C| - 2$ families in C have a pure cluster added to $\mathcal{E}$ by line 4 and one family has a pure cluster added to $\mathcal{E}$ by line 4. In both cases, C has exactly one family that adds no cluster to $\mathcal{E}$.*

Proof. By Proposition B.2, every family is created with at least two pure clusters. By Proposition B.1, at each iteration the number of pure clusters in a family either remains the same or decreases by one unit.

If F_C is created due to case (a), then exactly $|C| - 1$ families in C have at most one pure cluster. Thus, all of them reach line 4 and the result holds.

If F_C is created due to the case (b), then Proposition B.2 guarantees that at the beginning of the iteration in which F_C is created, C has exactly two families, say H and H', with exactly two pure clusters and all others with at most one pure cluster. Therefore, every family in $C - \{H, H'\}$ reaches line 4 and, thus, add a cluster to $\mathcal{E}$. Furthermore, exactly one family in $\{H, H'\}$ adds a cluster to $\mathcal{E}$ in line 4. $\square$

Proposition B.4. *The total number of clusters added to $\mathcal{E}$ by lines 4 and 4 of Algorithm 4 is at most k.*

Proof. Let $m_{>1} = |\{T_i^* | T_i^*$ has at least 2 points$\}|$ and $m_1 = |\{T_i^* | T_i^*$ has exactly 1 point$\}|$. Initially, m_1 clusters are added to $\mathcal{E}$ (line 4).

Let $D_1, D_2, \dots, D_p$ be the trees of the forest D at the end of the Algorithm 4. Moreover, let $int(D_i)$ and $leaves(D_i)$ be, respectively, the set of internal nodes and leaves of D_i. Fix an internal node v in D_i. Note that v corresponds to a family F_C for some connected component C of G that has at least two families (line 4). Each child of v corresponds to a family in C and by Proposition B.3 all of them but one add a cluster to $\mathcal{E}$. Hence, we can associate to v, $|children(v)| - 1$ clusters that are added to $\mathcal{E}$. Thus, the number of clusters added to $\mathcal{E}$ is at most

$$p + \sum_{i=1}^{p} \left(\sum_{v \in int(D_i)} (|children(v)| - 1) \right) = p + \sum_{i=1}^{p} \left(-|int(D_i)| + \sum_{v \in int(D_i)} |children(v)| \right) =$$

$$p + \sum_{i=1}^{p} (-|int(D_i)| + |D_i| - 1) = \sum_{i=1}^{p} leaves(D_i) = |leaves(D)| = m_{>1},$$

where the term p is due to the roots of the trees $D_1, \dots, D_p$ since they can also add pure clusters to $\mathcal{E}$.

Therefore, the total number of clusters in $\mathcal{E}$ never exceeds $m_{>1} + m_1 = k$. $\square$

The next proposition characterizes the clusters created by `complete-linkage`.

Proposition B.5. *At the beginning of iteration t of Algorithm 4, each cluster $h \in \mathcal{C}^{t-1}$ satisfies exactly one of the following possibilities:*

(i) $h \in \mathcal{E}$;

(ii) $h \notin \mathcal{E}$ *and* h *is pure w.r.t a family in G;*

(iii) $h \notin \mathcal{E}$, h *is non-pure and there is a component* C *in* G *such that* $h \subseteq \bigcup_{H \in C} \texttt{Pts}(H)$.

Proof. At the beginning of Algorithm 4, each cluster h satisfies (i) or (ii). We assume by induction that this property holds at the beginning of iteration t and prove that it also holds at the beginning of iteration $t + 1$.

We first argue that right after the merge of iteration t every cluster in $\mathcal{C}^t$ satisfies one of the desired conditions and, then, we argue that these clusters still satisfy the desired conditions by the end of iteration t or, equivalently, at the beginning of iteration $t + 1$.

Let h be a cluster in $\mathcal{C}^t$. If h also belongs to to $\mathcal{C}^{t-1}$ then, by induction, h satisfies one of the conditions at the beginning of iteration t. If h satisfies (i) (resp. (ii)), then it also satisfies (i) (resp. (ii)) right after the merge because the assumption that $h \in \mathcal{C}^t$ guarantees that h is not merged at iteration t. If h satisfies (iii), then $h \subseteq \bigcup_{H \in C} \texttt{Pts}(H)$, for some connected component of G. Hence, h satisfies the condition $h \subseteq \bigcup_{H \in C^{new}} \texttt{Pts}(H)$, after the merge, where C^{new} is the component in G that contains the families of C after the merge.

