Approximation Scheme for Weighted Metric Clustering via Sherali-Adams

Dmitrii Avdiukhin[1], **Vaggos Chatziafratis**[2], **Konstantin Makarychev**[1], **Grigory Yaroslavtsev**[3]

[1]Northwestern University, Illinois
[2]University of California at Santa Cruz, California
[3]George Mason University, Virginia
dmitrii.avdiukhin@northwestern.edu, vaggos@ucsc.edu, konstantin@northwestern.edu, grigory@gmu.edu

Abstract

Motivated by applications to classification problems on metric data, we study *Weighted Metric Clustering* problem: given a metric d over n points, the goal is to find a k-partition of these points into clusters $C_1, \dots, C_k$, while *minimizing* $\sum_{i=1}^{k} \sum_{j=1}^{k} \sum_{u \in C_i} \sum_{v \in C_j} A_{ij}\, d_{uv}$, where $\mathbf{A}$ is a $k \times k$ symmetric matrix with non-negative entries. Specific choices of A lead to Weighted Metric Clustering capturing well-studied graph partitioning problems in metric spaces, such as Min-Uncut, Min-k-Sum, Min-k-Cut, and more.

Our main result is that Weighted Metric Clustering admits a polynomial-time approximation scheme (PTAS). Our algorithm handles all the above problems using the Sherali-Adams linear programming relaxation. This subsumes several prior works, unifies many of the techniques for various metric clustering objectives, and yields a PTAS for several new problems, including metric clustering on manifolds and a new family of hierarchical clustering objectives. Our experiments on the hierarchical clustering objective show that it better captures the ground-truth structural information compared to the popular Dasgupta's objective.

1 Introduction

We introduce and study Weighted Metric Clustering problem: given n points from an arbitrary metric space (V, d), we want to find a k-partition of V, i.e. a partition into k clusters $C_1, \dots, C_k$, where k is assumed to be a fixed constant. Because the quality of clustering may depend on the application at hand, we allow for a user-defined $k \times k$ symmetric matrix $\mathbf{A}$ with non-negative entries to be part of the input. Matrix $\mathbf{A}$ determines the "cost penalty" for how the k different clusters interact: if u is assigned to cluster C_i and v is assigned to cluster C_j, then the pair (u, v) pays $A_{ij} d_{uv}$, where the distance between elements u, v is denoted as d_{uv}. Hence, our goal is to *minimize* the following objective:

$$\textsc{cost}(C_1, \dots, C_k) = \sum_{i=1}^{k} \sum_{j=1}^{k} \sum_{u \in C_i} \sum_{v \in C_j} A_{ij} d_{uv}. \qquad (\star)$$

In Weighted Metric Clustering, n is the number of input variables and k is assumed to be a fixed constant independent of n. Observe that $\sum_{u \in C_i} \sum_{v \in C_j} d_{uv}$ can be thought of as an overall measure of dissimilarity between clusters C_i and C_j, which is weighted with A_{ij} in the objective $(\star)$.

Note that we can interpret our objective $(\star)$ as a *minimization* valued Constraint Satisfaction Problem (MIN-CSP) on variables in V and domain $\mathcal{D} = \{1, \dots, k\}$. In this CSP, we have a constraint for all pairs of variables. The weight of the constraint between variables u and v equals the distance d_{uv}. The payoff function for each constraint is defined by matrix $\mathbf{A}$: namely, the cost of assigning labels i and j to variables u and v equals A_{ij}. The goal is to find an assignment, i.e. a mapping $\ell \colon V \to \mathcal{D}$ minimizing the total payoff: $\sum_{i=1}^{k} \sum_{j=1}^{k} \sum_{u \in V} \sum_{v \in V} A_{ij} d_{uv} \cdot \mathbb{1}\{\ell(u) = i; \ell(v) = j\}$.

The strength of objective $(\star)$ lies in the flexibility of choice of matrix $\mathbf{A}$, allowing it to cover many important problems.

Metric Min-Uncut (Indyk 1999) This is the complement of Max-Cut where we want to split into two clusters so as to minimize the sum of pairwise distances within clusters. If in $(\star)$ we set $k = 2$, $\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, then we pay d_{uv} only for elements u, v that end up in the same cluster.

Metric Min-k-Sum (Bartal, Charikar, and Raz 2001) Also termed Min-k-Uncut, this is the natural extension of the previous problem to k clusters, where we want to minimize the sum of distances between pairs of points assigned to the *same* cluster. Fixing $\mathbf{A} = I_{k \times k}$ to be the $k \times k$ identity matrix yields the problem.

Metric Multiway Cut (Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis 1994) We can also model problems where the cost is based on the *separated* u, v pairs. For example, taking $\mathbf{A} = J_{k \times k} - I_{k \times k}$, where J is the all-ones matrix, yields the Min-k-Cut objective, with the goal of minimizing the sum of distances among all pairs of separated points. Min-k-Cut problem additionally requires that all clusters are non-empty, and one possible approach is to fix one point per cluster; this variant of the problem, known as a *multiway cut*, is MAX SNP-hard (Dahlhaus et al. 1994) even for $k = 3$. Our algorithms are robust to such modifications of the objective and provide a PTAS for the metric case for fixed k.

A related problem is a *multicut* problem (see e.g. Costa, Létocart, and Roupin (2005)), where, given a set of k pairs

$\{(s_i, t_i)\}_{i=1}^{k}$, we need to remove the edges with the smallest possible weight so that s_i and t_i are disconnected for all i. For fixed k, similarly to the multiway cut problem, we can guess clusters for all s_i and t_i.

Metric Clustering on Manifolds Our formulation can also capture problems where data points reside on a manifold. In this case, the clusters are related (they can form a chain, a ring, or a grid) and we would like to find a clustering by grouping adjacent data points. As an example, the chain topology on four clusters, i.e. $C_1 - C_2 - C_3 - C_4$, can be represent by matrix

$$\mathbf{A} = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{pmatrix},$$

indicating that pairs of points in the same cluster pay 2, pairs in the neighboring clusters pay 1, and pairs in non-neighboring clusters pay 0. Understanding such problems on manifolds served as motivation for the original work by Song, Smola, Gretton, and Borgwardt (2007) that introduced the maximization variant of a special case of ($\star$) called Kernel Clustering. For such problems, to the best of our knowledge, no approximation was known for the minimization versions and our results provide the first PTAS.

New Application to Metric Hierarchical Clustering To highlight the versatility of objective ($\star$), we present an application to hierarchical clustering motivated by graph compression and graph reordering problems in social networks (Dhulipala, Kabiljo, Karrer, Ottaviano, Pupyrev, and Shalita 2016). We introduce a novel family of minimization objectives over hierarchies which depend on the *depth* of the Lowest Common Ancestors (LCA) for pairs of leaves. In contrast, almost all prior works considered hierarchical clustering objectives based on the *size* of the LCA (Dasgupta 2016).

