Correlation Clustering with a Fixed Number of Clusters[*]

Ioannis Giotis[†] Venkatesan Guruswami[*‡]

Abstract

We continue the investigation of problems concerning *correlation clustering* or *clustering with qualitative information*, which is a clustering formulation that has been studied recently [5, 7, 8, 3]. The basic setup here is that we are given as input a complete graph on n nodes (which correspond to nodes to be clustered) whose edges are labeled $+$ (for similar pairs of items) and $-$ (for dissimilar pairs of items). Thus we have only as input qualitative information on similarity and no quantitative distance measure between items. The quality of a clustering is measured in terms of its number of agreements, which is simply the number of edges it correctly classifies, that is the sum of number of $-$ edges whose endpoints it places in different clusters plus the number of $+$ edges both of whose endpoints it places within the same cluster.

In this paper, we study the problem of finding clusterings that maximize the number of agreements, and the complementary minimization version where we seek clusterings that minimize the number of disagreements. We focus on the situation when the number of clusters is stipulated to be a *small constant* k. Our main result is that for every k, there is a polynomial time approximation scheme for both maximizing agreements and minimizing disagreements. (The problems are NP-hard for every $k \geq 2$.) The main technical work is for the minimization version, as the PTAS for maximizing agreements follows along the lines of the property tester for Max k-CUT from [13].

In contrast, when the number of clusters is not specified, the problem of minimizing disagreements was shown to be APX-hard [7], even though the maximization version admits a PTAS.

1 Introduction

In this work, we investigate problems concerning an appealing formulation of clustering called *correlation clustering* or *clustering using qualitative information* that has been studied recently in several works, including [6, 17, 5, 7, 8, 3, 2, 4]. The basic setup here is to cluster a collection of n items given as input only qualitative information concerning similarity between pairs of items; specifically for every pair of items, we are given a (Boolean) label as to whether those items are similar or dissimilar. We are not provided with any quantitative information on how different pairs of elements are, as is typically assumed in most clustering formulations. These formulations take as input a metric on the items and then aim to optimize some function of the pairwise distances of the items within and across clusters. The objective in our formulation is to produce a partitioning into

[*]A version of this paper will appear in the *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January 2006.

[†]Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195. {giotis,venkat}@cs.washington.edu

[‡]Research supported in part by NSF Career Award CCF-0343672.

clusters that places similar objects in the same cluster and dissimilar objects in different clusters, to the extent possible.

An obvious graph-theoretic formulation of the problem is the following: given a complete graph on n nodes with each edge labeled either "+" (similar) or "−" (dissimilar), find a partitioning of the vertices into clusters that agrees as much as possible with the edge labels. The maximization version, call it MaxAgree, seeks to maximize the number of agreements: the number of + edges inside clusters plus the number of − edges across clusters. The minimization version, denoted MinDisAgree, aims to minimize the number of disagreements: the number of − edges within clusters plus the number of + edges between clusters.

In this paper, we study the above problems when the maximum number of clusters that we are allowed to use is stipulated to be a fixed constant k. We denote the variants of the above problems that have this constraint as MaxAgree[k] and MinDisAgree[k]. We note that, unlike most clustering formulations, the MaxAgree and MinDisAgree problems are not trivialized if we do not specify the number of clusters k as a parameter — whether the best clustering uses few or many clusters is automatically dictated by the edge labels. However, the variants we study are also interesting formulations, which are well-motivated in settings where the number of clusters might be an external constraint that has to be met, even if there are "better" clusterings (i.e., one with more agreements) with a different number of clusters. Moreover, the existing algorithms for, say MinDisAgree, cannot be modified in any easy way to output a quality solution with at most k clusters. Therefore k-clustering variants pose new, non-trivial challenges that require different techniques for their solutions.

In the above description, we have assumed that every pair of items is labeled as + or − in the input. In a more general variant, intended to capture situations where the classifier providing the input might be unable to label certain pairs of elements are similar or dissimilar, the input is an arbitrary graph G together with $\pm$ labels on its edges. We can again study the above problems MaxAgree[k] (resp. MinDisAgree[k]) with the objective being to maximize (resp. minimize) the number of agreements (resp. disagreements) on edges of E (that is, we do not count non-edges of G as either agreements or disagreements). In situations where we study this more general variant, we will refer to these problems as MaxAgree[k] on *general* graphs and MinDisAgree[k] on general graphs. When we don't qualify with the phrase "on general graphs", we will always mean the problems on complete graphs.

