Correlation Clustering with Partial Information

Erik D. Demaine and Nicole Immorlica*

Laboratory for Computer Science, MIT, Cambridge, MA 02139, USA.
`edemaine, nickle@theory.lcs.mit.edu`.

Abstract. We consider the following general *correlation-clustering problem* [1]: given a graph with real edge weights (both positive and negative), partition the vertices into clusters to minimize the total absolute weight of cut positive edges and uncut negative edges. Thus, large positive weights (representing strong correlations between endpoints) encourage those endpoints to belong to a common cluster; large negative weights encourage the endpoints to belong to different clusters; and weights with small absolute value represent little information. 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 weight $+1$ or -1. 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 APX-hard, and any $o(\log n)$-approximation would require improving the best approximation algorithms known for minimum multicut.

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. [6, 11, 14, 15, 16, 18].

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) [8], minimizing the average distance between pairs of clustered

* Research was supported in part by an NSF GRF.

this is simply the standard minimum-cut problem. For $k = 2$, Yannakakis et al. [21] gave a polynomial time algorithm. For $k \geq 3$, this problem was shown to be APX-hard by Dahlhaus et al. [4]. Currently, the best known approximation for this problem in general graphs is an $O(\log k)$ approximation algorithm by Garg et al. [7]. For graphs excluding any $K_{r,r}$ minor, Tardos and Vazirani [19] use a lemma of Klein et al. [12] to provide an $O(r^3)$ approximation (and thus constant approximation for planar graphs).

The only previous work on the correlation-clustering problem is that of Bansal et al. [1]. Their paper considers correlation clustering in a complete graph with all edges assigned weights from $\{+1, -1\}$, representing that every pair of objects has an estimate of either "similar" or "dissimilar". They address two main objective functions, minimizing disagreements and maximizing agreements between the input estimates and the output clustering. The decision versions of these two optimization problems are identical and shown to be NP-complete. For minimizing disagreements, they give a constant-factor approximation via a combinatorial algorithm. For maximizing agreements, they give a PTAS. Both algorithms assume the input graph is complete. However, in many applications, estimate information is incomplete.

In this paper, we consider minimizing disagreements in general graphs and with arbitrary weights. We give an $O(\log n)$-approximation algorithm for general graphs and an $O(r^3)$-approximation algorithm for 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 linear programming, rounding, and the "region growing" technique [13, 7]. Using ideas developed in Bejerano et al. [2], we are able to prove that this rounding technique yields a good approximation. We then use a lemma of Klein et al. [12] to extend our results to an $O(r^3)$ approximation for $K_{r,r}$-minor-free graphs [19, 2]. 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 as hard as the APX-hard problem minimum multicut [4], for which the current best approximation is $O(\log k)$ for a certain parameter k [7]. Any $o(\log n)$-approximation algorithm for our problem would require improving the state-of-the-art for approximating minimum multicut.

Almost simultaneously, two groups of researchers independently obtained results similar to this paper. Charikar et al. [3] and Emanuel and Fiat [5] both give $O(\log n)$ approximations for the minimization version and approximation-preserving reductions from minimum multicut, as we do. In addition, Charikar et al. [3] improve the Bansal et al. [1] result for complete graphs and give a constant factor approximation for the maximization version in general graphs. Emanuel and Fiat [5] also prove the equivalence of this problem with the minimum-multicut problem.

The rest of this paper is organized as follows. Section 2 formalizes the correlation-clustering problem, the objective of minimizing disagreements, and presents the linear-programming formulation. Section 3 demonstrates a rounding technique that yields an $O(\log n)$ approximation for this linear program in general points (k-clustering sum) [17], minimizing the maximum distance to a "centroid object" chosen for each cluster (k-center) [8], minimizing the average distance to such a centroid object (k-median) [10], minimizing the average squared distance to an arbitrary centroid point (k-means) [11], and maximizing the sum of distances between pairs of objects in different clusters (maximum k-cut) [13]. 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 interprets the distance between points as a measure of their similarity: the larger the distance, the more similar the objects. In this case, the typical objective is to find a k-clustering that minimizes the sum of distances between pairs of objects in different clusters (minimum k-cut) [13]. 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 et al. [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. Given some confidence about the similarity and dissimilarity of the names, our goal is to cluster the objects to maximize the probability of correctness. As we consider both similarity and dissimilarity measures, our objective 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 sizes squared of the clusters plus the weight of the cut induced by the clustering. The authors provide an $O(\log n)$ approximation for this problem in general graphs using region-growing techniques and an $O(r^3)$ approximation in $K_{r,r}$-minor-free graphs using a lemma of Klein et al. [12]. The similarities between these two problems allow us to apply many of the same techniques.

