Correlation Clustering in General Weighted Graphs[*]

Erik D. Demaine[†] Dotan Emanuel[‡] Amos Fiat[†] Nicole Immorlica[*]

February 5, 2005

Abstract

We consider the following general *correlation-clustering problem* [1]: given a graph with real nonnegative edge weights and a $\langle + \rangle / \langle - \rangle$ edge labeling, partition the vertices into clusters to minimize the total weight of cut $\langle + \rangle$ edges and uncut $\langle - \rangle$ edges. Thus, $\langle + \rangle$ edges with large weights (representing strong correlations between endpoints) encourage those endpoints to belong to a common cluster while $\langle - \rangle$ edges with large weights encourage the endpoints to belong to different clusters. In contrast to most clustering problems, correlation clustering specifies neither the desired number of clusters nor a distance threshold for clustering; both of these parameters are effectively chosen to be the best possible by the problem definition.

Correlation clustering was introduced by Bansal, Blum, and Chawla [1], motivated by both document clustering and agnostic learning. They proved NP-hardness and gave constant-factor approximation algorithms for the special case in which the graph is complete (full information) and every edge has the same weight. We give an $O(\log n)$-approximation algorithm for the general case based on a linear-programming rounding and the "region-growing" technique. We also prove that this linear program has a gap of $\Omega(\log n)$, and therefore our approximation is tight under this approach. We also give an $O(r^3)$-approximation algorithm for $K_{r,r}$-minor-free graphs. On the other hand, we show that the problem is equivalent to minimum multicut, and therefore APX-hard and difficult to approximate better than $\Theta(\log n)$.

1 Introduction

Clustering objects into groups is a common task that arises in many applications such as data mining, web analysis, computational biology, facility location, data compression, marketing, machine learning, pattern recognition, and computer vision. Clustering algorithms for these and other objectives have been heavily investigated in the literature. For partial surveys, see e.g. [8, 14, 19, 20, 21, 25].

In a theoretical setting, the objects are usually viewed as points in either a metric space (typically finite) or a general distance matrix, or as vertices in a graph. Typical objectives include minimizing the maximum diameter of a cluster (k-clustering) [11], minimizing the average distance between pairs of clustered points (k-clustering sum) [24], minimizing the maximum distance to a "centroid object" chosen for each cluster (k-center) [11], minimizing the average distance to such a centroid object (k-median) [13], minimizing the average squared distance to an arbitrary centroid point

[*]Preliminary versions of this paper appeared in APPROX 2003 [6] and ESA 2003 [7].

[†]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 32 Vassar Street, Cambridge, MA 02139. Email: {edemaine, nickle}@theory.lcs.mit.edu. Research was supported in part by an NSF GRF.

[‡]Department of Computer Science, School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. Email: fiat@tau.ac.il, dotan@telmap.com.

(k-means) [14], and maximizing the sum of distances between pairs of objects in different clusters (maximum k-cut) [17]. These objectives interpret the distance between points as a measure of their dissimilarity: the larger the distance, the more dissimilar the objects. Another line of clustering algorithms intreprets the distance or weights between pairs of points as a measure of their similarity: the larger the weight, the more similar the objects. In this case, the typical objective is to find a k-clustering that minimizes the sum of weights between pairs of objects in different clusters (minimum k-cut) [17]. All of these clustering problems are parameterized by the desired number k of clusters. Without such a restriction, these clustering objective functions would be optimized when $k = n$ (every object is in a separate cluster) or when $k = 1$ (all objects belong to a single cluster).

In the *correlation-clustering problem* introduced by Bansal, Blum, and Chawla [1], the underlying model is that objects can be truly categorized, and we are given probabilities about pairs of objects belonging to common categories. For example, the multiset of objects might consist of all authors of English literature, and two authors belong to the same category if they correspond to the same real person. This task would be easy if authors published papers consistently under the same name. However, some authors might publish under several different names such as William Shakespeare, W. Shakespeare, Bill Shakespeare, Sir Francis Bacon, Edward de Vere, and Queen Elizabeth I [1]. Given some confidence about the similarity and dissimilarity of the names, our goal is to cluster the objects to maximize the probability of correctness. There are two main objective functions for this problem, that of minimizing disagreements and that of maximizing agreements between the input estimates and the output clustering. The decision versions of these two optimization problems are identical, and known to be NP-complete [1]. However, they differ in their approximability.

As we consider both similarity and dissimilarity measures in our problem, it is in a sense a generalization of the typical clustering objectives mentioned above. In fact, an appropriate interpretation of our problem instance suggests that our objective is a combination of the minimum k-clustering sum and minimum k-cut clustering objectives. An interesting property of our problem is that the number k of clusters is no longer a parameter of the input; there is some "ideal" k which the algorithm must find. Another clustering problem with this property is *location area design*, a problem arising in cell phone network design. As formulated by Bejerano et al. [2], this problem attempts to minimize the sum of the squared sizes of the clusters plus the weight of the cut induced by the clustering. The similarities between these two problems allow us to apply many of the same techniques.

1.1 Our Contributions

In this paper, we focus on the objective of minimizing disagreements in (weighted) general graphs. We give an $O(\log n)$-approximation algorithm for weighted general graphs and an $O(r^3)$-approximation algorithm for weighted graphs excluding the complete bipartite graph $K_{r,r}$ as a minor (e.g., graphs embeddable in surfaces of genus $\Theta(r^2)$). Our $O(\log n)$ approximation is based on rounding a linear program using the *region growing* technique introduced in the seminal paper of Leighton and Rao [17] on multicommodity max-flow min-cut theorems. Using ideas developed in Bejerano et al. [2], we are able to prove that this rounding technique yields a good approximation. Their paper also inspires our $O(r^3)$ approximation which uses a rounding technique introduced by Tardos and Vazirani [27] in a paper on max-flow min-multicut ratio and based on a lemma of Klein, Plotkin,

[1]Some scholars believe William Shakespeare was actually a pen name for one of the wealthier and more renowned contemporaries of the Elizabethan era. Common candidates for the true authorship of Shakespeare's works include Sir Francis Bacon, Edward de Vere, and even, on occasion, Queen Elizabeth I [23].

and Rao [16]. We further prove that the gap in the linear program can be $\Omega(\log n)$, and therefore our bounds are tight for any algorithm based on rounding this linear program. We also prove that our problem is APX-hard because it is equivalent to the APX-hard problem minimum multicut [5], for which the current best approximation is $O(\log k)$ for a certain parameter k [9]. Any $o(\log n)$-approximation algorithm for our problem would require improving the state-of-the-art for approximating minimum multicut. This equivalence also implies that any $O(\log k)$ algorithm for minimum multicut yields an $O(\log n)$ for correlation clustering. Our results are summarized in Table 1.

Problem class	Approximation	Hardness	Equivalence
Unweighted complete graphs	$4 \log_2 n$		
Unweighted general graphs	$O(\log n)$	APX-hard	Unweighted multicut
Weighted general graphs	$O(\log n)$	APX-hard	Weighted multicut

Table 1: Our contributions.

We remark that even for unweighted complete graphs, our (more general) algorithm has a better approximation ratio than the constant-factor approximation algorithm of [1] for $n \leq 10^{81} \approx 2^{270}$, a rough upper bound on the number of particles in the known universe. This is because their constant factor is approximately 20,000, whereas our approximation factor is smaller than $4 \log_2 n$.

Most of our results were obtained simultaneously and independently by Charikar et al. [3]. The results of their paper are detailed in the next section.