If h does not belong to $\mathcal{C}^{t-1}$, then $h = g \cup g'$, where g and g' are the clusters merged at iteration t. If $g \in \mathcal{E}$ or $g' \in \mathcal{E}$ then $h \in \mathcal{E}$, so it satisfies (i) after the merge. If neither $g \in \mathcal{E}$ nor $g' \in \mathcal{E}$ then h is not added to $\mathcal{E}$. Moreover, if both g and g' are pure with respect to the same family H, then h is also pure w.r.t. H, so it satisfies (ii). Otherwise, h is not pure and it satisfies (iii) when it is created.

We have just proved that every cluster in $\mathcal{C}^t$ satisfies one of the conditions of the proposition right after the merge. Now we show that each cluster in $\mathcal{C}^t$ satisfies one of the conditions at the end of the iteration t.

Let $h \in \mathcal{C}^t$. We have the following cases:

- h satisfies (i) after the merge. Then, it will satisfy (i) at the beginning of iteration $t+1$.
- h satisfies (ii) after the merge. Then, h is pure w.r.t. some family H. Let C be the component where the family H lies. If C meets the conditions of cases (a) or (b) then h will satisfy (ii) at the beginning of iteration $t+1$. because h will be pure with respect to the new family F_C. If C meets the conditions of case (c) then h is added to $\mathcal{E}$, so it satisfies (i) at the beginning of iteration $t+1$. If C does not meet any of the cases, then h will satisfy either (i) or (ii) at the beginning of iteration $t+1$.
- h satisfies (iii) after the merge. Let C be a component of G, right after the merge, such that $h \subseteq \bigcup_{H \in C} \texttt{Pts}(H)$. Note the $|C| \geq 2$ and, thus C cannot meet case (c). If C meets either case (a) or (b) then h will satisfy (ii) at the beginning of iteration $t+1$ because h will be pure with respect to the new family F_C. If C does not meet (a) or (b), then h will satisfy (iii) at the beginning of iteration $t+1$.

□

Now we state and prove two propositions that, together, directly imply the correctness of Lemma 4.1.

Proposition B.6. *Every connected component C in G satisfies one of the following conditions: (i) $|C| = 1$ and the only family of C has at least two pure clusters or (ii) $|C| > 1$ and there exist two families in C such that each of them has at least two pure clusters.*

Proof. For the sake of reaching a contradiction, let t be the first iteration for which there is a component C in G that does not satisfy the conditions of the lemma at the beginning of iteration t. Note that $t \geq 2$.

If $|C| = 1$ and the only family F in C does not have at least 2 pure clusters, then C cannot be the component associated with the family F_C created by either case (a) or (b) at iteration $t-1$ because Proposition B.2 guarantees that the component where F_C lies satisfies condition (i). Thus, C satisfies the condition of case (c) at iteration $t-1$ and then it is removed from G at iteration $t-1$, which contradicts its existence at the beginning of iteration t

If both $|C| \geq 2$ and C has only one family with at least 2 pure clusters, then C would satisfy case (a) at iteration $t-1$ and it would be removed from G, which contradicts its existence at the beginning of iteration t. Similarly, if $|C| \geq 2$ and it has no family with at least 2 pure clusters then C would satisfy case (b) at iteration $t-1$ and it would be removed from G, which again contradicts its existence at the beginning of iteration t

We have established the proposition. □

Proposition B.7. *If all the nodes/families of G are removed at iteration t' then $t' = n - k$*

Proof. We first argue that at the beginning of iteration t', G has exactly one connected component. For the sake of reaching a contradiction, let us assume that G has two components say C and C'. By Proposition B.6, C has one family, say F, with $\texttt{pure}_{t'-1}(F) \geq 2$. Let g and g' be the clusters merged at iteration t'. If neither g not g' is pure with respect to F we will have $\texttt{pure}_{t'}(F) \geq 2$ and the component where F lies after the merge will not satisfy the condition of case (c). Hence, there still be nodes in G by the end of t'. Thus, one of the clusters, say g, is pure w.r.t. F. By using the same reasoning we conclude that C' has at least one family, say F', with $\texttt{pure}_{t'-1}(F') \geq 2$ and g' is pure w.r.t. F'. Since $g \cup g' \notin \mathcal{E}$ (item 4 of Proposition B.1) then $g \notin \mathcal{E}$ and $g' \notin \mathcal{E}$, so that the merge of g and g' will create a component $C \cup C'$ in G that has at least two families; this component does not not satisfy the conditions of case (c), so G will have nodes by the end of iteration t'