For our lower bounds, we exploit the similarities between the correlation-clustering problem and the minimum-multicut problem, introduced by Hu [9]. In the minimum-multicut problem, we are given an edge-weighted graph and a list of k pairs of vertices, and the goal is to remove edges of minimum total weight such that the resulting graph disconnects all k input pairs. For $k = 1$, graphs. Section 4 considers the special case of $K_{r,r}$-minor-free graphs and uses an alternate rounding technique to get an $O(r^3)$ approximation in these graphs. In Section 5, we prove lower bounds, establishing APX-hardness and a logarithmic gap in the linear program. We conclude with open problems in Section 6.

2 Problem Definition and Linear-Programming Formulation

An instance of the correlation-clustering problem is an undirected graph $G = (V, E)$ with edge weights $c_e \in (-\infty, +\infty)$ for each $e \in E$. Each edge weight can be interpreted as a confidence measure of the similarity or dissimilarity of the edge's endpoints. 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]. Hence, an edge $e = (u, v)$ of weight $c_e > 0$ corresponds to a belief that nodes u and v are similar. Larger c_e indicate higher confidence in this belief. Similarly, an edge weight of $c_e < 0$ suggests that u and v are dissimilar. An edge weight of $c_e = 0$ (or, equivalently, the lack of an edge between u and v), indicates no belief about the similarity of u and v.

In this paper, our goal is to output a partition or *clustering* $\mathbb{S} = \{S_1, \ldots, S_k\}$ of the vertices that minimizes disagreements. The disagreements or cost of a partition is the total weight of the "mistakes", that is, the weight of positive edges between clusters and the absolute value of the weight of negative edges within clusters. In the case $c_e \in \{-1, 0, +1\}$, the cost of a partition is simply the number of cut positive edges plus uncut negative edges. Intuitively, this objective penalizes the clustering whenever presumed similar objects are in different clusters and presumed dissimilar objects are in the same cluster. For the purposes of approximation algorithms, minimizing disagreements is different from maximizing agreements (the weight of cut negative edges plus uncut positive edges).

We introduce the following notation for the cost of a clustering:

$$\begin{aligned}
\mathrm{cost}(\mathbb{S}) &= \mathrm{cost}_p(\mathbb{S}) + \mathrm{cost}_m(\mathbb{S}), \\
\mathrm{cost}_p(\mathbb{S}) &= \sum \big\{ |c_e| \ : \ e = (u, v) \in E;\ c_e > 0;\ \text{and } \forall i, |\{u, v\} \cap S_i| \le 1 \big\}, \\
\mathrm{cost}_m(\mathbb{S}) &= \sum \big\{ |c_e| \ : \ e = (u, v) \in E;\ c_e < 0;\ \text{and } \exists i, |\{u, v\} \cap S_i| = 2 \big\}.
\end{aligned}$$

We will refer to the optimal clustering as OPT and its cost as cost(OPT).

Previous approximation algorithms. Bansal et al. [1] give a constant factor approximation for this problem in the special case of complete graphs with edge weights in $\{-1, +1\}$. Their algorithm 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 $+1$ edges and one -1 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, round it, and use its optimal solution to bound our approximation factor.

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 will 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 cost($\mathbb{S}$) 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. Define constants

$$m_e = \begin{cases} |c_e| & \text{if } c_e < 0, \\ 0 & \text{if } c_e \geq 0, \end{cases}$$

and

$$p_e = \begin{cases} |c_e| & \text{if } c_e > 0, \\ 0 & \text{if } c_e \leq 0. \end{cases}$$

Then

$$\text{cost}(\mathbb{S}) = \sum_{e \in E} m_e(1 - x_e) + \sum_{e \in E} p_e x_e .$$

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:

$$\begin{aligned} \text{minimize} \ & \sum_{e \in E} m_e(1 - x_e) + \sum_{e \in E} p_e x_e \\ \text{subject to} \ & x_{uv} \in [0, 1] \\ & x_{uv} + x_{vw} \geq x_{uw} \\ & x_{uv} = x_{vu} \end{aligned}$$

Because the solution set to this linear program contains the solution set to the integer program, the optimal solution to the linear program is a lower bound on the cost of the optimal clustering.