1.2 Related Work

The only previous work on the correlation-clustering problem is that of Bansal et al. [1]. Their paper considers correlation clustering in an unweighted complete graph, i.e., every pair of objects has an estimate of either “similar” or “dissimilar”. For minimizing disagreements, they give a constant-factor approximation via a combinatorial algorithm. For maximizing agreements, they give a polynomial-time approximation scheme (PTAS). The limitation of these two algorithms is that they assume the input graph is complete, while in many applications, estimate information is incomplete. Table 2 summarizes the results of their paper.

Problem class	Approximation	Hardness
Unweighted complete graphs	$c \approx 20{,}000$	NP-hard
Unweighted general graphs	Open	NP-hard
Weighted general graphs	Open	APX-hard

Table 2: Previous results on minimizing disagreements from [1].

The constant-factor algorithm of Bansal et al. [1] for unweighted complete graphs is combinatorial. It iteratively “cleans” clusters until every cluster C is *δ-clean* (i.e., for every vertex $v \in C$, v has at least $(1-\delta)|C|$ plus neighbors in C and at most $\delta|C|$ plus neighbors outside C). They bound the approximation factor of their algorithm by counting the number of “bad” triangles (triangles with two $\langle + \rangle$ edges and one $\langle - \rangle$ edge) in a δ-clean clustering and use the existence of these bad triangles to lower bound OPT. Complete graphs have many triangles, and the counting arguments for counting bad triangles rely heavily on this fact. When we generalize the problem to graphs that are not necessarily complete, bad triangles no longer form a good lower bound on OPT. It may be possible to find a combinatorial algorithm for this problem that bounds the approximation factor by counting bad cycles (cycles with exactly one minus edge). However, in this paper, we formulate the problem as a linear program and use the *region-growing* technique to round it.

Simultaneously, and independent of our work, Charikar et al. [3] gave several algorithmic and hardness results for the correlation clustering problem. They studied both the minimization and maximization variants of the problem. For minimizing disagreements, they also formulated the problem as a linear program and used the *region-growing* technique to round it. For complete graphs, they prove that this approach gives a factor 4 approximation. For general graphs, their $O(\log n)$ approximation is identical to ours. For both results, they investigate the integrality gap of the linear-programming formulation, finding examples with gaps of 2 and $\Omega(\log n)$ respectively. For maximizing agreements, the authors give a factor 0.7664 approximation in general graphs. As for hardness results, they prove APX-hardness of minimizing disagreements in complete graphs and maximizing agreements in general graphs. They also note that minimizing disaggreements in general graphs is as hard as the minimum multicut problem.

Independent of Charikar et al. [3], the paper of Swamy [26] gives a 0.7666-approximation algorithm for the problem of maximizing agreements in general graphs. Recently, Charikar and Wirth [4] studied the problem of maximizing the *correlation*, or weight of agreements minus disagreements, of a clustering. Their paper details an $\Omega(1/\log n)$ approximation for this problem.

1.3 Paper Structure

The rest of this paper is organized as follows. Section 2 formalizes the correlation-clustering problem and some necessary notation. Section 3 presents explicit approximation algorithms for the problem in general and $K_{r,r}$-minor-free graphs. Section 4 proves that the problem is equivalent to the minimum multicut problem, thus yielding an APX-hardness result and another implicit $O(\log n)$ algorithm. Section 5 discusses some applications of the correlation-clustering problem formulation. We conclude with open problems in Section 6.

2 Preliminaries

2.1 Problem Definition

Bansal, Blum, and Chawla [1] present the following clustering problem. We are given a graph on n vertices, where every edge (u, v) is labelled either $\langle + \rangle$ or $\langle - \rangle$ according to whether u and v have been deemed to be similar or dissimilar. In addition, each edge has a weight $c_e \in [0, \infty)$ which can be interpreted as a confidence measure of the similarity or dissimilarity of the edge's endpoints (higher weight denotes higher confidence).[2] The goal is to produce a partition of the vertices (a clustering) that agrees as much as possible with the edge labels (see, for example, Figure 1). More precisely, we want a clustering that maximizes the weight of agreements: the weight of $\langle + \rangle$ edges within clusters plus the weight of $\langle - \rangle$ edges between clusters. Alternatively, the clustering should minimize the weight of disagreements: the weight of $\langle - \rangle$ edges inside clusters plus the weight of $\langle + \rangle$ edges between clusters. Although the optimal solution to these two problems are the same, they differ in their approximability.

In this paper, we focus on the problem of minimizing disagreements. In the rest of this paper when we refer to the "correlation-clustering problem" or "the clustering problem" we mean the problem of minimizing disagreements. We also say "positive edge" when referring to an edge labelled $\langle + \rangle$ and "negative edge" when referring to an edge labelled $\langle - \rangle$. Note that the terminology

[2]For example, if there is a function $f(u, v)$ that outputs the probability of u and v being similar, then a natural assignment of weight to edge $e = (u, v)$ is $c_e = |\log \frac{f(u,v)}{1-f(u,v)}|$ [1].

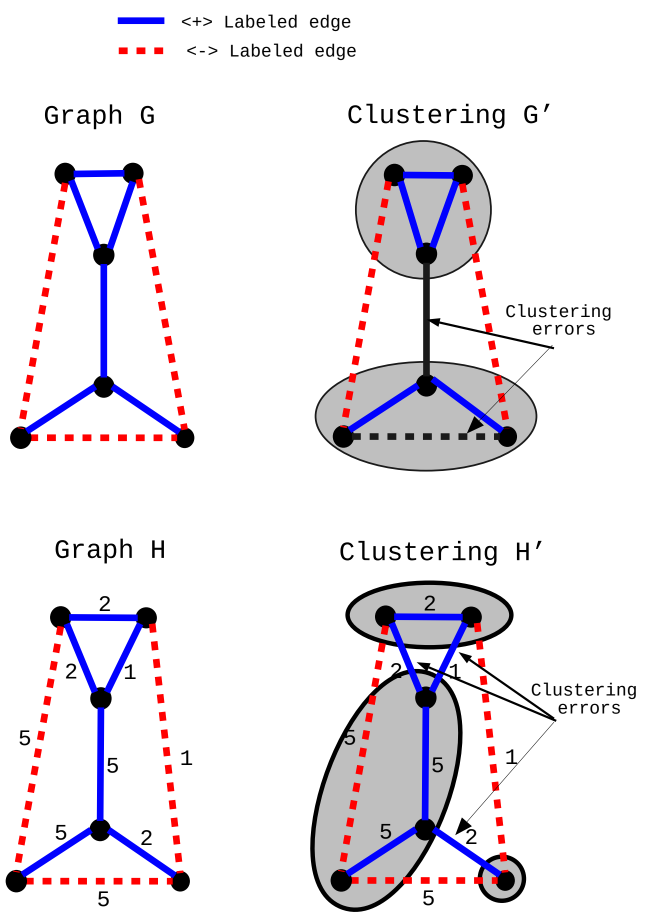

Figure 1: Two clustering examples for unweighted and weighted (general) graphs. In the unweighted case we give an optimal clustering with two errors: one error on an edge labelled $\langle + \rangle$ and one error on an edge labelled $\langle - \rangle$. For the weighted case we get a different optimal clustering with three errors on $\langle + \rangle$ edges and total weight 5.

"positive" and "negative" here refers to the edge label and not the weight; edge weights are always nonnegative regardless of the label.