Our main result in this paper is a polynomial time approximation scheme (PTAS) for MaxAgree[k] as well as MinDisAgree[k] for $k \geq 2$. We now discuss prior work on these problems, followed by a more detailed description of results in this paper.

1.1 Previous and related work

The above problem seems to have been first considered by Ben-Dor et al. [6] motivated by some computational biology questions. Later, Shamir et al. [17] studied the computational complexity of the problem and showed that MaxAgree (and hence also MinDisAgree), as well as MaxAgree[k] (and hence also MinDisAgree[k]) for each $k \geq 2$ is NP-hard. They, however, used the term "Cluster Editing" to refer to this problem.

Partially motivated by some machine learning problems concerning document classification, Bansal, Blum, and Chawla [5] also independently formulated and considered this problem. In particular, they initiated the study of approximate solutions to MinDisAgree and MaxAgree, and presented a PTAS for MaxAgree and a constant factor approximation algorithm for MinDis-

Agree (the approximation guarantee was a rather large constant, though). They also noted a simple factor 3 approximation algorithm for MinDisAgree[2]. Charikar, Guruswami and Wirth [7] proved that MinDisAgree is APX-hard, and thus one cannot expect a PTAS for the minimization problem similar to the PTAS for MaxAgree. They also gave a factor 4 approximation algorithm for MinDisAgree by rounding a natural LP relaxation using the region growing technique. Recently, Ailon et al [2] presented an elegant combinatorial factor 3 approximation algorithm with a clever analysis for MinDisAgree; they also get a factor 5/2 approximation using LP techniques on top of their basic approach.

The problems on general graphs have also received attention. It is known that both MaxAgree and MinDisAgree are APX-hard [5, 7]. Using a connection to minimum multicut, several groups [7, 11, 12] presented an $O(\log n)$ approximation algorithm for MinDisAgree. In fact, it was noted in [12] that the problem is as hard to approximate as minimum multicut (and so this $\log n$ factor seems very hard to improve). For the maximization version, algorithms with performance ratio better than 0.766 are known for MaxAgree [7, 18]. The latter work by Swamy [18] shows that a factor 0.7666 approximation can also be achieved when the number of clusters is specified (i.e., for MaxAgree[k] for $k \geq 2$).

Another problem that has been considered, let us call it MaxCorr, is that of maximizing correlation, defined to be the difference between the number of agreements and disagreements. A factor $O(\log n)$ approximation for MaxCorr on complete graphs is presented in [16, 8], and an $O(\log \theta(\overline{G}))$ approximation is presented in [3] for general graphs G, where $\theta(\cdot)$ is the Lovász Theta Function. Alon et al [3] showed an integrality gap of $\Omega(\log n)$ for the standard semidefinite program relaxation for MaxCorr (the largest such integrality gap for a graph is called the *Grothendieck constant* of the graph — thus these results establish the Grothendieck constant of the complete graph on n vertices to be $\Theta(\log n)$). Very recently, Arora et al [4] proved a factor $\log^{\alpha} n$ inapproximability result for the weighted version of MaxCorr for some $\alpha > 0$.

1.2 Our results

The only previous approximation for MinDisAgree[k] was a factor 3 approximation algorithm for the case $k = 2$ [5]. The problems were shown to be NP-hard for every $k \geq 2$ in [17] using a rather complicated reduction. In this paper, we will provide a much simpler NP-hardness proof and prove that both MaxAgree[k] and MinDisAgree[k] admit a polynomial time approximation scheme for every $k \geq 2$.[1] The existence of a PTAS for MinDisAgree[k] is perhaps surprising in light of the APX-hardness of MinDisAgree when the number of clusters is not specified to be a constant (recall that the maximization version *does* admit a PTAS even when k is not specified).

It is often the case that minimization versions of problems are harder to solve compared to their complementary maximization versions. The APX-hardness of MinDisAgree despite the existence of a PTAS for MaxAgree is a notable example. The difficulty in these cases is when the optimum value of the minimization version is very small, since then even a PTAS for the complementary maximization problem need not provide a good approximation for the minimization problem. In this work, we first give a PTAS for MaxAgree[k]. This algorithm uses random sampling and follows closely along the lines of the property testing algorithm for Max k-Cut due to [13]. We then develop a PTAS for MinDisAgree[k], which is our main result. This requires more work

[1] Our approximation schemes will be randomized and deliver a solution with the claimed approximation guarantee with high probability. For simplicity, we do not explicitly mention this from now on.

and the algorithm returns the better of two solutions, one of which is obtained using the PTAS for MaxAgree[k].