3 Approximation in General Graphs

We use the 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 different. 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. A modification to this approach, outlined in Section 4, will yield an $O(r^3)$ approximation for $K_{r,r}$-minor-free 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 $x_{uv} \leq r$, 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* cut(S) of a set S of nodes is the cost of the positive edges with exactly one endpoint in S, i.e.,

$$\text{cut}(S) = \sum_{|\{v,w\} \cap S| = 1,\ (v,w) \in E} p_{vw}.$$

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

$$\text{vol}(S) = \sum_{\{v,w\} \subset S,\ (v,w) \in E} p_{vw} x_{vw}.$$

Finally, the *volume* of a ball is the volume of $B(u, r)$ including the fractional weighted distance of edges leaving $B(u, r)$. In other words, if $(v, w) \in E$ is a cut edge of ball $B(u, r)$ with $v \in B(u, r)$ and $w \notin B(u, r)$, then (v, w) contributes $p_{vw} \cdot x_{vw} \cdot (r - x_{uv})$ 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 $\text{cost}_p(\text{FRAC}) = F$. Let the initial volume I of the balls defined in the algorithm be F/n.

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 $\mathrm{cut}(B(u,r)) \leq c\ln(n+1) \times \mathrm{vol}(B(u,r))$.
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.

In this algorithm, c is some constant which we will determine later. This algorithm is clearly polynomial 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 will refer to the optimal fractional solution as FRAC, the solution our algorithm returns as SOL, and the optimal integral solution as OPT. We also use $\mathrm{FRAC}(x_{uv})$ and $\mathrm{SOL}(x_{uv})$ to denote the fractional and rounded solution to 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):

$$\begin{aligned}
\mathrm{cost}_p(\mathrm{SOL}) &= \sum_{(u,v)\in E} p_{uv}\,\mathrm{SOL}(x_{uv}) \\
&= \tfrac{1}{2} \sum_{\mathrm{ball}\ B} \mathrm{cut}(B) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \sum_{\mathrm{ball}\ B} \mathrm{vol}(B) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \left(\sum_{(u,v)\in E} p_{uv}\,\mathrm{FRAC}(x_{uv}) + \sum_{\mathrm{ball}\ B} \frac{F}{n} \right) \\
&\leq \tfrac{c}{2}\ln(n+1) \times \big(\mathrm{cost}_p(\mathrm{FRAC}) + F\big) \\
&\leq c\ln(n+1) \times \mathrm{cost}_p(\mathrm{OPT})
\end{aligned}$$

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 [20]:

Lemma 1. *For any vertex u and family of balls $B(u,r)$, the condition $\mathrm{cut}(B(u,r)) \leq c\ln(n+1) \times \mathrm{vol}(B(u,r))$ 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):

$$\begin{aligned}
\mathrm{cost}_m(\mathrm{FRAC}) &= \sum_{(u,v)\in E} m_{uv}(1 - \mathrm{FRAC}(x_{uv})) \\
&\geq \sum_{\mathrm{balls}\ B} \sum_{(u,v)\in B\cap E} m_{uv}(1 - \mathrm{FRAC}(x_{uv}))
\end{aligned}$$

$$\begin{aligned}
&\geq \sum_{\text{balls } B} \sum_{(u,v)\in B\cap E} m_{uv}(1-2/c) \\
&\geq (1-2/c) \sum_{\text{balls } B} \sum_{(u,v)\in B\cap E} m_{uv} \\
&= \tfrac{c-2}{c} \operatorname{cost}_m(\text{SOL})
\end{aligned}$$

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

$$\begin{aligned}
\operatorname{cost}(\text{SOL}) &= \operatorname{cost}_p(\text{SOL}) + \operatorname{cost}_m(\text{SOL}) \\
&\leq \frac{c}{2}\ln(n+1) \times \operatorname{cost}_p(\text{OPT}) + \frac{c-2}{c}\operatorname{cost}_m(OPT) \\
&\leq \max\left\{\frac{c}{2}\ln(n+1), \frac{c-2}{c}\right\} \operatorname{cost}(\text{OPT})
\end{aligned}$$

and thus we have an $O(\ln n)$ approximation, where the lead constant, $c/2$, is just slightly larger than 1.