2 Previous Work and Our Results

While there were important works on obtaining PTAS's for minimization problems (Indyk 1999; de la Vega, Karpinski, and Kenyon 2004; de la Vega, Karpinski, Kenyon, and Rabani 2003), it was not a priori clear whether a PTAS for these problems could exist. This is mainly due to pessimistic hardness results that hold for related minimization problems: for example, for every $k > 2$ and $\varepsilon > 0$, the Min-k-Sum problem cannot be approximated within $n^{2-\varepsilon}$, even for dense graphs (Kann, Khanna, Lagergren, and Panconesi 1996). For more background on maximization and MIN-CSPs (Appendix A).

Surprisingly, we show that every problem within our Weighted Metric Clustering ($\star$) framework admits a PTAS. As a consequence, this gives alternative PTAS for various problems, e.g., it subsumes known PTAS results for Metric Min-Uncut (Indyk 1999) and Metric Min-k-Sum (Bartal, Charikar, and Raz 2001). Furthermore, we give new PTAS's for various other problems, since any matrix $\mathbf{A}$ gives rise to a new clustering problem. In particular, our framework gives the first PTAS for metric minimization version of clustering on manifolds mentioned above (Song et al. 2007), multiway cut (Dahlhaus et al. 1994), and multicut (Costa, Létocart, and Roupin 2005) problems. Furthermore, we give PTAS for a new family of hierarchical clustering objectives motivated by graph compression and graph relabeling.

An interesting aspect of our result is that a *single* algorithmic technique based on the Sherali–Adams LP relaxation can accommodate all problems. Notice that just Min-k-Sum required a variety of tools (and often ad hoc ideas) to get a PTAS: for example, the PTAS of Indyk (1999) for $k = 2$ relied on the already known PTAS for metric Max-Cut, the first non-trivial approximation of Min-k-Sum (for general k) relied on metric embeddings into hierarchically separated trees combined with dynamic programming, and finally, the PTAS of de la Vega, Karpinski, Kenyon, and Rabani (2003) used sampling and exhaustive search combined with careful reassignment of nodes to the k clusters. Our main result can be seen as a unified method that provides PTAS not only for Min-k-Sum, but all other metric problems in our framework.

Sherali–Adams. The Sherali-Adams lift-and-project method (Sherali and Adams 1990) is a powerful technique for strengthening linear programming relaxations. This as well as other lift-and-project methods (e.g., by Lovász and Schrijver (1991)) have been extensively studied in Computer Science and Operations Research.[1] They asked if Sherali-Adams can be used to improve approximation guarantees for constraint satisfaction and combinatorial optimization problems. It turns out, that in many cases, the answer to this question is negative. Yannakakis (1988) proved the Traveling Salesman Problem (TSP) cannot be solved exactly using a symmetric "extended formulation" of polynomial size and, in particular, by a Sherali-Adams relaxation of polynomial size. De la Vega and Kenyon-Mathieu (2007) and Charikar, Makarychev, and Makarychev (2009a) showed that Sherali-Adams relaxation can not be used to improve approximation guarantees for many constraint satisfaction problems if we do not make additional assumptions about the structure of the CSP instances (see also Alekhnovich, Arora, and Tourlakis (2011)).

However, in some cases, Sherali-Adams can be used to obtain better approximations for MAX-CSPs. In particular, Yoshida and Zhou (2014) gave a PTAS for dense instances of MAX-CSPs (but not MIN-CSPs!). For additional examples of MAX-CSP approximations using Sherali-Adams, we refer the reader to recent papers by Thapper and Zivny (2017); Hopkins, Schramm, and Trevisan (2020); Romero, Wrochna, and Živnỳ (2021); Cohen-Addad, Lee, and Newman (2022); Mezei, Wrochna, and Živnỳ (2023).

Kernel Clustering Motivated by applications in machine learning and statistics, Kernel Clustering was proposed by Song, Smola, Gretton, and Borgwardt (2007) as a broad family of clustering methods based on the *maximization* of dependence between the input variables and their cluster labels. It is a unified framework for various clustering methods

[1] See the survey by Chlamtac and Tulsiani (2012) for an overview of results.

arising from geometric, spectral or statistical considerations, and it has connections to k-means, clustering under topological constraints, and hierarchical clustering. Formally, their goal is to *maximize* objective ($\star$) under the assumption that both the distance matrix d and the cost matrix $\mathbf{A}$ are positive semidefinite. On the other hand, while we require d to be a metric, we don't require d and $\mathbf{A}$ to be positive semidefinite.

Kernel Clustering is a generalization of the positive semidefinite Grothendieck problem (Nesterov 1998) that has found many algorithmic applications (Alon and Naor 2004; Charikar and Wirth 2004; Charikar, Makarychev, and Makarychev 2009b), and has further connections to semidefinite programming, non-convex optimization and the Unique Games Conjecture (Khot and Naor 2008, 2013). Khot and Naor (2008, 2013) studied Kernel Clustering, presenting constant factor approximations and hardness results. In our paper, we show a PTAS for the minimization version of the problem under metric assumption.

2.1 Main Result

The main question we address here is the following:

What is the best approximation for the Weighted Metric Clustering objective ($\star$)?

Our main result shows that we can get an arbitrary good approximation.

Theorem 2.1. *(Informal) There is a PTAS[2] for the Weighted Metric Clustering objective ($\star$).*

As a corollary, we get a PTAS not only for all the above-mentioned problems, but also many more, since any choice of the matrix $\mathbf{A}$ generates a new, different clustering objective. In particular, with careful choice of $\mathbf{A}$, we provide PTAS's for problems where the PTAS's were not previously known, such as clustering on manifolds and a family of hierarchical clustering objectives (Section 3), where each pair of elements is penalized depending on the depth of their least common ancestors. We describe the depth-based hierarchical clustering objectives in Section 3, with additional motivation based on the Minimum Logarithmic Arrangement presented in Appendix F, and we empirically demonstrate the advantage of these objectives in Section 6.

Note that without metric assumption, we cannot have a PTAS even when $k = 3$ (Khot and Naor 2013) under the Unique Games Conjecture, hence it's remarkable that a PTAS for the metric minimization version is possible. Moreover, we handle Weighted Metric Clustering using a single algorithmic technique via the Sherali–Adams linear programming (Sherali and Adams 1990). This subsumes several prior works, unifies many of the techniques on various clustering objectives, and yields PTAS's for new problems, including a new family of hierarchical clustering objectives.

Our Techniques While it is already known that the Sherali–Adams hierarchy can be used to get PTAS's for CSPs, the naïve approach would result in additive error terms, which can be acceptable for *maximization* objectives but are intolerable for *minimization* objectives, such as ($\star$). Our algorithm makes Sherali–Adams relaxations applicable to a wide class of minimization objectives and has two stages: Stage I assigns most of the elements via independent rounding and Stage II carefully handles the rest of the points, which we refer to as outliers. To handle the outliers, we rely on the second objective $\mathrm{LP_{II}}$, which is optimized simultaneously with the Sherali–Adams relaxation $\mathrm{LP_I}$; formally, we minimize $\max(\mathrm{LP_I}, \mathrm{LP_{II}})$ and ensure that it is upper-bounded by OPT. On the other hand, a solution to $\mathrm{LP_{II}}$ simplifies the process of assigning the outliers to the clusters. See Section 4 for the details.