2.2 Notation

Let $G = (V, E)$ be a graph on n vertices with edge weights $c_e \geq 0$. Let $e(u, v)$ denote the label $(\langle + \rangle, \langle - \rangle)$ of the edge (u, v). Let $E^{\langle + \rangle}$ be the set of positive edges and let $G^{\langle + \rangle}$ be the graph induced by $E^{\langle + \rangle}$, $E^{\langle + \rangle} = \{(u, v) \mid e(u, v) = \langle + \rangle\}$, $G^{\langle + \rangle} = (V, E^{\langle + \rangle})$. Likewise we define $E^{\langle - \rangle}$ and $G^{\langle - \rangle}$ for negative edges.

In general, for a clustering $\mathbb{S} = \{S_1, \ldots, S_k\}$, let $C(v)$ be the set of vertices in the same cluster as v. We call an edge (u, v) a positive mistake if $e(u, v) = \langle + \rangle$ and yet $u \notin C(v)$. We call an edge (u, v) a negative mistake if $e(u, v) = \langle - \rangle$ and $u \in C(v)$. The number of mistakes of a clustering $\mathbb{S}$ is the sum of positive and negative mistakes. The weight of the clustering is the sum of the weights of mistaken edges in $\mathbb{S}$. We introduce the following notation for the weight of a clustering $\mathbb{S}$:

$$\begin{aligned} w(\mathbb{S}) &= w_p(\mathbb{S}) + w_m(\mathbb{S}), \\ w_p(\mathbb{S}) &= \sum\{c_e : e = (u, v) \in E^{\langle + \rangle},\ u \notin C(v)\}, \end{aligned}$$

$$w_m(\mathbb{S}) \quad = \quad \sum\{c_e : e = (u,v) \in E^{\langle - \rangle},\ u \in C(v)\}.$$

We refer to the optimal clustering as OPT and its weight or cost as $w(\text{OPT})$. For a general set of edges $T \subset E$, we define the weight of T to be the sum of the weights in T, $w(T) = \sum_{e \in T} c_e$. For a graph $G = (V, E)$ and a set of edges $T \subset E$ we define the graph $G \setminus T$ to be the graph $(V, E \setminus T)$.

3 $O(\log n)$ Approximation

In this section, we use a linear-programming formulation of this problem to design an approximation algorithm. The algorithm first solves the linear program. The resulting fractional values are interpreted as distances between vertices; close vertices are most likely similar, far vertices are most likely dissimilar. The algorithm then uses region-growing techniques to group close vertices and thus round the fractional variables. Using ideas from Bejerano et al. [2], we are able to show that this approach yields an $O(\log n)$ approximation. We also present a modification to this approach that yields an $O(r^3)$ approximation for $K_{r,r}$-minor-free graphs.

3.1 Linear-Programming Formulation

Consider assigning a zero-one variable x_{uv} to each pair of vertices (hence $x_{uv} = x_{vu}$). When $(u, v) \in E$, we sometimes write x_{uv} as x_e where it is understood that $e = (u, v)$. Given a clustering, set $x_{uv} = 0$ if u and v are in a common cluster, and $x_{uv} = 1$ otherwise. To express $w(\mathbb{S})$ in this notation, notice that $1 - x_e$ is 1 if edge e is within a cluster and 0 if edge e is between clusters. Thus

$$w(\mathbb{S}) \quad = \quad \sum_{e \in E^{\langle - \rangle}} c_e(1 - x_e) + \sum_{e \in E^{\langle + \rangle}} c_e x_e.$$

Our goal is to find a valid assignment of x_{uv}'s to minimize this cost. An assignment of x_{uv}'s is valid (corresponds to a clustering) if $x_{uv} \in \{0, 1\}$ and the x_{uv}'s satisfy the triangle inequality.

We relax this integer program to the following linear program:

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{e \in E^{\langle - \rangle}} c_e(1 - x_e) + \sum_{e \in E^{\langle + \rangle}} c_e x_e \\ \text{subject to} & x_{uv} \in [0, 1] \\ & x_{uv} + x_{vw} \geq x_{uw} \\ & x_{uv} = x_{vu} \end{array}$$

We will round this linear program to get an $O(\log n)$ approximation for our problem. This is in fact the best approximation factor that we can hope for with this linear program because it has an $\Omega(\log n)$ integrality gap.

Theorem 3.1 *The integrality gap of the above linear program is $\Omega(\log n)$ in the worst case.*

Proof: Recall that a d-regular graph $G = (V, E)$ is a (d, c)-*expander* if, for every subset of vertices U of size at most $n/2$, $|N(U)| \geq c|U|$ where $N(U)$ is the set of vertices in $V \setminus U$ adjacent to vertices in U. Consider a (d, c)-expander G' for some constants c and d, whose existence is shown in e.g. [22]. To construct our example G, set the weight of every edge in G' to 1 and its label to $\langle + \rangle$. For each vertex v in G, add an edge of weight $nd/2$ to all neighbors u whose distance from v is at least $(\log_d(n/2) - 1)/2$. Set the label of these edges to $\langle - \rangle$. Notice since there are at most $nd/2$ edges in G', the weight of a single $\langle - \rangle$ edge is at least the total weight of all $\langle + \rangle$ edges. Therefore, the optimal clustering $\mathbb{S}$ must cut all $\langle - \rangle$ edges, and so the diameters of the resulting clusters must be at most $\log_d(n/2) - 1$. Because the vertices of G' have bounded degree d, the size of the cluster of diameter r is bounded by d^{r+1}. Therefore, the clusters of the optimal clustering have size at most $n/2$. By the expansion property of G', the number of $\langle + \rangle$ edges we must cut is at least $(c \sum_{S \in \mathbb{S}} |S|)/2 = (cn)/2$, and so the weight of the optimal clustering is $\Omega(n)$.

On the other hand, assigning $x_e = 2/(\log_d(n/2) - 1)$ for $\langle + \rangle$ edges and $x_e = 1$ for $\langle - \rangle$ edges is a feasible fractional solution of value at most $(dn/2) \times (2/(\log_d(n/2) - 1))$, and so the weight of the optimal fractional solution is $O(n/\log n)$. The theorem follows. $\square$

3.2 Rounding in General Graphs

Region growing. We iteratively grow balls of at most some fixed radius (computed according to the fractional x_{uv} values) around nodes of the graph until all nodes are included in some ball. These balls define the clusters in the final approximate solution. As high x_{uv} values hint that u and v should be in separate clusters, this approach seems plausible. The fixed radius guarantees an approximation ratio on disagreements within clusters while the region-growing technique itself guarantees an approximation ratio on disagreements between clusters.

First we present some notation that we need to define the algorithm. A *ball* $B(u, r)$ of radius r around node u consists of all nodes v such that $x_{uv} \le r$, the subgraph induced by these nodes, and the fraction $(r - x_{uv})/x_{vw}$ of edges (v, w) with only endpoint $v \in B(u, r)$. The *cut* of a set S of nodes, denoted by $\operatorname{cut}(S)$, is the weight of the positive edges with exactly one endpoint in S, i.e.,

$$\operatorname{cut}(S) = \sum_{|\{v,w\} \cap S| = 1,\ (v,w) \in E^{\langle + \rangle}} c_{vw}.$$

The *cut* of a ball is the cut induced by the set of vertices included in the ball. The *volume* of a set S of nodes, denoted by $\operatorname{vol}(S)$, is the weighted distance of the edges with both endpoints in S, i.e.,

$$\operatorname{vol}(S) = \sum_{\{v,w\} \subset S,\ (v,w) \in E^{\langle + \rangle}} c_{vw} x_{vw}.$$

Finally, the *volume* of a ball is the volume of $B(u, r)$ including the fractional weighted distance of positive edges leaving $B(u, r)$. In other words, if $(v, w) \in E^{\langle + \rangle}$ is a cut positive edge of ball $B(u, r)$ with $v \in B(u, r)$ and $w \notin B(u, r)$, then (v, w) contributes $c_{vw} \cdot (r - x_{uv})$ weight to the volume of ball $B(u, r)$. For technical reasons, we also include an initial volume I to the volume of every ball (i.e., ball $B(u, 0)$ has volume I).