The difficulty in getting a PTAS for the minimization version is similar to that faced in the problem of Min k-sum clustering, which has the complementary objective function to Metric Max k-Cut. We remark that while an elegant PTAS for Metric Max k-Cut due to de la Vega and Kenyon [10] has been known for several years, only recently has a PTAS for Min k-sum clustering been obtained [9]. We note that the case of Min 2-sum clustering though was solved in [14] soon after the Metric Max Cut algorithm of [10], but the case $k > 2$ appeared harder. Similarly to this, for MinDisAgree[k], we are able to quite easily give a PTAS for the 2-clustering version using the algorithm for MaxAgree[2], but we have to work harder for the case of $k > 2$ clusters. Some of the difficulty that surfaces when $k > 2$ is detailed in Section 4.1.

In Section 5, we also note some results on the complexity of MaxAgree[k] and MinDisAgree[k] on general graphs — these are easy consequences of connections to problems like Max CUT and graph colorability.

Our work seems to nicely complete the understanding of the complexity of problems related to correlation clustering. Our algorithms not only achieve excellent approximation guarantees but are also sampling-based and are thus simple and quite easy to implement.

2 NP-hardness of MinDisAgree and MaxAgree

In this section we show that the exact versions of problems we are trying to solve are NP-hard. An NP-hardness result for MaxAgree on complete graphs was shown in [5]; however their reduction crucially relies on the number of clusters growing with the input size, and thus does not yield any hardness when the number of clusters is a fixed constant k. It was shown by Shamir, Sharan, and Tsur [17], using a rather complicated reduction, that these problems are NP-hard for each fixed number $k \geq 2$ of clusters. We will provide a short and intuitive proof that MinDisAgree[k] and MaxAgree[k] are NP-hard.

Clearly it suffices to establish the NP-hardness of MinDisAgree[k] since MaxAgree[k] can be easily reduced on a complimentary graph. We will first establish NP-hardness for $k = 2$, the case for general k will follow by a simple "padding" with $(k-2)$ large collection of nodes with $+$ edges between nodes in each collection and $-$ edges to everywhere else.

Theorem 1 MinDisAgree[2] *on complete graphs is NP-hard.*

Proof: We know that Graph Min Bisection, namely partitioning the vertex set of a graph into two equal halves so that the number of edges connecting vertices in different halves is minimized, is NP-hard. From an instance G of Min Bisection with n(even) vertices we obtain a complete graph G' using the following polynomial time construction.

Start with G and label all existing edges of G as $+$ edges in G' and non-existing edges as $-$ edges. For each vertex v create an additional set of n vertices. Let's call these vertices together with v, a "group" V_v. Connect with $+$ edges all pairs of vertices within V_v. All other edges with one endpoint in V_v are labeled as $-$ edges (except those already labeled).

We will now show that any 2-clustering of G' with the minimum number of disagreements, has 2 clusters of equal size with all vertices of any group in the same cluster. Consider some optimal 2-clustering W with 2 clusters W_1 and W_2 such that $|W_1| \neq |W_2|$ or not all vertices of some group are in the same cluster. Pick some group V_v such that not all its vertices are assigned in the same cluster. If such a group cannot be found, pick a group V_v from the larger cluster. Place all the vertices of the group in the same cluster obtaining W' such that $||W'_1| - |W'_2||$ is minimized.

Let's assume that V_v^1 vertices of group V_v were in W_1 and V_v^2 in W_2. Wlog, let's assume that W' is obtained by moving the V_v^1 group vertices in cluster W_2.

$$W'_1 = W_1 \setminus V_v^1, W'_2 = W_2 \cup V_v^1$$

We now observe the following facts about the difference in the number of disagreements between W' and W.

- Clearly the number of disagreements between vertices not in V_v and between one vertex in V_v^2 with one in W'_1 remains the same.
- The number of disagreements is decreased by $|V_v^1| \cdot |V_v^2|$ based on the fact that all edges within V_v are $+$ edges.
- It is also decreased by at least $|V_v^1| \cdot |W'_1| - (n-1)$ based on the fact that all but at most $n-1$ edges connecting vertices of V_v to the rest of the graph are $-$ edges.
- The number of disagreements increases at most $|V_v^1| \cdot |W_2 \setminus V_v^2|$ because (possibly) all of the vertices in V_v^1 are connected with $-$ edges with vertices in W_2 outside their group.