Practical Algorithm and Experiments In Section 6, we introduce a practical version of our algorithm based on $\mathrm{LP_{II}}$, which provides a constant-factor approximation to objective ($\star$). We run our experiments on 10^4 data points and show that our hierarchical clustering objective recovers a ground-truth clustering better compared to the popular Dasgupta's objective (Dasgupta 2016).

3 Application to Hierarchical Clustering

We showcase how our general Weighted Metric Clustering framework ($\star$) can be applied to the problem of finding a hierarchy over clusters rather than a partition. In *Hierarchical Clustering* (*HC*), given a set of points V, the goal is to bijectively map the points on the leaves of a tree $\mathcal{T}$. HC is a very popular method with a wide range of applications (Leskovec, Rajaraman, and Ullman 2020). Recent literature (Dasgupta 2016; Moseley and Wang 2017; Cohen-Addad, Kanade, Mallmann-Trenn, and Mathieu 2019) introduces a number of HC objectives where, for the hierarchical tree $\mathcal{T}$, each pair of elements (u, v) is penalized based on the number of leaves under the Lowest Common Ancestor (LCA) of u and v in $\mathcal{T}$, denoted as $\mathrm{LCA}_{\mathcal{T}}(u, v)$ (for literature review, see Appendix F). Instead of using the number of leaves under the LCA, here we propose an optimization objective for HC where the penalty term is defined based on the depth of the LCA. For a node $v \in \mathcal{T}$, let $h(v)$ denote the depth of v in the tree, defined as the number of edges on the shortest path from the root to v (e.g. $h(r) = 0$ if r is the root node). Our goal is to minimize the following over all possible binary trees $\mathcal{T}$:

$$H(\mathcal{T}) = \sum_{u,v \in V} d_{uv}\, h(\mathrm{LCA}_{\mathcal{T}}(u, v)) \qquad \text{(Depth-HC)}$$

Here d is a metric, and we shall note that HC has been extensively studied for metric spaces (Agarwala, Bafna, Farach, Paterson, and Thorup 1998; Ailon and Charikar 2005; Dasgupta and Long 2005). Objective Depth-HC captures the fact that it is better to separate the distant points early in the hierarchical structure, i.e. $h(\mathrm{LCA}_{\mathcal{T}}(u, v))$ should be small when d_{uv} is large. For HC, we show the following result (proof in Appendix F).

Theorem 3.1. *For any metric d, there exists a PTAS for minimizing the objective Depth-HC.*

[2] For a minimization problem, a PTAS is an algorithm that, given $\varepsilon > 0$ as a parameter, returns a $(1 + \varepsilon)$-approximation to the optimal value and runs in polynomial time for any constant ε. For maximization, we seek a $(1 - \varepsilon)$-approximation.

In order to map the HC objective to Weighted Metric Clustering ($\star$), we must appropriately choose matrix $\mathbf{A}$. The main idea is to show that it suffices to recover the tree up to the depth $\log(1/\varepsilon)$ and build random trees on deeper levels. Let $\mathcal{T}_\circ$ be a full binary tree $\mathcal{T}_\circ$ of depth $\log(1/\varepsilon)$, and we associate the $k = 1/\varepsilon$ leaves $\ell_1, \dots, \ell_k$ of $\mathcal{T}_\circ$ with corresponding clusters $C_1, \dots, C_k$. For different clusters C_i and C_j, we define A_{ij} as the depth of their LCA, i.e. $h(\mathrm{LCA}_{\mathcal{T}_\circ}(\ell_i, \ell_j))$.

Depth-based objectives are useful in Graph Compression and Vertex Reordering problems (Raghavan and Garcia-Molina 2003; Boldi and Vigna 2004; Chierichetti et al. 2009; Dhulipala et al. 2016), where the goal is to find space-efficient labeling schemes for the nodes in the graph. Roughly speaking, the depth $h(\mathrm{LCA}_{\mathcal{T}}(i,j))$ corresponds to the bits needed to represent a vertex in the graph, and, exploiting the fact that similar nodes tend to have similar sets of neighbors, one can significantly reduce the bit-complexity of the graph representation. A more in-depth discussion and the proof of Theorem 3.1 are deferred to Appendix F.

Extensions. We note that our result for objective Depth-HC in Theorem 3.1 also holds for more general cost functions than the hierarchical clustering objective Depth-HC specified above. For example, instead of the depth of the lowest-common ancestor, $h(\mathrm{LCA}_{\mathcal{T}}(i,j))$, we could also penalize according to the logarithm of the depth, i.e. $\log h(\mathrm{LCA}_{\mathcal{T}}(i,j))$, or the square of the depth, i.e. $h^2(\mathrm{LCA}_{\mathcal{T}}(i,j))$; our algorithms and proofs would still guarantee a PTAS in these cases. In fact, any function which depends on the depth subexponentially works. For the formal statement regarding the more general hierarchical clustering objectives, see Appendix F.

4 Sherali–Adams and Local Probability Distributions

Our $(1+\varepsilon)$-approximation algorithm for Weighted Metric Clustering uses a Sherali–Adams relaxation for the problem. Sherali–Adams (Sherali and Adams 1990) is a lift-and-project method for strengthening linear programming (LP) relaxations. In this paper, we will use a "local probability distribution" approach to Sherali–Adams (de la Vega and Kenyon-Mathieu 2007; Charikar, Makarychev, and Makarychev 2009a). We also use a method for removing dependencies between random variables in local distributions, which was developed by Raghavendra and Tan (2012) (see also Barak, Raghavendra, and Steurer (2011) and Yoshida and Zhou (2014)).

We now describe the Sherali–Adams LP relaxation. For every tuple of points $\mathbf{v} \in V^r$, where $r \geq 2$ is a fixed integer parameter, we have a set of LP variables that defines a probability distribution of "labels" on $v_1, \dots, v_r$. For every $\boldsymbol{\ell} \in \{1, \dots, k\}^r$, we introduce a variable $\mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big]$. Each of these k^r variables (sometimes called pseudo-probabilities) lies in $[0,1]$ and represents the probability that point v_i is assigned to cluster C_{ℓ_i} for all i.[3] For every $\mathbf{v} \in V^k$, the linear programming relaxation has the constraint $\sum_{\boldsymbol{\ell} \in \{1,\dots,k\}^r} \mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big] = 1$. This constraint ensures that in a feasible LP solution, every $\mathbb{P}_{\mathbf{v}}$ indeed defines a local probability distribution on points $v_1, \dots, v_r$.