Algorithm. We can now present the algorithm for rounding a fractional solution FRAC to an integral solution SOL. Suppose the volume of the entire graph is F, and thus $w_p(\text{FRAC}) = F$. Let the initial volume I of the balls defined in the algorithm be F/n, and let c be some constant which we determine later.

Algorithm Round

1. Pick any node u in G.

2. Initialize r to 0.

3. Grow r by $\min\{(d_{uv} - r) > 0 : v \notin B(u,r)\}$ so that $B(u,r)$ includes another entire edge, and repeat until $\text{cut}(B(u,r)) \leq c\ln(n+1) \times \text{vol}(B(u,r))$.
4. Output the vertices in $B(u,r)$ as one of the clusters in $\mathbb{S}$.
5. Remove vertices in $B(u,r)$ (and incident edges) from G.
6. Repeat Steps 1–5 until G is empty.

This algorithm clearly runs in polynomial time and terminates with a solution that satisfies the constraints. We must show that the resulting cost is not much more than the original fractional cost. Throughout the analysis section, we refer to the optimal integral solution as OPT. We also use $\text{FRAC}(x_{uv})$ and $\text{SOL}(x_{uv})$ to denote the fractional and rounded solution of the variable x_{uv} in the linear program.

Positive edges. The termination condition on the region-growing procedure guarantees an $O(\log n)$ approximation to the cost of positive edges (between clusters). Let $\mathcal{B}$ be the set of balls found by our algorithm in step 3. Then,

$$\begin{aligned}
w_p(\text{SOL}) &= \sum_{(u,v)\in E^{\langle+\rangle}} c_{uv}\,\text{SOL}(x_{uv}) \\
&= \tfrac{1}{2}\sum_{B\in\mathcal{B}} \text{cut}(B) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \sum_{B\in\mathcal{B}} \text{vol}(B) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \left(\sum_{(u,v)\in E^{\langle+\rangle}} c_{uv}\,\text{FRAC}(x_{uv}) + \sum_{B\in\mathcal{B}} \frac{F}{n} \right) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \big(w_p(\text{FRAC}) + F\big) \\
&\leq c\ln(n+1) \times w_p(\text{FRAC})
\end{aligned}$$

where the fourth line follows from the fact that the balls found by the algorithm are disjoint.

The rest of our analysis hinges on the fact that the balls returned by this algorithm have radius at most $1/c$. This fact follows from the following known lemma [28]:

Lemma 3.2 *For any vertex u and family of balls $B(u,r)$, the condition* $\text{cut}(B(u,r)) \leq c\ln(n+1) \times \text{vol}(B(u,r))$ *is achieved for some $r \leq 1/c$.*

Negative edges. As in Bejerano et al. [2], we can use this radius guarantee to bound the remaining component of our objective function. We see that our solution gives a $\frac{c}{c-2}$-approximation to the cost of negative edges (within clusters). Again, let $\mathcal{B}$ be the set of balls found by our algorithm in step 3. Then,

$$\begin{aligned}
w_m(\text{FRAC}) &= \sum_{(u,v)\in E^{\langle-\rangle}} c_{uv}(1 - \text{FRAC}(x_{uv})) \\
&\geq \sum_{B\in\mathcal{B}} \sum_{(u,v)\in B\cap E^{\langle-\rangle}} c_{uv}(1 - \text{FRAC}(x_{uv})) \\
&\geq \sum_{B\in\mathcal{B}} \sum_{(u,v)\in B\cap E^{\langle-\rangle}} c_{uv}(1 - 2/c)
\end{aligned}$$

$$\begin{aligned} &\geq (1-2/c)\sum_{B\in\mathcal{B}}\sum_{(u,v)\in B\cap E^{\langle-\rangle}} c_{uv} \\ &= \tfrac{c-2}{c} w_m(\mathrm{SOL}) \end{aligned}$$

where the third line follows from the triangle inequality and the $1/c$ bound on the radius of the balls. The approximation ratio $\frac{c}{c-2}$ is $O(1)$ provided $c > 2$.

Overall approximation. Combining these results, we pay a total of

$$\begin{aligned} w(\mathrm{SOL}) &= w_p(\mathrm{SOL}) + w_m(\mathrm{SOL}) \\ &\leq c\ln(n+1)\times w_p(\mathrm{OPT}) + \left(\frac{c}{c-2}\right)\times w_m(OPT) \\ &\leq \max\left\{c\ln(n+1), \frac{c}{c-2}\right\} w(\mathrm{OPT}) \end{aligned}$$

and thus we have an $O(\ln n)$ approximation, where the lead constant, c, is just slightly larger than 2.

3.3 Rounding in $K_{r,r}$-Minor-Free Graphs

In $K_{r,r}$-minor-free graphs, we can use a theorem of Klein et al. [16] to round our linear program in a way that guarantees an $O(r^3)$ approximation to the cost of disagreements between clusters. The clusters produced by this rounding have radius at most $1/c$, and thus the rest of the results from the previous section follow trivially. The theorem states that, in graphs with unit-length edges, there is an algorithm to find a "small" cut such that the remaining graph has "small" connected components:

Theorem 3.3 [16] *In a graph G with weight u on the edges which satisfy the triangle inequality, one can find in polynomial time either a $K_{r,r}$ minor or an edge cut of weight $O(rU/\delta)$ whose removal yields connected components of weak diameter*[3] *$O(r^2\delta)$ where U is the total weight of all edges in G.*

As in the case of the region-growing technique, this theorem allows us to cluster the graph cheaply, subject to some radius guarantee. As this clustering cost is independent of n, this technique is typically applied in place of the region-growing technique to get better approximations for $K_{r,r}$-minor-free graphs. (See, for example, Tardos and Vazirani [27] or Bejerano et al. [2].) In particular, this implies constant-factor approximations for various clustering problems on planar graphs.

The idea is as follows. Given a $K_{r,r}$-minor-free graph G with weights c_e and edge lengths x_e as defined by the linear program, we subdivide each edge $e \in E^{\langle+\rangle}(G)$ into a chain of $\lceil kx_e \rceil$ edges of the same weight c_e for some appropriate k, yielding a new graph G'. (Note that we do not include the negative edges of G in G'.) We apply Theorem 3.3 to G', getting an edge cut F' which maps to an edge cut F in G of at most the same weight. This creates the natural correspondence between the resulting components of G' and G. Note two nodes at distance d in G are at distance at least kd in G'. Hence, if we take $\delta = O(k/(cr^2))$, for a sufficient setting of the constants the components in G will have diameter at most $2/c$.