We have proved that if G has all its nodes removed at iteration t', then there is only one component in G at the beginning of iteration t'. Let C be this component: C must have exactly one family, say F, and F must have at most two pure clusters, otherwise case (c) is not reached and G will still have nodes by the end of iteration t'.

Since $|C| = 1$ no cluster satisfies condition (iii) of Proposition B.5 and, thus, the total of clusters at the beginning of iteration t' is given by the number of clusters that satisfy either condition (i) or (ii) of Proposition B.5.

Proposition B.4 guarantees that the number of clusters $h \in \mathcal{E}$ is at most k. Thus, if F has less than two pure clusters at the beginning of t', then the number of clusters h that satisfies (i) or (ii) of Proposition B.5. is at most $k+1$.

If F has two pure clusters at the beginning of t' and C satisfies case (c) at iteration t', then a pure cluster in F is added to $\mathcal{E}$. Thus, Proposition B.4 guarantees that $\mathcal{E}$ has at most $k-1$ clusters at the beginning of iteration t'. Hence, the number of clusters that that satisfies (i) or (ii) of Proposition B.5 is, again, at most $(k-1)+2 = k+1$.

Thus, the total number at the beginning of iteration t' is at most $k+1$ and, hence, $t' = n-k$. □

Proof of Lemma 4.1. It follows directly from Propositions B.6 and B.7.

C. Proof of Proposition 4.2

Proof. Let F be a family that has at least two pure clusters right before the merge m_i and let h and h' be two pure clusters w.r.t. F. We first note that

$$\mathtt{diam}(g_i) \leq \max\{dist(x,y)|(x,y) \in h \times h'\} \leq \mathtt{diam}(h \cup h') \leq \mathtt{diam}(F),$$

where the first inequality holds due to the choice of `Complete-Link` and Proposition 2.1. Moreover,

$$\mathtt{diam}(g_i) \leq \mathtt{diam}(F')$$

for any family in C that does not exist before the merge. In fact, by condition (i) of Lemma 4.1 F' is created with at least two clusters, say h and h', and $\mathtt{diam}(g_i) \leq \mathtt{diam}(h \cup h') \leq \mathtt{diam}(F')$, where the first inequality follows from Proposition 2.1. Hence, we can conclude that $\mathtt{diam}(g_1) \leq \mathtt{DM}_1$ because before the first merging each family in C that already exists is an isolated node in G and has at least two pure clusters (condition (i) of Lemma 4.1).

Now, we consider the case $i > 1$. Let a be the number of families in C that have at least two pure clusters right before the merge m_i and let b be the number of families in C that have not been created yet. It is enough to show that $a+b \geq |C|-i+2$ (claim below). In fact, in this case $|C|-i+2$ families in C have diameter not smaller than $\mathtt{diam}(g_i)$ so that $\mathtt{diam}(g_i) \leq \mathtt{DM}_{i-1}$

Claim. $a+b \geq |C|-i+2$

Proof. Right before m_i, the families in C are distributed in $(|C|-b)-i+1$ connected components in the graph G. If one of these components has just one family, it follows from condition (i) of Lemma 4.1 that this family must have at least two pure clusters. If one of these components has at least two families, then it follows from condition (ii) of Lemma 4.1 that there are two families in this component, each of them with at least two pure clusters.

Since $i > 1$, at least one component has at least two families. Thus, there are at least $(|C|-b)-i+2$ families with at least two pure clusters right before m_i. We conclude that $a \geq (|C|-b)-i+2$ and, hence, $a+b \geq |C|-i+2$

End of Proof. □

D. Proof of Proposition 4.3

Proof. For a given point x, we use F_x to denote the family in connected component C where x lies right before the families in C are replaced with F_C. Let a and b be the two farthest points of F_C. We split the proof into two cases:

Case i) The path from the family F_a to F_b in T_C has less than $|C|-1$ edges.