We also add a constraint that guarantees that this probability does not depend on the order of points $v_1, \dots, v_r$. For example, for $r = 2$, we impose constraint $\mathbb{P}[a \in C_1, b \in C_2] = \mathbb{P}[b \in C_2, a \in C_1]$, where a and b are arbitrary points from V. Specifically, for every permutation σ of $\{1, \dots, k\}$:

$$\begin{aligned}&\mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big]\\ &= \mathbb{P}_{\mathbf{v}}\big[v_{\sigma(1)} \in C_{\ell_{\sigma_1}}, \dots, v_{\sigma_k} \in C_{\ell_{\sigma_k}}\big].\end{aligned}$$

LP variables $\mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big]$ prescribe probabilities to elementary events $\big\{v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big\}$ and thus define probabilities for all events: for $\mathcal{E} \subseteq \{1, \dots, k\}^r$, we let $\mathbb{P}_{\mathbf{v}}[\mathbf{v} \in \mathcal{E}] = \sum_{\boldsymbol{\ell} \in \mathcal{E}} \mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big]$. In other words, $\mathbb{P}_{\mathbf{v}}[\mathbf{v} \in \mathcal{E}]$ is the probability that labels for $v_1, \dots, v_r$ drawn from local distribution $\mathbb{P}$ are $C_{\ell_1}, \dots, C_{\ell_r}$ (respectively) with $\boldsymbol{\ell} \in \mathcal{E}$. To avoid ambiguity, we will use a different notation to denote probabilities associated with our algorithm. We shall write $\Pr[v_1 \in X_1, \dots, v_r \in X_r]$ to denote the probability that points $v_1, \dots, v_r$ belong to random sets $X_1, \dots, X_r$ chosen by the algorithm.

An important constraint of the Sherali–Adams relaxation is that *all local distributions are locally consistent*, as we explain next. Consider two tuples $\mathbf{u}$ and $\mathbf{v}$. Let $\mathbf{z}$ be the set of common points in $\mathbf{u}$ and $\mathbf{v}$. Both $\mathbf{u}$ and $\mathbf{v}$ define marginal probability distributions on cluster labels for points in $\mathbf{z}$. We require that these marginal distributions be the same. Specifically, we add a constraint to the linear program that enforces that label distributions on $\mathbf{u}$ and $\mathbf{v}$ agree on the intersection $\mathbf{z} = \mathbf{u} \cap \mathbf{v}$. We denote the marginal probability distribution on every set $\mathbf{z}$ of size at most r by $\mathbb{P}_{\mathbf{z}}$. If $\mathbf{z}$ consists of one point u or two points u,v, we write $\mathbb{P}_u$ and $\mathbb{P}_{uv}$, respectively.

We stress that even though all local distributions $\mathbb{P}$ are locally consistent, generally speaking, there is no global distribution of cluster labels that is consistent with all local distributions. We also note that the size of the Sherali–Adams is exponential in r, since the number of variables equals $n^r \cdot k^r$. Thus, if we want to solve a Sherali–Adams relaxation in polynomial time, the parameter r must be a constant.

When each variable in a solution to the Sherali–Adams relaxation is equal to 0 or 1, we call the solution integral. An integral solution corresponds to an actual clustering in which u belongs to C_i if and only if $\mathbb{P}_u[u \in C_i] = 1$. Moreover, $\mathbb{P}_{\mathbf{v}}\big[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\big] = 1$ if and only if $v_1 \in C_{\ell_1}$,..., $v_r \in C_{\ell_r}$. That is, $\mathbb{P}_{\mathbf{v}}\left[v_1 \in C_{\ell_1}, \dots, v_r \in C_{\ell_r}\right] = \mathbb{1}\{v_1 \in C_{\ell_r}, \dots, v_r \in C_{\ell_r}\}$, where $\mathbb{1}\{\mathcal{E}\}$ is the indicator of the event $\mathcal{E}$. We now define the objective function for our Sherali–Adams relaxation and introduce some additional constraints. We assume that we know the sizes of the optimal clusters $n_1 = |C_1^*|$,..., $n_k = |C_k^*|$. We additionally assume that we know their centers $c_1 \in C_1^*, \dots, c_k \in C_k^*$ which guarantee 3-approximation (see Lemma B.2). Note that there are at most $O(n^{2k})$ combinations of different c_i's and n_j's, and hence we can try all possibilities. We use $\mathbf{\Pi}$ to denote the particular choice of c_i's and n_j's and call it the clustering profile.

[3] Formally, one should think about assigning point v_i to C_{ℓ_i} as of assigning label ℓ_i to point v_i

The objective of our linear programming relaxation is the maximum of $\mathrm{LP_I}$ and $\mathrm{LP_{II}}$ under the constraints above:

$$\textbf{minimize } LP = \max(\mathrm{LP_I}, \mathrm{LP_{II}}), \tag{1}$$

$$\mathrm{LP_I} = \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k}\sum_{u,v\in V} A_{ij}\, d_{uv}\, \mathbb{P}_{uv}\big[u \in C_i,\, v \in C_j\big]$$

$$\mathrm{LP_{II}} = \frac{1}{3}\sum_{i=1}^{k}\sum_{u\in V} F_{\mathbf{\Pi}}(u,i)\, \mathbb{P}_u\big[u \in C_i\big],$$

$$F_{\mathbf{\Pi}}(u,i) = \sum_{j=1}^{k} n_j\, a_{ij}\, d_{uc_{i\wedge j}}, \tag{2}$$

where $i \wedge j$ is defined as $\min(i,j)$. The first objective $\mathrm{LP_I}$ is a direct relaxation of the objective function of Weighted Metric Clustering: in an integral LP solution – when each $\mathbb{P}_{uv}\big[u \in C_i,\, v \in C_j\big]$ is 0 or 1 – the value of $\mathrm{LP_I}$ equals the cost of the corresponding combinatorial solution to Weighted Metric Clustering. Consequently, in the optimal integral solution to the problem, $\mathrm{LP_I} = \mathrm{OPT}$, where $\mathrm{OPT} = \textsc{cost}(C_1^*, \ldots, C_k^*)$ is the value of the optimal solution.

The second objective $\mathrm{LP_{II}}$ is upper bounded by OPT in the optimal integral solution by Lemma B.2 when c_i's and n_j's are guessed correctly. This is due to the fact that for any $u \in V$, $i \in [k]$, and the correct guess of n_i and c_i, $n_i\, d_{uc_i}$ is a good approximation of $\sum_{v\in C_j} d_{uv}$ (de la Vega, Karpinski, Kenyon, and Rabani 2003). Therefore, $\mathrm{OPT}_{\mathrm{LP_I}} \le \mathrm{OPT}$ and $\mathrm{OPT}_{\mathrm{LP_{II}}} \le \mathrm{OPT}$, where $\mathrm{OPT}_{\mathrm{LP_I}}$ and $\mathrm{OPT}_{\mathrm{LP_{II}}}$ are the values of $\mathrm{LP_I}$ and $\mathrm{LP_{II}}$ in the optimal solution to our linear program.

Intuitively, $\mathrm{LP_{II}}$ is used for bounding the error terms in the analysis and, compared to $\mathrm{LP_{II}}$, has the following advantages. First, every term involves a single point u (note that other variables in each term are either guessed or fixed), and hence it's easy to optimize. Second, $\mathrm{LP_{II}}$ refers to cluster centers instead of clusters themselves, which is important for the case when there are multiple equivalent solutions to the original problem, e.g. in the case of Min-Uncut. In the analysis, we often use triangle inequality to bound $d_{uv} \le d_{uc} + d_{cv}$, with the choice of c being crucial. $\mathrm{LP_{II}}$ forces the center for each cluster, which makes the choice of c clear in each particular case.