For any constant $c > 2$, we can bound the weight of the negative mistakes as in Section 3.2 by $(\frac{c}{c-2})w_m(FRAC)$. The weight of the positive mistakes is simply the weight of the cut F', and so,

[3]The *weak diameter* of a connected component in a modified graph is the maximum distance between two vertices in that connected component as measured in the original graph. For our purposes, distances are computed according to the $x_{u,v}$ which satisfy the triangle inequality and are defined on all pairs of vertices, so the weak diameter equals the diameter.

by Theorem 3.3, we just need to bound the total weight U' of the graph G'. Let $U = \sum_{e \in E^{\langle + \rangle}(G)} c_e$ be the total weight of positive edges in G and recall $\mathrm{vol}(G) = \sum_{e \in E^{\langle + \rangle}(G)} c_e x_e$. Then

$$\begin{aligned} U' &= \sum_{e \in G'} c_e \\ &= \sum_{e \in E^{\langle + \rangle}(G)} \lceil k x_e \rceil c_e \\ &\leq \sum_{e \in E^{\langle + \rangle}(G)} (k x_e + 1) c_e \\ &= k \, \mathrm{vol}(G) + U. \end{aligned}$$

By Theorem 3.3, the weight of F, which equals the weight of F', is $O(rU'/\delta) = O(r^3(k \, \mathrm{vol}(G) + U)/k)$. Taking $k = U/\mathrm{vol}(G)$, this becomes $O(r^3 \, \mathrm{vol}(G))$ and is thus an $O(r^3)$ approximation of the cost of disagreements between clusters, as desired. The size of G' may be pseudopolynomial in the size of G. However, the algorithm of Klein et al. [16] consists of r breath-first searches of G', and these can be implemented without explicitly subdividing G. Thus, the algorithm runs in polynomial time.

4 Relation to Multicut

In this section, we prove that the correlation-clustering problem is equivalent to the multicut problem, thus yielding another $O(\log n)$ approximation for our problem as well as some hardness results.

Definition 4.1 *The* weighted multicut problem *is the following problem: Given an undirected graph G, a weight function w on the edges of G, and a collection of k pairs of distinct vertices (s_i, t_i) of G, find a minimum weight set of edges of G whose removal disconnects every s_i from the corresponding t_i.*

The multicut problem was first stated by Hu in 1963 [12]. For $k = 1$, the problem coincides of course with the ordinary min-cut problem. For $k = 2$, it can be also solved in polynomial time by two applications of a max flow algorithm [31]. The problem was proven NP-hard and APX-hard for any $k \geq 3$ in by Dahlhaus, Johnson, Papadimitriou, Seymour, and Yannakakis [5]. The best known approximation ratio for weighted multicut in general graphs is $O(\log k)$ [9] . For planar graphs, Tardos and Vazirani [27] give an approximate max-flow min-cut theorem and an algorithm with a constant approximation ratio. For trees, Garg, Vazirani, and Yannakakis give an algorithm with an approximation ratio of 2 [10].

4.1 Reduction from Correlation Clustering to Weighted Multicut

We now show that finding an optimal clustering is equivalent to finding a minimum-weight edge set that covers all erroneous cycles. An *erroneous cycle* is a simple cycle containing exactly one negative edge. An edge *covers* a cycle if the edge is contained in the cycle, i.e., removing the edge breaks the cycle. Figure 2 illustrates the equivalence.

Guided by this observation, we define a multicut problem derived from our original graph by replacing the negative edges with source-sink pairs (and some other required changes). We show that a solution to the newly formed multicut problem induces a solution to the clustering problem

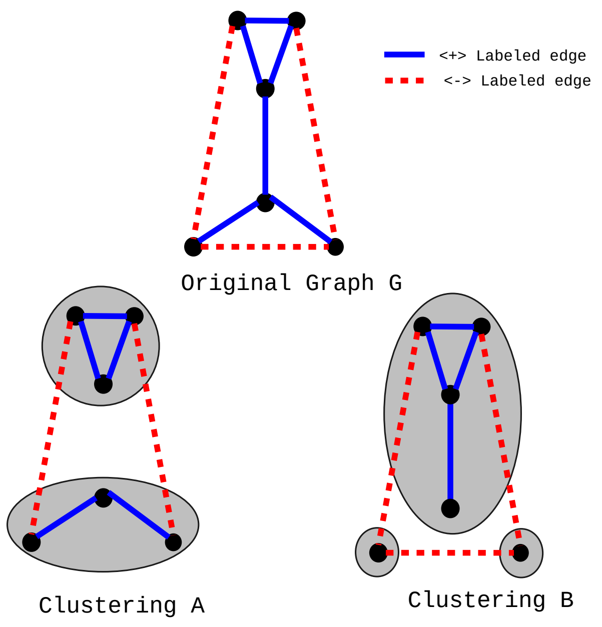

Figure 2: Two optimal clusterings for G. For both of these clusterings we have removed two edges (different edges) so as to eliminate all the erroneous cycles in G. After the edges were removed every connected component of $G^{\langle + \rangle}$ is a cluster. Note that the two clusterings are consistent; no positive edges connect two clusters and no negative edges connect vertices within the same cluster.

with the same weight, and that optimal solution to the multicut problem induces an optimal solution to the clustering problem.

These reductions imply that the $O(\log k)$-approximation algorithm for the multicut problem [9] induces an $O(\log n)$-approximation algorithm for the correlation-clustering problem. We prove this for weighted general graphs, which implies the same result for unweighted general graphs. We start by proving two simple lemmata. We call a clustering a *consistent clustering* if it contains no mistakes, i.e., if there are no cut positive edges or uncut negative edges.

Lemma 4.2 *A graph contains no erroneous cycles if and only if it has a consistent clustering.*

Proof: Let G be a graph with no erroneous cycles, and let $\mathbb{S}$ be the set of connected components of $G^{\langle + \rangle}$. We argue that $\mathbb{S}$ is a consistent clustering. Assume that (u, v) is a negative mistake in $\mathbb{S}$, i.e., $e(u, v) = \langle - \rangle$ and $u \in C(v)$. Because u and v belong to the same cluster, they are on the same connected component of $G^{\langle + \rangle}$. It follows that there is a simple path of positive edges from u to v. This path, combined with the negative edge (u, v), creates an erroneous cycle. From the construction of $\mathbb{S}$ (the connected components of $G^{\langle + \rangle}$) it is obvious that $\mathbb{S}$ contains no positive mistakes either, and so $\mathbb{S}$ is a consistent clustering.

Let $\mathbb{S}$ be a consistent clustering on a graph G. We claim that G contains no erroneous cycle. Assume $(v_1, v_2, \ldots, v_k)$ is erroneous cycle of size k. Without loss of generality we let (v_1, v_2) be the negative edge in the cycle. If $v_1 \in C(v_2)$ then we have a negative mistake which means $\mathbb{S}$ is not consistent. If $v_1 \notin C(v_2)$ then there must exist a point $1 \leq i \leq k-1$ along the cycle such that $v_i \notin C(v_{i+1})$ which in turn means that the edge (v_i, v_{i+1}) is a positive mistake. $\square$

Lemma 4.3 *Let F be a set of edges of minimum weight such that $G \setminus F$ contains no erroneous cycles and T be a set of edges on which an optimal clustering makes mistakes. Then the weight of F equals the weight of T.*

Proof: Let $\mathbb{S}$ be a clustering on G, and let T be the set of mistakes made by this clustering. Then $\mathbb{S}$ is a consistent clustering on $G \setminus T$, and so the result follows from Lemma 4.2. $\square$

We give a reduction from the problem of correlation clustering to the weighted multicut problem. The reduction translates an instance of unweighted correlation clustering into an instance of unweighted graph multicut, and an instance of weighted correlation clustering into an instance of weighted graph multicut. Refer to Figure 3 for an example.

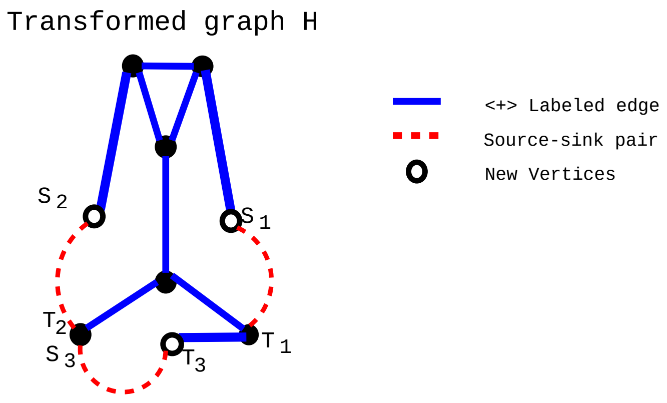

Figure 3: The original graph from Figure 2 after the transformation.