Let $u_1, \ldots, u_t$, with $u_1 = F_a$ and $u_t = F_b$, be such a path. Note that $t < |C|$. Recall that in the construction of T_C, an edge between families u_i and u_{u+1} is associated with some cluster g. Let p'_i and p_{i+1} be, respectively, arbitrarily chosen points in $\mathtt{Pts}(u_i)$ and $\mathtt{Pts}(u_{i+1})$ that belong to g. Now, consider the sequence of points $(a = p_1, p'_1, p_2, p'_2, \ldots, p_t, p'_t = b)$. We have that

$$\begin{aligned} \mathtt{diam}(F_C) = dist(a,b) \leq \\ \sum_{i=1}^{t} dist(p_i, p'_i) + \sum_{i=1}^{t-1} dist(p'_i, p_{i+1}) \leq \\ \sum_{i=2}^{|C|} \mathtt{DM}_i + \sum_{i=1}^{|C|-2} \mathtt{DM}_i, \end{aligned}$$

where the first inequality holds due to the triangle inequality and for the second one we use the fact that $\sum_{i=1}^{t} dist(p_i, p'_i)$ can be upper bounded by the $|C|-1$ largest diameters of the families in C and Proposition 4.2 assures that $\sum_{i=1}^{t-1} dist(p'_i, p_{i+1})$ can be upper bounded by the sum of the weights of the $|C|-2$ most expensive edges of T_C.

Case ii) The path from F_a to F_b in T_C has $|C|-1$ edges.

Since $|C|>1$ we have that $F_a \neq F_b$. It follows from Proposition B.3 that there is $y \in \{a,b\}$ such that a pure cluster w.r.t. family F_y is added to $\mathcal{E}$, before the creation of F_C, by either line 4 or line 4.

We assume w.l.o.g. that $y=a$. Let g be the pure cluster w.r.t. F_a that is added to $\mathcal{E}$. We assume that F_C is created at iteration t and the addition of g to $\mathcal{E}$ happened at iteration t', so that $t' \leq t$. We cannot have $a \in g$ because points that belong to clusters in $\mathcal{E}$ are not in $\texttt{Pts}(F_C)$. Moreover, a cannot be in a pure cluster w.r.t. F_a after the t'-th merge, otherwise we would have $\texttt{pure}_{t'}(F_a) \geq 2$ and g would not have been added to $\mathcal{E}$. Thus, right after the t'-th merge, a belongs to a cluster that contains a point, say x, from a family F_x different from F_a.

We must have

$$dist(a,x) \leq \texttt{diam}(F_a) \tag{37}$$

since the cluster that contains a and x was created when F_a still had at least two pure clusters.

Now consider the path $(F_x = v_1, \ldots, v_t = F_b)$ from F_x to F_b in T_C. This path does not include F_a, otherwise the path from F_a to F_b would have at most $|C|-2$ edges, which is not possible since we are in case (ii). If the edge in T_C that connects families v_i to v_{u+1} corresponds to cluster g then choose p'_i and p_{i+1} as points in v_i and v_{i+1}, respectively, that belong to g. Now, consider a sequence of points $(a, p_1, p'_1, p_2, p'_2, \ldots, p_t, p'_t)$, where $p_1 = x$ and $p'_t = b$. From the triangle inequality,

$$dist(a,b) \leq dist(a,x) + \sum_{i=1}^{t} dist(p_i, p'_i) + \sum_{i=1}^{t-1} dist(p'_i, p_{i+1}).$$

Moreover, we have

$$\sum_{i=1}^{t} dist(p_i, p'_i) \leq \sum_{i=1}^{|C|} \texttt{DM}_i - \texttt{diam}(F_a)$$

and due to Proposition 4.2

$$\sum_{i=1}^{t-1} dist(p'_i, p_{i+1}) \leq \sum_{i=2}^{|C|-1} \texttt{DM}_{i-1},$$

Hence,

$$\begin{aligned} dist(a,b) \leq \\ dist(a,x) - \texttt{diam}(F_a) + \sum_{i=1}^{|C|} \texttt{DM}_i + \sum_{i=2}^{|C|-1} \texttt{DM}_{i-1} \leq \\ \sum_{i=1}^{|C|} \texttt{DM}_i + \sum_{i=2}^{|C|-1} \texttt{DM}_{i-1} = \sum_{i=1}^{|C|} \texttt{DM}_i + \sum_{i=1}^{|C|-2} \texttt{DM}_i, \end{aligned}$$

where the last inequality follows from (37). □

E. Proof of Theorem 4.5

Proof. Due to the definition of α_k, it is enough to show that the diameter of every cluster created by `complete-linkage` is at most $\text{OPT}_{\texttt{DM}}(k) \cdot k^{\alpha_k}$.