Finally, we add capacity constraints to our relaxation, which are satisfied in the integral solution to Weighted Metric Clustering. For all $i \in \{1, \ldots, k\}$: $\sum_{u\in V} \mathbb{P}_u[u \in C_i] \le n_i$. These constraints are important since $\mathrm{LP_{II}}$ is a good approximation of $(\star)$ only if cardinalities are guessed and enforced correctly.

4.1 Making Point Distributions Nearly Independent

We now define *nearly independent* local distributions and then describe a procedure MAKEINDEPENDENT that transforms local distributions $\mathbb{P}$ obtained by solving the Sherali–Adams LP relaxation into a nearly independent local distributions $\mathbb{P}^*$. This procedure uses the *conditional probability technique* for Sherali–Adams (Raghavendra and Tan 2012). The main difference between our result and theirs is that we require that local distributions $\mathbb{P}^*$ (see below) are simultaneously nearly independent for k sets $D_1, \ldots, D_k$, while Raghavendra and Tan (2012) obtain a globally uncorrelated solution which corresponds to the case when we have only one set $A = V$. For us, it is crucial to have sets $D_1, \ldots, D_k$ in the definition because some sets D_i may have size $o(n)$ (e.g., $\sqrt{n}$). In that case, the guarantees of the algorithm by Raghavendra and Tan (2012) are not sufficient for us.

First, we introduce some notation. Denote the distribution of pairs u and v in which u and v are sampled independently with distributions $\mathbb{P}_u$ and $\mathbb{P}_v$ by $\mathbb{P}_u \otimes \mathbb{P}_v$:

$$(\mathbb{P}_u \otimes \mathbb{P}_v)[u \in C_i,\ v \in C_j] = \mathbb{P}_u[u \in C_i] \cdot \mathbb{P}_v[v \in C_j].$$

Definition 4.1. Let $D_1, \ldots, D_k$ be subsets of V. We say that a family of local probability distributions $\{\mathbb{P}\}$ are (γ,δ)-nearly independent for sets $D_1, \ldots, D_k$ if the following condition holds: for every $u \in V$ and every $j \in \{1, \ldots, k\}$, for all but γ fraction of v in D_j, we have $\|\mathbb{P}_u \otimes \mathbb{P}_v - \mathbb{P}_{u,v}\|_{TV} \le \delta$. Equivalently, for all $u \in V$ and $i \in [k]$, the number of elements $v \in D_j$ such that $\|\mathbb{P}_u \otimes \mathbb{P}_v - \mathbb{P}_{u,v}\|_{TV} > \delta$ must be at most $\gamma|D_i|$. If $\|\mathbb{P}_u \otimes \mathbb{P}_v - \mathbb{P}_{u,v}\|_{TV} \le \delta$, we say that u and v are δ-nearly independent according to $\mathbb{P}_{u,v}$.

Theorem 4.2. *For every $\delta, \gamma, \eta \in (0,1)$ and integer $k > 1$, there exists a randomized polynomial-time procedure that given a solution $\mathbb{P}$ to the Sherali–Adams relaxation with $r \ge 2 + \frac{k \log_2 k}{2\delta^2\gamma\eta}$ rounds, outputs a family of local probability distributions $\{\mathbb{P}^*_u\}_u$ and $\{\mathbb{P}^*_{uv}\}_{uv}$ and exit status ("success" or "failure") such that*

1. *If the algorithm succeeds, then $\mathbb{P}^*$ is (γ,δ)-nearly independent.*
2. *For all $u, v \in V$ and $i, j \in \{1, \ldots, k\}$,*
$$\mathbb{E}\left[\mathbb{P}^*_u[u \in C_i]\right] = \mathbb{P}_u[u \in C_i]$$
$$\mathbb{E}\left[\mathbb{P}^*_{uv}[u \in C_i, v \in C_j]\right] = \mathbb{P}_{uv}[u \in C_i, v \in C_j].$$
3. *The algorithm fails with probability at most η.*

The goal of algorithm MAKEINDEPENDENT is to build a (γ,δ)-nearly independent family $\{\mathbb{P}^*\}$ while preserving the expectation of the LP value. The algorithm builds a sequence of distributions $\{\mathbb{P}^{(0)}\} = \{\mathbb{P}\}, \{\mathbb{P}^{(1)}\}, \ldots$. At iteration t, it finds a point u violating the (γ,δ)-nearly independence condition, and then *conditions* local distributions on the event $\mathbb{P}^{(t)}[u \in C_i]$ for i drawn from distribution $\mathbb{P}^{(t)}_u$. Loosely speaking, every time we do the conditioning step, we make more pairs (u,v) nearly independent. We show that a certain measure – entropy – decreases with each iteration by at least a fixed amount, and hence in approximately r steps, we get nearly independence with the desired parameters. We provide more details and prove this theorem in Appendix C.

5 Main Algorithm

In this section, we outline our $(1+\varepsilon)$-approximation algorithm or PTAS (polynomial-time approximation scheme) for the Weighted Metric Clustering problem. We provide full details in Appendices D and E. The pseudocode is provided in Algorithm 1.

Algorithm 1: PTAS for Weighted Metric Clustering

input : V – set of points, $\{d_{uv}\}_{u,v\in V}$ – pairwise distances, $\{A_{ij}\}_{i,j=1}^{k}$ – inter-cluster costs

parameters: r – number of rounds of SA relaxation, η – outlier probability threshold, δ – fraction of dependent points, γ – independence threshold

Guess cluster centers $c_1, \dots, c_k$ and sizes $n_1, \dots, n_k$

Let $\{\mathbb{P}\}$ be the r-round solution to SA relaxation for Problem (1)

$D_i = \{u \in V : \mathbb{P}_u[u \in C_i] \geq \eta\}$ for all i

$\{\mathbb{P}^*\} = \textsc{MakeIndependent}(\{\mathbb{P}\}, \{D_i\}, \delta, \gamma)$

`// Tentative assignment via independent rounding`

for *all* $u \in V$ **do**

  Assign u to C_i with probability $\mathbb{P}^*[u \in C_i]$

`// Stage I: Assigning non-outliers`

if $\mathbb{P}^*$ *is* (γ, δ)*-nearly independent for* $D_1, \dots, D_k$ **then**

  $X_i = C_i \cap D_i, \quad O = \bigcup_i (C_i \setminus D_i)$

else

  $O = V$ `// Every point is outlier`

`// Stage II: Assigning outliers`

for *all* $u \in O$ **do**

  Assign u to Y_i with probability $\mathbb{P}[u \in C_i]$.

return $(X_1 \cup Y_1, \dots, X_k \cup Y_k)$

Algorithm 2: $\textsc{MakeIndependent}(\{\mathbb{P}\}, \{D_i\}, \delta, \gamma)$

input : $\{\mathbb{P}\}$ – r-round solution to SA relaxation, $\{D_i\}_{i=1}^{k}$ – candidate sets for each cluster, δ – fraction of dependent points, γ – independence threshold

Let $\{\mathbb{P}^{(0)}\}$ be $\{\mathbb{P}\}$

for $t = 0, 1, \dots, r-3$ **do**

  if $\{\mathbb{P}^{(t)}\}$ *is* (γ, δ)*-nearly independent for sets* $D_1, \dots, D_k$ *(Def. 4.1)* **then**

    return $\{\mathbb{P}^{(t)}\}$

  Let u be a point violating the (γ, δ)-nearly independence condition.