4 Approximation in $K_{r,r}$-Minor-Free Graphs

In $K_{r,r}$-minor-free graphs, we can use a theorem of Klein et al. [12] 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 1. [12] *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*[1] *$O(r^2\delta)$ 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 [19] or Bejerano et al. [2]). In particular, this implies constant factor approximations for planar graphs.

[1] 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.

The idea is as follows. Given a $K_{r,r}$-minor-free graph G with weights p_e and edge lengths x_e as defined by the linear program, we subdivide each edge e into a chain of $\lceil kx_e \rceil$ edges of the same weight p_e for some appropriate k, yielding a new graph G'. We apply Theorem 1 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 kd in G'. Hence, if we take δ such that $O(r^2\delta) < 2k/c$, the components in G will have diameter at most $2/c$. It is sufficient to take $\delta = O(k/r^2)$. To bound the weight of the cut F, we just need to bound the total weight U' of the graph G'. Let $U = \sum_{e \in G} p_e$ be the total weight of edges in G and recall $\operatorname{vol}(G) = \sum_{e \in G} p_e x_e$. Then

$$\begin{aligned} U' &= \sum_{e \in G'} p_e \\ &= \sum_{e \in G} \lceil kx_e \rceil p_e \\ &\leq \sum_{e \in G} (kx_e + 1) p_e \\ &= k \operatorname{vol}(G) + U. \end{aligned}$$

By Theorem 1, the weight of F is $O(rU'/\delta) = O(r^3(k \operatorname{vol}(G) + U)/k)$. Taking $k = U/\operatorname{vol}(G)$, this becomes $O(r^3 \operatorname{vol}(G))$ 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. [12] consists of r breath-first searches of G', and these can be implemented without explicitly subdividing G. Thus, the algorithm is polynomial.

5 Lower Bounds

We prove that it is APX-hard to minimize disagreements in correlation clustering. We use a reduction from the APX-hard problem minimum multicut [4]: given a graph G and k pairs of nodes $P = \{(u_1, v_1), \ldots, (u_k, v_k)\}$, find a set of edges of minimum weight that, when removed, separate each pair of nodes $p \in P$.

Theorem 2. *An r-approximation for minimizing disagreements in correlation clustering implies an r-approximation for the minimum-multicut problem.*

Proof. Given a multicut instance G', construct graph G as follows. For every edge in G' of weight c'_e, add an edge to G of weight $c_e = c'_e$. Note all these c_e are positive. Let M be the maximum c_e. For each pair $p = (u_i, v_i)$, add an edge e between u_i and v_i of weight $c_e = -(M+1)n^2$. Note we have added at most n^2 edges and increased the maximum weight by factor at most n^2 so G is polynomial in the size of G'.

We claim that the cost of the optimal multicut in G' equals the cost of the optimal clustering in G. A correlation clustering of G that puts every vertex in its own component costs at most Mn^2. However, any solution that does not separate all pairs costs at least $(M+1)n^2$, and so the optimum solution must separate all pairs. As the only negative edges in G are those between these pairs, the optimum solution only makes mistakes on positive edges (disagreements between clusters). Therefore the optimum clustering in G induces a multicut of the same cost in G'. In fact, any clustering which only makes positive mistakes induces a multicut of the same cost in G'. Furthermore, any multicut in G' cuts all negative edges in G and thus induces a clustering in G of the same cost. In particular, the optimum multicut in G' induces a clustering in G of the same cost, and the claim follows.

Now suppose we have an r-approximation algorithm for the correlation-clustering problem. Consider the output of this algorithm on graph G. If the outputted clustering only makes mistakes on positive edges (and so separates all pairs), then the above arguments show that this clustering induces a multicut which is an r-approximation to the optimal multicut in G'. If the output clustering does not cut some negative edge, then the cost is at least $(M+1)n^2$. In this case, the clustering which places every node in a separate cluster costs at most Mn^2 and is an r-approximation. Therefore, cutting every edge in G' is an r-approximation to the optimal multicut in G'. Thus, given an r-approximation algorithm for the correlation-clustering problem, we can design an r-approximation algorithm for the minimum-multicut problem. $\square$

Because the minimum-multicut problem is APX-hard, this theorem shows that there is no PTAS for minimizing disagreements in correlation clustering unless $P = NP$. Furthermore, it shows that this problem is as hard as minimum multicut. The current best approximation for minimum multicut is $O(\log k)$ [7]. Because k can be $\Omega(n^2)$ in the worst case, an $o(\log n)$ approximation for our problem would require improving the $O(\log k)$ approximation for minimum multicut, which a long-standing open problem.