Given a weighted graph G whose edges are labelled $\{\langle + \rangle, \langle - \rangle\}$, we construct a new graph H_G and a collection of source-sink pairs $S_G = \{\langle s_i, t_i \rangle\}$ as follows:

- For every negative edge $(u, v) \in E^{\langle - \rangle}$ we introduce a new vertex $v_{\widehat{u,v}}$, a new edge $(v_{\widehat{u,v}}, u)$ with weight equal to that of (u, v), and a source-sink pair $\langle v_{\widehat{u,v}}, v \rangle$.
- Let V_{new} denote the set of new vertices, E_{new}, the set of new edges, and S_G, the set of source-sink pairs. Let $V^{'} = V \cup V_{\text{new}}$, $E^{'} = E^{\langle + \rangle} \cup E_{\text{new}}$, $H_G = (V^{'}, E^{'})$. The weight of the edges in $E^{\langle + \rangle}$ remains unchanged. We now have a multicut problem on (H_G, S_G).

We claim that any solution to the multicut problem implies a solution to the correlation-clustering problem with the exact same value, and that an approximate solution to the former gives an approximate solution to the latter.

Theorem 4.4 *(H_G, S_G) has a cut of weight W if and only if G has a clustering of weight W, and we can easily construct one from the other. In particular, the optimal clustering in G of weight W implies an optimal multicut in (H_G, S_G) of weight W and vice versa.*

The proof follows from the following two lemmas:

Lemma 4.5 *Let $\mathbb{S}$ be a clustering on G with weight W. Then there exists a multicut T' in (H_G, S_G) with weight W.*

Proof: Let $\mathbb{S}$ be a clustering of G with weight W, where T is the set of mistakes made by $\mathbb{S}$ ($w(T) = W$). Let $T' = (T \cap G^{\langle + \rangle}) \cup \{(v_{\widehat{u,v}}, u) \mid (u, v) \in T \cap G^{\langle - \rangle}\}$. Note that $w(T) = w(T')$. We now argue that T' is a multicut. Assume not; then there exists a pair $(v_{\widehat{u,v}}, v) \in S_G$ and a path from $v_{\widehat{u,v}}$ to u that contains no edge from T'. By construction, this implies that there exists a path from u to v in $G^{\langle+\rangle} \setminus T$ which, together with the negative edge (u, v), forms an erroneous cycle in $G \setminus T$. This is a contradiction since, by definition of T, $G \setminus T$ has a clustering with no mistakes. Note that the proof is constructive. □

Lemma 4.6 *If T' is a multicut in (H_G, S_G) of weight W, then there exists a clustering $\mathbb{S}$ in G of weight W.*

Proof: We construct a set T from the cut T' by replacing all edges in T' that are in E_{new} with the corresponding negative edges in G, and define a clustering $\mathbb{S}$ by taking every connected component of $G^{\langle+\rangle} \setminus T$ as a cluster. Note that T has the same total weight as T'. We argue that T is precisely the set of mistakes of $\mathbb{S}$, and thus the weight of $\mathbb{S}$ is W. In otherwords, it suffices to prove that $\mathbb{S}$ is a consistent clustering on $G \setminus T$. Assume not; then there exists an erroneous cycle in $G \setminus T$ (Lemma 4.2). Let (u, v) be the negative edge along this cycle. Then it follows that there is a path from $v_{\widehat{u,v}}$ to v in H_G which does not traverse any edge in T'. But the pair $\langle v_{u,v}, v\rangle$ are a source-sink pair which is in contradiction to T' being a multicut. □

We have presented a mapping f from an instance x of the correlation-clustering problem to an instance $f(x)$ of the multicut problem. We have also presented a mapping g from a feasible solution of $f(x)$ (multicut problem) to a feasible solution of x (the clustering problem) such that the value of the solution does not change. This means that if s is a solution to the multicut problem $f(x)$ such that $w(s) = O(\alpha \cdot w(\text{OPT(Multicut)}))$ then $g(s)$ is a solution to the clustering problem such that $w(g(s)) = \alpha \cdot w(\text{OPT(Clustering)})$. We can now use the approximation algorithm of [9] to get an $O(\log k)$ approximation solution to the multicut problem (k is the number of source-sink pairs) which translates into an $O(\log |E^{\langle-\rangle}|) \leq O(\log n^2) = O(\log n)$ solution to the clustering problem. Note that this result holds for both weighted and unweighted graphs and that the reduction of the unweighted correlation-clustering problem results in a multicut problem with unit capacities and demands.

4.2 Reduction from Multicut to Correlation Clustering

In the previous section we showed that every correlation-clustering problem can be presented (and approximately solved) as a multicut problem. We now show that the opposite is true as well, that every instance of the multicut problem can be transformed to an instance of a correlation-clustering problem, and that transformation has the following properties: any solution to the correlation-clustering problem induces a solution to the multicut problem with lower or equal weight, and an optimal solution to the correlation-clustering problem induces an optimal solution to the multicut problem.

In the previous section we could use one reduction for the weighted version and the unweighted version. Here we present two slightly different reductions from unweighted multicut to unweighted correlation clustering and from weighted multicut to weighted correlation clustering.

4.2.1 Reduction from Weighted Multicut to Weighted Correlation Clustering

Given a multicut problem instance: an undirected graph H, a weight function w on the edges of H, $w : E \to \mathcal{R}^+$, and a collection of k pairs of distinct vertices $S = \{\langle s_i, t_i\rangle, \ldots, \langle s_k, t_k\rangle)\}$ of H, we construct a correlation-clustering problem as follows:

- We start with $G_H = H$, all edge weights are preserved and all edges labelled $\langle+\rangle$.
- In addition, for every source-sink pair $\langle s_i, t_i\rangle$ we add to G_H a negative edge $e_i = (s_i, t_i)$ with weight $w(e_i) = \sum_{e \in H} w(e) + 1$.

Our transformation is polynomial, adds at most $O(n^2)$ edges, and increases the largest weight in the graph by a multiplicative factor of at most $O(n^2)$.

Theorem 4.7 *A clustering on G_H with weight W induces a multicut on (H, S) with weight at most W. Similarly, a multicut in (H, S) with weight W induces a clustering on G_H with weight W.*

Proof: Let $\mathbb{S}$ be a clustering on G_H of weight W. If $\mathbb{S}$ contains no negative mistakes, then the set of positive mistakes T is a multicut on H of weight W. If $\mathbb{S}$ contains a negative mistake, say (u, v), take one of the endpoints (u or v) and place it in a cluster of its own, thus eliminating this mistake. Because every negative edge has weight at least the sum of all positive edges, the gain by splitting the cluster exceeds the loss introduced by new positive mistakes. Therefore the new clustering $\mathbb{S}'$ on G has weight $W' < W$, and contains no negative mistakes. Thus, the set of positive mistakes T' is a multicut on H of weight at most W.