  Assign u to C_i with probability $\mathbb{P}^{(t)}[u \in C_i]$.

  Let $\{\mathbb{P}^{(t+1)}\}$ be $\{\mathbb{P}^{(t)}\}$ conditioned on $u \in C_i$.

return $\{\mathbb{P}^{(r-2)}\}$

Algorithm Outline In the first step, the algorithm guesses the cluster centers $\{c_i\}$ and sizes $\{n_j\}$, which we call the clustering profile and denote by $\mathbf{\Pi}$. Note that all choices of $\{c_i\}$ and $\{n_j\}$ can be enumerated in polynomial time, and our analysis assumes the correct choice. Then, the algorithm solves the r-round Sherali–Adams relaxation for Weighted Metric Clustering (see Section 4) and obtains local distributions $\mathbb{P}$. For constant r, the size of the relaxation is polynomial in n, and thus it can be solved in polynomial time. We then assign points to clusters using a two-stage algorithm.

At Stage I, we assign most points to clusters $X_1, \dots, X_k$ and place the remaining points, which we call "outliers", in set O. We guarantee that the cost of the partial clustering $X_1, \dots, X_k$ is at most $(1+\varepsilon)\mathrm{OPT}$ in expectation and each point is an outlier with probability at most ηk (see Lemma D.2 for the formal statement), where η is a small parameter depending on ε.

At Stage II, we cluster the outliers from set O. For this purpose, we use a variant of the 3-approximation algorithm, which we provide in Section B. Since the number of outliers is very small, the cost of clustering them is also small despite the fact that we use a constant factor approximation for outliers. Finally, we combine the clusterings obtained at Stage I and Stage II and get a clustering of cost at most $(1+\varepsilon)\mathrm{OPT}$. The algorithm for clustering outliers is discussed in Appendix E.

Stage I We now examine the first stage of the algorithm in more detail. It is inspired by Yoshida and Zhou (2014) and Raghavendra and Tan (2012). The general idea is to transform the solution for the Sherali–Adams relaxation to a family of local distributions $\{\mathbb{P}^*\}$ such that

$$\mathbb{P}^*_{uv}[u \in C_i; v \in C_j] \approx \mathbb{P}^*_u[u \in C_i] \cdot \mathbb{P}^*_v[v \in C_j] \qquad (3)$$

for most pairs of points. This can be done using the method discussed in the previous section. Next, we want to independently assign every point u to cluster i with probability $\mathbb{P}_u[u \in C_i]$. If condition (3) holds for some pair (u, v) and all i, j, then the expected cost this algorithm pays for clustering pair (u, v), $\sum_{ij} d_{uv} A_{ij} \mathbb{P}_u\big[u \in C_i\big] \cdot \mathbb{P}_v\big[v \in C_j\big]$, is approximately equal to the LP cost of this pair, $\sum_{ij} d_{uv} A_{ij} \mathbb{P}_{uv}\big[u \in C_i,\, v \in C_j\big]$. The problem, however, is that condition (3) does not hold for all pairs (u, v). Furthermore, it may happen that the algorithm creates a very expensive small cluster X_i such that for all pairs $u, v \in X_i$ we do not have approximate equality (3). Consequently, the cost of such a cluster cannot be charged to the LP relaxation.

The discussion above leads to the following idea: let us make local distributions not only nearly independent for most pairs (u, v) but nearly independent for each point u and most v's in each cluster the algorithm creates. This is formally stated in Definition 4.1. However, the problem is that the algorithm does not know in advance what clusters it is going to produce. So, it uses a proxy for these clusters – sets of candidate points $D_1, \dots, D_k$. Set D_i contains points that are somewhat likely to be assigned to cluster i.

We now summarize Stage I. First, the algorithm solves the Sherali–Adams relaxation. Then, it defines sets of candidates $D_1, \dots, D_k$, where each D_i contains points u for which $\mathbb{P}[u \in C_i] \geq \eta$ (where η is a small constant depending on ε). It calls algorithm $\textsc{MakeIndependent}$ (described Section 4.1) with sets $D_1, \dots, D_k$ and obtains (γ, δ)-nearly independent local distributions $\mathbb{P}^*$. Next, it randomly assigns points to clusters using distribution $\mathbb{P}^*$. To make sure that we can pay for each created cluster X_i, this cluster needs to be a subset of the corresponding candidate set D_i. Thus, if point u is assigned to X_i but u is not in D_i, we remove u from X_i and mark u as an outlier. Stage I returns sets $X_1, \dots, X_k$

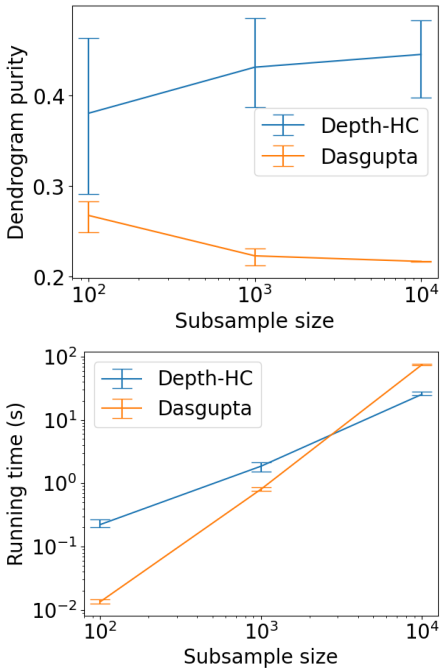

Figure 1: Comparison of the depth-based objective (Depth-HC) and the Dasgupta's objective. Data points correspond to averages over 10 runs, and error bars correspond to the 10% and 90% quantiles. The experiments are performed on a single-core Intel Xeon 2.2GHz CPU.

along with the set of outliers O, which are assigned to clusters at Stage II. We can now charge the cost of all pairs (u, v) that are nearly independent to the LP objective. Using triangle inequalities, we can also bound the cost of all other pairs (u, v) in $V \setminus O$. We provide all details in Appendix D.

Stage II At Stage II, we assign outliers to clusters. Our approach to dealing with outliers is somewhat similar to the approach introduced by Makarychev, Makarychev, and Razenshteyn (2019). As discussed above, the number of outliers is small, which is one of the main reasons why their assignment does not significantly change the objective. The outliers are assigned using independent rounding based on $\mathbb{P}$ (instead of $\mathbb{P}^*$ for non-outliers). In Theorem E.3, we analyze the cost of assigning outliers to clusters. Putting everything together, we prove that our algorithm provides a PTAS.

Theorem 5.1. *For* $\delta = \gamma = \frac{\eta^2\varepsilon}{9}$ *and* $r = 2 + \frac{k \log_2 k}{2\delta^2\gamma\eta}$, *Algorithm 1 finds clustering with the expected objective value within* $(1+\varepsilon)$*-factor of* OPT *with probability at least* $1 - \eta$ *for any* $\eta \le \frac{\varepsilon^2}{90k^2}$.