The above reduction is also useful in leading us to find difficult instances for the correlation-clustering problem. Garg, Vazirani, and Yannakakis [7] construct an example that shows that the ratio between the value of the minimum multicut and maximum multicommodity flow (i.e., optimal multicut linear-program value) can be as large as $\Omega(\log k)$. The example uses a *bounded-degree expander.*

Definition 1. *A graph G is a* bounded-degree expander *if there is some constant d such that all nodes have degree at most d and for any set S of vertices, $|S| < n/2$, the number of edges that cross S is at least $c|S|$ for some constant c.*

We can use the same example to prove that the gap of our linear program (the ratio between OPT and FRAC) can be $\Omega(\log n)$, suggesting that it is probably not possible to obtain a $o(\log n)$ approximation by rounding this linear program.

Theorem 3. *The gap of the linear program presented in Section 2 is $\Omega(\log n)$ in the worst case.*

Proof. Consider a bounded-degree expander G'. Note since the degree of each node is at most d, there are at least $n - \sqrt{n}$ vertices at a distance of at least

$\log_d n/2$ from any vertex v. Construct $O(n^2)$ pairs of vertices as follows: for each vertex v, add the $O(n)$ pairs (v, u) where u is a vertex of distance at least $(\log_d n)/2$ from v. Assign all edges in the graph weight $c_e = 1$. Perform the above reduction to get graph G. As discussed, the optimal integral solution separates all the $O(n^2)$ pairs of vertices. Hence, the diameters of the resulting clusters must be $o(\log_d n)$. Because the vertices have bounded degree, the size of the clusters is bounded by $n/2$. By the expansion property of G', we must cut at least $c \sum_{S \in \mathbb{S}} |S| = cn$ positive edges, and so cost(OPT) $= \Omega(n)$.

On the other hand, assigning $x_e = 2/\log_d n$ for positive edges and $x_e = 1$ for negative edges is a feasible fractional solution of value at most $(dn/2) \times (2/\log_d n)$, and so cost(FRAC) $= O(n/\log n)$. The theorem follows. □

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, and an $O(r^3)$ approximation for $K_{r,r}$-minor-free graphs. We also showed that this problem is as hard as minimum multicut, and that the natural linear-programming formulation has a gap of $\Omega(\log n)$.

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 "bad" 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 PTAS for maximizing agreements in complete graphs with edge weights in $\{-1, +1\}$. In *maximizing agreements*, the cost of a solution is the weight of positive agreements (uncut positive edges) plus negative agreements (cut negative edges). They 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. [12] rounding technique both provide a radius guarantee on the outputted 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 Shuchi Chawla, Avrim Blum, Mohammad Mahdian, David Liben-Nowell, and Grant Wang. Many results in this paper were inspired by conversations with Seffi Naor.

References

[1] Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *IEEE Symp. on Foundations of Computer Science*, 2002.

[2] Y. Bejerano, N. Immorlica, S. Naor, and M. Smith. Location area design in cellular networks. *International Conference on Mobile Computing and Networking*, 2003.

[3] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. Unpublished Manuscript.

[4] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiway cuts. *ACM Symp. on Theory of Comp.*, 1992.

[5] Dotan Emanuel and Amos Fiat. Correlation clustering — minimizing disagreements on arbitrary weighted graphs. *European Symp. on Algorithms*, 2003.

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

[7] N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. Comp.*, 25, 1996.

[8] D. S. Hochbaum and D. B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *Journal of the ACM*, 33, 1986.

[9] T. C. Hu. Multicommodity network flows. *Operations Research*, 1963.

[10] Kamal Jain and Vijay V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. *IEEE Symp. on Foundations of Computer Science*, 1999.

[11] 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), 2002.

[12] Philip N. Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. *ACM Symp. on Theory of Comp.*, 1993.

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

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

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

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

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

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

[19] 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.

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

[21] M. Yannakakis, P. C. Kanellakis, S. C. Cosmadakis, and C. H. Papadimitriou. Cutting and partitioning a graph after a fixed pattern. *10th Intl. Coll. on Automata, Languages, and Programming*, 1983.