Now let T denote a multicut in (H, S) of weight W and define clustering $\mathbb{S}$ on G_H as the connected components of $G^{\langle + \rangle} \setminus T$. Note that $\mathbb{S}$ contains no negative mistakes and so has weight $w(T) = W$. □

Corollary 4.8 *An optimal clustering in G_H induces an optimal multicut in (H, S).*

Proof: This follows directly from the statements in the above lemma. □

4.2.2 Reduction from Unweighted Multicut to Unweighted Correlation Clustering

Given an unweighted multicut problem instance: an undirected graph H and a collection of k pairs of distinct vertices $S = \{\langle s_i, t_i \rangle, \ldots, \langle s_k, t_k \rangle\}$ of H, we construct an unweighted correlation-clustering problem as follows; refer to Figure 4.

- For every v such that $\langle v, u \rangle \in S$ or $\langle u, v \rangle \in S$ (i.e., v is either a source or a sink), we add $n-1$ new vertices and connect those vertices and v in a clique with positive edges (weight 1). We denote this clique by Q_v.
- For every pair $\langle s_i, t_i \rangle \in S$, we connect all vertices of Q_{s_i} to t_i and all vertices of Q_{t_i} to s_i using edges labelled $\langle - \rangle$.
- Other vertices of H are added to the vertex set of G_H, and edges of H are added to the edge set of G_H and labelled $\langle + \rangle$.

Our goal is to emulate the previous argument for weighted general graphs in the context of unweighted graphs. We do so by replacing the single edge of high weight with many unweighted negative edges. Our transformation is polynomial time, adds at most n^2 vertices and at most $O(n^3)$ edges.

Lemma 4.9 *Given a clustering C on G, we can construct another clustering C' on G such that C' is pure and $w(C') \leq w(C)$.*

Proof: For every Q_v that is split amongst two or more cluster we take all vertices of Q_v to form a new cluster. By doing so we may be adding up to $n-1$ new mistakes, (positive mistakes, positive edges adjacent to v in original graph). Merging these vertices into one cluster component reduces the number of errors by at least $n-1$.

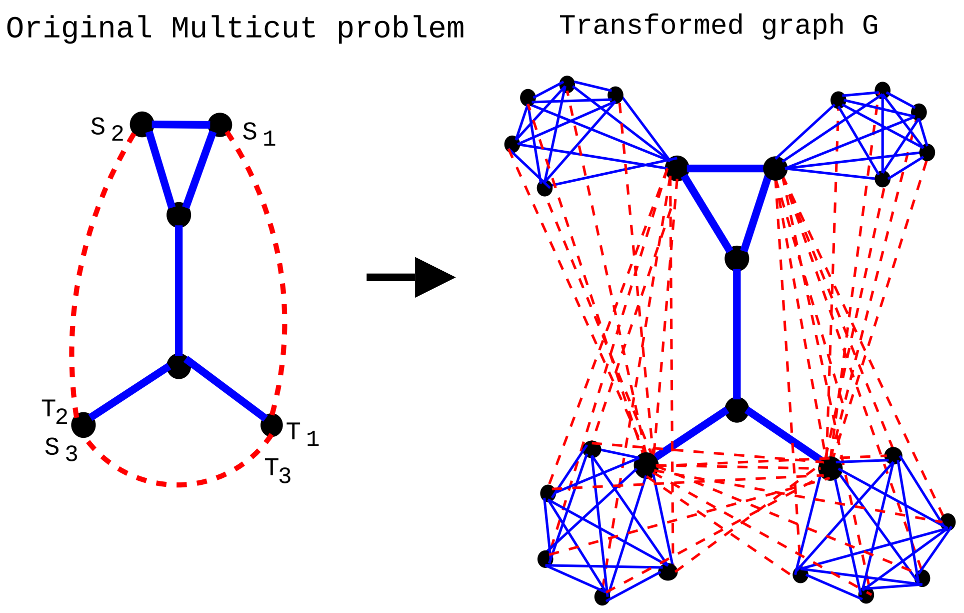

Figure 4: Transformation from the unit-capacity multicut problem (on the left) to the unweighted correlation-clustering problem (on the right).

If two Q_v and Q_w are in the same cluster component, we can move one of them into a cluster of its own. As before, we may be introducing as many as $n-1$ new positive mistakes but simultaneously eliminating $2n$ negative mistakes. $\square$

Theorem 4.10 *A clustering on G_H with weight W induces a multicut on (H, S) with weight $\leq W$. An optimal clustering in G of weight W induces an optimal multicut for (H, S) of weight W.*

Proof: We call a clustering *pure* if all vertices that belong to the same Q_v are in the same cluster, and that if $\langle v, w\rangle \in S$ then Q_v and Q_w are in different clusters. The following proposition states that we can "fix" any clustering to be a pure clustering without increasing its weight.

Given a clustering C on G_H we first "fix" it using the technique of Lemma 4.9 to obtain a pure clustering C'. Any mistake for pure clustering must be a positive mistake, the only negative edges are between clusters.

Let T be the set of positive mistakes for C', we now show that T is a multicut on (H, S). No source-sink pair are in the same cluster because the clustering is pure and removing the edges of T disconnects every source/sink pair. Thus, T is a multicut for (H, S).

Let OPT be the optimal clustering on G. Without loss of generality, we can assume that OPT is pure (otherwise, by Lemma 4.9, we can construct another clustering with no higher cost) and therefore induces a multicut on (H, S). Let T denote the minimum multicut in (H, S). T induces a pure-clustering on G as follows: take the connected component of $G^{\langle + \rangle} \setminus T$ as clusters and for every terminal $v \in S$ add every node in Q_v to the cluster containing vertices v. It can be easily seen that this gives a pure clustering, and that the only mistakes on the clustering are the edges in T. The result follows. $\square$

4.3 Remarks

The two-way reduction we just presented proves that the correlation-clustering problem and the multicut problem are essentially identical. This reduction also allows us to transfer hardness-of-approximation results from one problem to the other. Because the multicut problem is APX-hard and remains APX-hard even in the unweighted case, the unweighted correlation-clustering problem is also APX-hard.

An interesting observation comes from the constant-factor approximation for the correlation clustering problem on the unweighted complete graph from [1]. This result implies that the corresponding unweighted multicut problem, in which every pair $\langle u, v \rangle$ of nodes is either an edge or a source-sink pair, has a constant-factor approximation.

On the other hand, correlation-clustering problems where $G^{\langle + \rangle}$ is a planar graph or has a tree structure has a constant-factor approximation (as follows from [27, 10]).

5 Use of Correlation Clustering

In this section we motivate correlation clustering by describing several clustering situations where it is useful.

5.1 When We Have Attraction/Rejection Data

The correlation-clustering approach gives us a natural and unique way to solve problems that have more than one distance measure or problems that need to balance between two possible contradicting measures. For example, suppose we have a set of guests, each of whom has preferences for people they would like to sit with and for people they would prefer to avoid. We must group the guests into tables in a way that enhances amicability of the atmosphere. Using the correlation-clustering setting we can solve this problem and get a provable approximation bound.

5.2 As a Way of Solving Constraint-Clustering Problems

There has been a great deal of work on solving a clustering problem subject to constraints [18, 29, 30, 15]. In a constraint clustering problem we have the following inputs:

1. A set of objects, a distance measure between them, and an objective function to be minimized (the traditional clustering setting)
2. A set of pairwise constraints among the objects.

One seeks a clustering that minimizes the objective function and satisfies the set of constraints simultaneously. One common approach to address constraint-clustering problems is to use a clustering/search algorithm (e.g., k-means [18], k-median, hierarchal clustering) in the space of all feasible clusterings (those respecting the constraint). This approach treats the constraints as hard and consistent (must and can be simultaneously satisfied), and assumes that one can search through the space of feasible constraints [29, 30, 15].