6 Experiments

In this section, we perform experiments on the hierarchical clustering objective (Depth-HC) defined in Section 3:

$$H(\mathcal{T}) = \sum_{u,v \in V} d_{uv}\, h(\mathrm{LCA}_{\mathcal{T}}(u, v))$$

For our experiments, we use a simplified version of the algorithm, based on the LP_{II} relaxation from Section 4, which achieves 3-approximation (Appendix B):

$$\sum_{i=1}^{k} \sum_{j=1}^{k} \sum_{u \in V} n_j\, A_{ij}\, d_{u c_{i \wedge j}}\, \mathbb{P}_u\big[u \in C_i\big],$$

where $n_1, \dots, n_k$ are cluster cardinalities and $c_1, \dots, c_k$ are cluster centers. This objective can be optimized efficiently as an instance of a minimum-cost flow problem, while precisely satisfying the imposed cardinality constraints. We run the algorithm multiple times with different guesses of $\{n_i\}$ and $\{c_i\}$, and, since the guesses might be not precise, we improve the resulting solution using local search.

Datasets We perform evaluation on various hierarchical datasets. In this section, we present experiments on random subsamples (of sizes 10^2, 10^3, and 10^4) of a well-known 20 NEWSGROUPS dataset (Lang 1995), and in Appendix G we present additional experiments on ZEBRAFISH (Wagner et al. 2018), CIFAR-10 (Krizhevsky and Hinton 2009), and other datasets. The inputs in 20 NEWSGROUPS are text documents, which we transform to the Euclidean vectors using a pre-trained language model (see Appendix G for details). Finally, we use the ground-truth hierarchical structure to obtain a flat clustering based on the top-level split.

Objectives We compare the following objectives:

- **Depth-based objective** (Depth-HC). Based on the algorithm from Section 3, we approximate the objective by building a hierarchical tree up to a certain level and building random trees on the resulting clusters. We select the level ℓ so that the number of clusters 2^ℓ is close to the number of ground-truth clusters.
- **Dasgupta's objective (Dasgupta 2016)**, defined as $\sum_{u<v} w(u, v)\, |\mathrm{LCA}_{\mathcal{T}}(u, v)|$, where w is the similarity between items. We convert distances to similarities using the standard RBF kernel: $w(x, y) = \exp\left(-\frac{\|x-y\|^2}{2}\right)$. We optimize Dasgupta's objective using recursive Min-Cut (Chatziafratis et al. 2020), for which we use METIS (Karypis and Kumar 1995).

Evaluation and Results We evaluate how well the above objectives recover the ground-truth clustering information using the dendrogram purity objective:

$$DP(\mathcal{T}) = \frac{1}{\sum_{i=1}^{m} |C_i|^2} \sum_{i=1}^{m} \sum_{u,v \in C_i} \frac{|C_i \cap \mathrm{LCA}_{\mathcal{T}}(u, v)|}{|\mathrm{LCA}_{\mathcal{T}}(u, v)|},$$

where $C_1, \dots, C_m$ are the ground-truth clusters. Intuitively, this objective measures how well-separated are the ground-truth clusters in the tree.

Figure 1 shows that (Depth-HC) objective achieves significantly better dendrogram purity compared with Dasgupta's objective. Moreover, the complexity of our algorithm is noticeably slower, and, with the increase in the number of data points, the gap in quality increases, exceeding the factor of two for 10^4 points. To conclude, these experiments demonstrate the usefulness of our hierarchical objective as well as the existence of efficient approaches for its optimization.

We provide additional experiments in Appendix G.

Acknowledgements

Konstantin Makarychev is supported by the NSF Awards CCF-1955351, CCF-1934931, and EECS-2216970.

References

Agarwala, R.; Bafna, V.; Farach, M.; Paterson, M.; and Thorup, M. 1998. On the approximability of numerical taxonomy (fitting distances by tree metrics). *SIAM Journal on Computing*, 28(3): 1073–1085.

Ailon, N.; and Alon, N. 2007. Hardness of fully dense problems. *Information and Computation*, 205(8): 1117–1129.

Ailon, N.; and Charikar, M. 2005. Fitting tree metrics: Hierarchical clustering and phylogeny. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, 73–82. IEEE.

Alekhnovich, M.; Arora, S.; and Tourlakis, I. 2011. Towards strong nonapproximability results in the Lovász-Schrijver hierarchy. *computational complexity*, 20: 615–648.

Alon, N.; Azar, Y.; and Vainstein, D. 2020. Hierarchical clustering: A 0.585 revenue approximation. In *Conference on Learning Theory*, 153–162. PMLR.

Alon, N.; de la Vega, W. F.; Kannan, R.; and Karpinski, M. 2002. Random sampling and approximation of MAX-CSP problems. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 232–239.

Alon, N.; and Naor, A. 2004. Approximating the cut-norm via Grothendieck's inequality. In *Proceedings of the 36th annual ACM symposium on Theory of computing*, 72–80.

Arora, S.; Karger, D. R.; and Karpinski, M. 1999. Polynomial Time Approximation Schemes for Dense Instances of NP-Hard Problems. *J. Comput. Syst. Sci.*, 58(1): 193–210.

Bansal, N.; Blum, A.; and Chawla, S. 2004. Correlation clustering. *Machine learning*, 56(1): 89–113.

Barak, B.; Raghavendra, P.; and Steurer, D. 2011. Rounding semidefinite programming hierarchies via global correlation. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, 472–481. IEEE.

Bartal, Y.; Charikar, M.; and Raz, D. 2001. Approximating min-sum k-clustering in metric spaces. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, 11–20.

Bazgan, C.; de la Vega, W. F.; and Karpinski, M. 2003. Polynomial time approximation schemes for dense instances of minimum constraint satisfaction. *Random Struct. Algorithms*, 23(1): 73–91.

Boldi, P.; and Vigna, S. 2004. The webgraph framework I: compression techniques. In *Proceedings of the 13th international conference on World Wide Web*, 595–602.

Charikar, M.; and Chatziafratis, V. 2017. Approximate hierarchical clustering via sparsest cut and spreading metrics. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 841–854. SIAM.

Charikar, M.; Guruswami, V.; and Wirth, A. 2005. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3): 360–383.

Charikar, M.; Hajiaghayi, M. T.; Karloff, H.; and Rao, S. 2010. ℓ_2^2 spreading metrics for vertex ordering problems. *Algorithmica*, 56(4): 577–604.

Charikar, M.; Makarychev, K.; and Makarychev, Y. 2009a. Integrality gaps for Sherali-Adams relaxations. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 283–292.

Charikar, M.; Makarychev, K.; and Makarychev, Y. 2009b. Near-optimal algorithms for maximum constraint satisfaction problems. *ACM Transactions on Algorithms (TALG)*, 5(3): 1–14.

Charikar, M.; and Wirth, A. 2004. Maximizing quadratic programs: Extending Grothendieck's inequality. In *45th Annual IEEE Symposium on Foundations of Computer Science*, 54–60. IEEE.

Chatziafratis, V.; Yaroslavtsev, G.; Lee, E.; Makarychev, K.; Ahmadian, S.; Epasto, A.; and Mahdian, M. 2020. Bisect and Conquer: Hierarchical Clustering via Max-Uncut Bisection. In *Proceedings of the 33rd International Conference on Artificial Intelligence and Statistics*, 3121–3132. PMLR.