We prove it by induction on the number of iterations of the Algorithm 4. At the beginning, all n clusters have a diameter of 0, so the result holds.

We assume by induction that the result holds at the beginning of iteration t. At the beginning of this iteration, by Lemma 4.1 there is a family, say F, with at least 2 pure clusters. Let h and h' be these clusters. Moreover, let g and g' be the clusters merged at iteration t. We have that

$$\begin{aligned}
\texttt{diam}(g \cup g') = \\
\max\{\texttt{diam}(g), \texttt{diam}(g'), \texttt{dist}_{CL}(g, g')\} \leq \\
\max\{\texttt{diam}(g), \texttt{diam}(g'), \texttt{dist}_{CL}(h, h')\} \leq \\
\max\{\texttt{diam}(g), \texttt{diam}(g'), \texttt{diam}(h \cup h')\} \leq \\
\max\{\texttt{diam}(g), \texttt{diam}(g'), \texttt{diam}(F)\} \leq \\
\text{OPT}_{\texttt{DM}}(k) k^{\alpha_k}
\end{aligned}$$

where the first inequality is due to the choice of `complete-linkage` and the last inequality holds due to Lemma 4.4, the inductive hypothesis and the fact that $\phi(k) \leq k$. □

F. Proof of Theorem 5.2

Proof. We prove by induction on the number of iterations of $\texttt{Link}_f$ (in parallel on $\texttt{Algo}_f$) that each cluster A created by $\texttt{Link}_f$ satisfies $\texttt{cost}(A) \leq k^{\log_2 3} \text{OPT}_{\texttt{AV}}(k)$. At the beginning, this holds because every cluster A is a point, so that $\texttt{cost}(A) = 0$ due to condition (ii). We assume by induction that the desired property holds at the beginning of iteration t.

Let g and g' be two clusters merged at iteration t. By Proposition 3.1 there is a regular family F at the beginning of the t-th iteration. Let h and h' be two clusters in F. Therefore,

$$\begin{aligned}
\texttt{cost}(g \cup g') \leq & \quad (38) \\
\max\{\texttt{cost}(g), \texttt{cost}(g'), f(g, g')\} \leq & \quad (39) \\
\max\{\texttt{cost}(g), \texttt{cost}(g'), f(h, h')\} \leq & \quad (40) \\
\max\{\texttt{cost}(g), \texttt{cost}(g'), \texttt{diam}(h \cup h')\} \leq & \quad (41) \\
\max\{\texttt{cost}(g), \texttt{cost}(g'), \texttt{diam}(F)\} \leq & \quad (42) \\
k^{1.59} \text{OPT}_{\texttt{AV}}(k) & \quad (43)
\end{aligned}$$

where the first inequality holds due to condition (iii), the second due to the choice of $\texttt{Link}_f$, the third due to condition (i) and the last one follows from induction and Proposition 5.1. □

G. Useful inequalities

Proposition G.1. *Let $p = \log_2 3 - 1$ and let a, b, x, y real numbers with $0 \leq a, b$ and $x, y \geq 1$. Moreover, let $ax^p \geq by^p$. Then,*

$$ax^p + 2by^p \leq (a + b)(x + y)^p$$

Proof. Let

$$f(a, b, x, y) = (a + b)(x + y)^p - ax^p - 2by^p.$$

We have that

$$\frac{\partial f}{\partial a} = (x + y)^p - x^p > 0.$$

Since $ax^p \geq by^p$, in the minimum of f, we must have $a = b(y/x)^p$. When $a = b(y/x)^p$, we have that

$$f(a, b, x, y) = b\left(\left(\frac{y}{x}\right)^p + 1)(x + y)^p - 3by^p\right) = b\left((y^p + x^p)(x + y)^p - 3x^p y^p\right)$$

Since $b \geq 0$ it is enough to prove that $(y^p + x^p)(x + y)^p - 3x^p y^p \geq 0$