We suggest a different approach: instead of treating a constraint-clustering problem as an unconstrained clustering problem in a trimmed search space, we look at the problem as a soft constraint satisfaction problem. We convert the distance between any two items into a constraint, combine this set of constraints with the original set of constraints and solve the resulting constraint satisfaction problem. Restructuring the problem as a constant satisfaction problem allows us to use the correlation-clustering algorithm and get a provable approximation result as opposed to the search heuristics used in previous constrained clustering algorithms.

5.3 As a Way to Determine the Number of Clusters

Traditional clustering methods use positive distance or similarity measures and seek to minimize the distance or maximize the similarity of elements within the same cluster. To avoid the problem of converging to the trivial solutions of putting all elements in the same cluster (maximize similarity) or putting each element in a cluster of his own (minimize distances within a cluster) one usually limits the number of clusters. (E.g., k-means, k-median, etc).

Correlation clustering suggests a different approach. Given a distance measure, we set a threshold, where all distances below the threshold represent attraction (the elements are close, they should be in the same cluster) and distance above the threshold represents rejection (the elements are far apart, they should not be in the same cluster). We create a graph were each item is a node and between any two elements we have an edge. We label these edges $\langle + \rangle$ if the distance between the elements is below the threshold and $\langle - \rangle$ if it is above the threshold. We set the weight of the edge according to the original distance between the elements; small distances become a positive labelled edges with large weight, whereas large distance become negative labelled edges of large weight. We now solve the correlation-clustering problem on the resulting graph.

One advantage of this approach is that choosing a threshold can be much more meaningful then choosing the number of clusters. The threshold is a property of the distance measure; it is just our observation of "near" and "far". It is independent of the actual data, and therefore, unlike an arbitrary number of clusters, k, may have a "real" meaning. Another possible advantage of this approach is that the correlation-clustering algorithm can also handle outliers easily. Outliers will usually get a cluster of their own, and when the algorithm ends we can simply discard small isolated clusters.

5.4 When We Have Only Partial Pairwise Measures

Most k-based algorithms assume points are positioned in a metric space, or at least that we have a known weight for every pair of items. This is usually not the case in real world problems. When using the correlation-clustering framework we can work with any subset of pairwise distance measure and just omit the unknown relations from the graph.

6 Conclusion

In this paper, we have investigated the problem of minimizing disagreements in the correlation-clustering problem. We gave an $O(\log n)$ approximation for general graphs, an $O(r^3)$ approximation for $K_{r,r}$-minor-free graphs, and showed that the natural linear-programming formulation upon which these algorithms are based has a gap of $\Omega(\log n)$ in general graphs. We also showed that this problem is equivalent minimum multicut, yielding another implicit $O(\log n)$ approximation as well as an APX-hardness result.

A natural extension of this work would be to improve the approximation factor for minimizing disagreements. Given our hardness result and the history of the minimum-multicut problem, this goal is probably very difficult. Another option is to improve the lower bound, but for the same reason, this goal is probably very difficult. On the other hand, one might try to design an alternate $O(\log n)$-approximation algorithm that is combinatorial, perhaps by counting erroneous cycles in a cycle cover of the graph.

Another interesting direction is to explore other objective functions of the correlation-clustering problem. Bansal et al. [1] give a polynomial-time approximation scheme (PTAS) for maximizing agreements in unweighted complete graphs. For weighted general graphs, one of two trivial clusterings (every vertex in a distinct cluster or all vertices in a single cluster) is a 1/2 approximation. Recently, Swamy [26] gave a 0.7666-approximation algorithm for the unweighted general graphs based on semidefinite programming. It would be interesting to obtain better approximations, in particular a PTAS, for general graphs. Bansal et al. [1] also mention the objective of maximizing agreements minus disagreements. This objective is of particular practical interest. However, there are no known approximation algorithms for this objective, even for complete graphs.

Finally, it would be interesting to apply the techniques presented here to other problems. The region-growing technique and Klein et al. [16] rounding technique both provide a radius guarantee on the output clusters. Many papers have used this radius guarantee to demonstrate that the solution is *feasible*, i.e., satisfies the constraints. Inspired by Bejerano et al. [2], we also use the radius guarantee to *bound the approximation factor*. This idea might be applicable to other problems.

Acknowledgements

Many thanks go to Avrim Blum, Shuchi Chawla, Mohammad Mahdian, David Liben-Nowell, and Grant Wang for helpful discussions. Many results in this paper were inspired by conversations with Seffi Naor.

References

[1] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 238–250, Vancouver, Canada, November 2002.

[2] Yigal Bejerano, Nicole Immorlica, Joseph Naor, and Mark Smith. Efficient location area planning for personal communication systems. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, pages 109–121, San Diego, CA, September 2003.

[3] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 524–533, 2003.

[4] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck's inequality. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 54–60, 2004.

[5] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 4(23):864–894, 1994.

[6] Erik D. Demaine and Nicole Immorlica. Correlation clustering with partial information. In *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 1–13, Princeton, NJ, August 2003.

[7] Dotan Emanuel and Amos Fiat. Correlation clustering — minimizing disagreements on arbitrary weighted graphs. In *Proceedings of the 11th Annual European Symposium on Algorithms*, pages 208–220, 2003.

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. Clustering for mining in large spatial databases. *KI-Journal*, 1:18–24, 1998. Special Issue on Data Mining. ScienTec Publishing.

[9] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

[10] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, 1997.

[11] Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *Journal of the ACM*, 33:533–550, 1986.

[12] T. C. Hu. Multicommodity network flows. *Operations Research*, 11:344–360, 1963.

[13] Kamal Jain and Vijay V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 2–13, 1999.

[14] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):881–892, July 2002.

[15] Dan Klein, Sepandar D. Kamvar, and Christopher D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the 19th International Conference on Machine Learning*, pages 307–314, 2002.

[16] Philip N. Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, pages 682–690, 1993.

[17] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999.

[18] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Symposium on Math, Statistics, and Probability*, pages 281–297, 1967.

[19] Marina Meila and David Heckerman. An experimental comparison of several clustering and initialization methods. *Conference on Uncertainty in Artificial Intelligence*, pages 386–395, 1998.

[20] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983.

[21] Cecilia M. Procopiuc. Clustering problems and their applications. Department of Computer Science, Duke University. `http://www.cs.duke.edu/~magda/clustering-survey.ps.gz`.

[22] Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Electronic Colloquium on Computational Complexity (ECCC)*, 8(18), 2001.

[23] Michael Satchell. Hunting for good Will: Will the real Shakespeare please stand up? *U.S. News and World Report*, pages 71–72, July 24 2000.

[24] Leonard J. Schulman. Clustering for edge-cost minimization. *Electronic Colloquium on Computational Complexity (ECCC)*, 6(035), 1999.

[25] Michael Steinbach, George Karypis, and Vipin Kumar. A comparison of document clustering techniques. *KDD-2000 Workshop on Text Mining*, 2000.

[26] Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 526–527, New Orleans, LA, 2004. Society for Industrial and Applied Mathematics.

[27] Eva Tardos and Vijay V. Vazirani. Improved bounds for the max-flow min-multicut ratio for planar and $K_{r,r}$-free graphs. *Information Processing Letters*, 47(2):77–80, 1993.

[28] Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, 2001.

[29] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1103–1110, 2000.

[30] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained K-means clustering with background knowledge. In *Proceedings of the 18th International Conference on Machine Learning*, pages 577–584, 2001.

[31] Mihalis Yannakakis, Paris C. Kanellakis, Stavros C. Cosmadakis, and Christos H. Papadimitriou. Cutting and partitioning a graph after a fixed pattern. In *Proceedings of the 10th International Colloquium on Automata, Languages and Programming*, pages 712–722, 1983.