Chierichetti, F.; Kumar, R.; Lattanzi, S.; Mitzenmacher, M.; Panconesi, A.; and Raghavan, P. 2009. On compressing social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 219–228.

Chlamtac, E.; and Tulsiani, M. 2012. Convex relaxations and integrality gaps. *Handbook on semidefinite, conic and polynomial optimization*, 139–169.

Cohen-Addad, V.; Das, D.; Kipouridis, E.; Parotsidis, N.; and Thorup, M. 2022. Fitting distances by tree metrics minimizing the total error within a constant factor. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, 468–479. IEEE.

Cohen-Addad, V.; Kanade, V.; Mallmann-Trenn, F.; and Mathieu, C. 2019. Hierarchical clustering: Objective functions and algorithms. *Journal of the ACM*, 66(4): 1–42.

Cohen-Addad, V.; Lee, E.; and Newman, A. 2022. Correlation clustering with sherali-adams. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 651–661. IEEE.

Costa, M.-C.; Létocart, L.; and Roupin, F. 2005. Minimal Multicut and Maximal Integer Multiflow: A Survey. *European Journal of Operational Research*, 162(1): 55–69.

Dahlhaus, E.; Johnson, D. S.; Papadimitriou, C. H.; Seymour, P. D.; and Yannakakis, M. 1994. The Complexity of Multiterminal Cuts. *SIAM Journal on Computing*, 23(4): 864–894.

Dasgupta, S. 2016. A cost function for similarity-based hierarchical clustering. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, 118–127.

Dasgupta, S.; and Long, P. M. 2005. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4): 555–569.

de la Vega, W. F.; Karpinski, M.; and Kenyon, C. 2004. Approximation schemes for Metric Bisection and partitioning.

In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, 506–515. SIAM.

de la Vega, W. F.; Karpinski, M.; Kenyon, C.; and Rabani, Y. 2003. Approximation schemes for clustering problems. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 50–58.

de la Vega, W. F.; and Kenyon-Mathieu, C. 2007. Linear programming relaxations of maxcut. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 53–61.

Dhulipala, L.; Kabiljo, I.; Karrer, B.; Ottaviano, G.; Pupyrev, S.; and Shalita, A. 2016. Compressing graphs and indexes with recursive graph bisection. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1535–1544.

Fernandez de la Vega, W. 1996. MAX-CUT has a randomized approximation scheme in dense graphs. *Random Structures & Algorithms*, 8(3): 187–198.

Frieze, A. M.; and Kannan, R. 1996. The Regularity Lemma and Approximation Schemes for Dense Problems. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, 12–20. IEEE Computer Society.

Giotis, I.; and Guruswami, V. 2005. Correlation clustering with a fixed number of clusters. *arXiv preprint cs/0504023*.

Håstad, J. 2001. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4): 798–859.

Hopkins, S. B.; Schramm, T.; and Trevisan, L. 2020. Subexponential LPs approximate max-cut. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 943–953. IEEE.

Indyk, P. 1999. A Sublinear Time Approximation Scheme for Clustering in Metric Spaces. In *40th Annual Symposium on Foundations of Computer Science*, 154–159.

Kann, V.; Khanna, S.; Lagergren, J.; and Panconesi, A. 1996. On the Hardness of Approximating Max k-Cut and Its Dual. In *Israeli Symposium on Theoretical Computer Science*.

Karypis, G.; and Kumar, V. 1995. Metis-Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 2.0. *University of Minnesota*.

Khot, S. 2002. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 767–775.

Khot, S.; and Naor, A. 2008. Approximate Kernel Clustering. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, 561–570. IEEE Computer Society.

Khot, S.; and Naor, A. 2013. Sharp kernel clustering algorithms and their associated Grothendieck inequalities. *Random Structures & Algorithms*, 42(3): 269–300.

Krizhevsky, A.; and Hinton, G. 2009. Learning Multiple Layers of Features from Tiny Images.

Lang, K. 1995. NewsWeeder: Learning to Filter Netnews. In *Machine Learning Proceedings 1995*, 331–339. San Francisco (CA): Morgan Kaufmann. ISBN 978-1-55860-377-6.

Leskovec, J.; Rajaraman, A.; and Ullman, J. D. 2020. *Mining of massive data sets*. Cambridge university press.

Lovász, L.; and Schrijver, A. 1991. Cones of Matrices and Set-Functions and 0–1 Optimization. *SIAM Journal on Optimization*, 1(2): 166–190.

Makarychev, K.; Makarychev, Y.; and Razenshteyn, I. 2019. Performance of Johnson-Lindenstrauss Transform for k-Means and k-Medians Clustering. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, 1027–1038. Association for Computing Machinery.

Mezei, B. F.; Wrochna, M.; and Živnỳ, S. 2023. PTAS for sparse general-valued CSPs. *ACM Transactions on Algorithms*, 19(2): 1–31.

Moseley, B.; Vassilvtiskii, S.; and Wang, Y. 2021. Hierarchical clustering in general metric spaces using approximate nearest neighbors. In *International Conference on Artificial Intelligence and Statistics*, 2440–2448. PMLR.

Moseley, B.; and Wang, J. 2017. Approximation Bounds for Hierarchical Clustering: Average Linkage, Bisecting K-means, and Local Search. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Nesterov, Y. 1998. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization methods and software*, 9(1-3): 141–160.

Raghavan, S.; and Garcia-Molina, H. 2003. Representing web graphs. In *Proceedings 19th International Conference on Data Engineering*, 405–416. IEEE.

Raghavendra, P.; and Tan, N. 2012. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *Proceedings of the 33rd annual ACM-SIAM symposium on Discrete Algorithms*, 373–387. SIAM.

Romero, M.; Wrochna, M.; and Živnỳ, S. 2021. Treewidth-pliability and PTAS for Max-CSPs. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 473–483. SIAM.

Sherali, H. D.; and Adams, W. P. 1990. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3): 411–430.

Song, L.; Smola, A.; Gretton, A.; and Borgwardt, K. M. 2007. A dependence maximization view of clustering. In *Proceedings of the 24th international conference on Machine learning*, 815–822.

Thapper, J.; and Zivny, S. 2017. The power of Sherali–Adams relaxations for general-valued CSPs. *SIAM Journal on Computing*, 46(4): 1241–1279.

Wagner, D. E.; Weinreb, C.; Collins, Z. M.; Briggs, J. A.; Megason, S. G.; and Klein, A. M. 2018. Single-Cell Mapping of Gene Expression Landscapes and Lineage in the Zebrafish Embryo. *Science*, 360(6392): 981–987.

Yannakakis, M. 1988. Expressing combinatorial optimization problems by linear programs. In *Proceedings of the 20th annual ACM symposium on Theory of computing*, 223–228.

Yoshida, Y.; and Zhou, Y. 2014. Approximation schemes via Sherali-Adams hierarchy for dense constraint satisfaction problems and assignment problems. In *Innovations in Theoretical Computer Science, 2014*, 423–438. ACM.