By the AGM inequality

$$x^p + y^p \geq 2(x^{p/2})(y^{p/2})$$

and

$$(x+y)^p \geq (2(xy)^{1/2})^p$$

By multiplying these inequalities we get

$$(x^p+y^p)(x+y)^p \geq 2^p 2(x^p y^p) = 3x^p y^p$$

□

Proposition G.2. *The following holds*

$$\frac{\log 6}{\log 4} = \max\left\{\frac{\log(2i-2)}{\log i} | i \textit{ is an integer larger than } 1\right\}$$

Proof. We can inspect manually that $\log(2i-2)/\log i \leq \frac{\log 6}{\log 4}$, for every integer smaller than 11. For $i \geq 11$ we have that

$$\frac{\log(2i-2)}{\log i} < \frac{\log(2i)}{\log i} = 1 + \frac{1}{\log i} \leq 1 + \frac{1}{\log 11} \leq \frac{\log 6}{\log 4}$$

. □

Proposition G.3. *Let p be a real number that satisfies $p \geq \log_i(2i-2)$, for every $i > 1$. Moreover, let ℓ be a positive number larger than 1 and let $1 \leq a_1 \leq a_2 \ldots \leq a_\ell$. Then,*

$$a_\ell^p + a_{\ell-1}^p + \sum_{i=1}^{\ell-2} 2a_i^p \leq \left(\sum_{i=1}^{\ell} a_i\right)^p$$

Proof. Let $\mathbf{a} = (a_1, \ldots, a_\ell)$. We define

$$f(\mathbf{a}) := \left(\sum_{i=1}^{\ell} a_i\right)^p - \left(a_\ell^p + a_{\ell-1}^p + \sum_{i=1}^{\ell-2} 2a_i^p\right).$$

We need to show that $f(\mathbf{a}) \geq 0$, for every valid $\mathbf{a}$.

First, we consider the case $\ell = 2$. In this case,

$$\frac{\partial f}{\partial a_2} = p(a_1+a_2)^{p-1} - pa_2^{p-1} > 0$$

Thus, in the minimum of f we must have $a_1 = a_2$. When this happens,

$$f(\mathbf{a}) = (2a_1)^p - 2a_1^p > 0.$$

Now, we consider the case $\ell > 2$. For $t \leq \ell - 2$, we define

$$f^t(\mathbf{a}) := \left(\sum_{i=1}^{t-1} a_i + (\ell - t + 1)a_t\right)^p - \left(2(\ell-t)a_t^p + \sum_{i=1}^{t-1} 2a_i^p\right)$$

Note that $f^t(\mathbf{a}) = f(\mathbf{a}')$ where $a'_i = a_i$ for $i < t$ and $a'_i = a_t$ for $i \geq t$. We show that the minimum of f is equal to the minimum of f^t.

For $j > \ell - 2$,

$$\frac{\partial f}{\partial a_j} = p(\sum_{i=1}^{\ell} a_i)^{p-1} - pa_j^{p-1} > 0.$$

Hence, in the minimum of f we have that $a_{\ell-2} = a_{\ell-1} = a_\ell$. Thus, the minimum of f equals the minimum of $f^{\ell-2}$. Now, we show that the minimum of f^t is equal to the minimum of f^{t-1} for $t \leq \ell - 2$.

We have that

$$\frac{\partial f^t}{\partial a_t} = (\ell - t + 1)p \left(\sum_{i=1}^{t-1} a_i + (\ell - t + 1)a_t \right)^{p-1} - 2(\ell - t)pa_t^{p-1} > p(\ell - t + 1)^p a_t^{p-1} - 2(\ell - t)pa_t^{p-1} \geq 0,$$

where the second inequality holds because the definition of p assures that $p \geq \frac{\log 2(\ell - t)}{\log(\ell - t + 1)}$. Hence, in the minimum of f^t we have that $a_t = a_{t-1}$, so that the minimum of f^t equal the minimum of f^{t-1}. Therefore, we can conclude that the minimum of f is equal to the minimum of f^1.

Now we note that

$$f^1(\mathbf{a}) = (\ell a_1)^p - (2\ell - 2)(a_1)^p \geq (2\ell - 2)(a_1)^p - (2\ell - 2)(a_1)^p = 0,$$

where the inequality holds because $p \geq \log_\ell(2\ell - 2)$ □