Overall, the difference in the number of disagreements is at most $|V_v^1| \cdot |W_2 \setminus V_v^2| - |V_v^1| \cdot |V_v^2| - |V_v^1| \cdot |W'_1| + (n-1)$. Notice that since $||W'_1| - |W'_2||$ was minimized it must be the case that $|W'_1| \geq |W_2 \setminus V_v^2|$. Moreover since a group has an odd number of vertices and the total number of vertices of G' is even, it follows that $|W'_1| \neq |W_2 \setminus V_v^2|$ and $|W'_1| - |W_2 \setminus V_v^2| \geq 1$. Therefore the total number of disagreements increases at most $(n-1) - |V_v^1| \cdot (|V_v^2| + 1)$. Since $|V_v^1| + |V_v^2| = n+1$ and V_v^1 cannot be empty, it follows that $|V_v^1| \cdot (|V_v^2| + 1) \geq n$ and the number of disagreements strictly decreases contradicting the optimality of W.

Therefore the optimal solution to the MinDisAgree[2] instance has 2 clusters of equal size and all vertices of any group are contained in a single cluster. It is now trivial to see that an optimal solution to the Min Bisection problem can be easily derived from the MinDisAgree[2] solution which completes the reduction. ■

We are now able to easily derive the following NP-hardness result.

Theorem 2 *For every $k \geq 2$, the problems* MaxAgree[k] *and* MinDisAgree[k] *on complete graphs are NP-hard.*

Proof: Consider an instance of the MinDisAgree[2] problem on a graph G with n vertices. Create a graph G' by adding to G, $k-2$ "groups" of $n+1$ vertices each. All edges within a group are marked as $+$ edges, while the remaining edges are marked as $-$ edges.

Consider now a k-clustering of G' such that the number of disagreements is minimized. It is easy to see that all the vertices of a group must make up one cluster. Also observe that any of the original vertices cannot end up in one group's cluster since that would induce $n+1$ disagreements, strictly more than it could possibly induce in any of the 2 remaining clusters. Therefore the 2 non-group clusters are an optimal 2-clustering of G. The theorem easily follows. ■

3 PTAS for maximizing agreement with k clusters

In this section we will present a PTAS for MaxAgree[k] for every fixed constant k. Our algorithm follows closely the PTAS for Max k-CUT by Goldreich et al.[13]. In the next section, we will present our main result, namely a PTAS for MinDisAgree[k], using the PTAS for MaxAgree[k] together with additional ideas.[2]

Theorem 3 *For every $k \geq 2$, there is a polynomial time approximation scheme for* MaxAgree[k].

Proof: We first note that for every $k \geq 2$, and every instance of MaxAgree[k], the optimum number OPT of agreements is at least $n^2/16$. Let n_+ be the number of positive edges, and $n_- = \binom{n}{2} - n_+$ be the number of negative edges. By placing all vertices in a single cluster, we get n_+ agreements. By placing vertices randomly in one of k clusters, we get an expected $(1-1/k)n_-$ agreements just on the negative edges. Therefore $\mathsf{OPT} \geq \max\{n_+, (1-1/k)n_-\} \geq (1-1/k)\binom{n}{2}/2 \geq n^2/16$. The proof now follows from Theorem 4 which guarantees a solution within additive εn^2 of OPT for arbitrary $\varepsilon > 0$. ■

Theorem 4 *On input ε, δ and a labeling $\mathcal{L}$ of the edges of a complete graph G with n vertices, with probability at least $1-\delta$, algorithm* MaxAg *outputs a k-clustering of the graph such that the number of agreements induced by this k-clustering is at least $\mathsf{OPT} - \varepsilon n^2/2$, where OPT is the optimal number of agreements induced by any k-clustering of G. The running time of the algorithm is $n \cdot k^{O(\varepsilon^{-3}\log(k/(\varepsilon\delta)))}$.*

The proof of this theorem is presented in Section 3.2, and we now proceed to describe the algorithm in Figure 1.

3.1 Overview.

Our algorithm is given a complete graph $G(V,E)$ on n vertices. All the edges are marked as $+$ or $-$, denoting whether adjacent vertices are on agreement or disagreement respectively. For a vertex v, let $\Gamma^+(v)$ be the set of vertices adjacent to v via $+$ edges, and $\Gamma^-(v)$ the set of vertices adjacent to v via $-$ edges.

The algorithm works in $m = O(1/\varepsilon)$ steps. At each step we are placing $\Theta(\varepsilon n)$ vertices into clusters. We will show that with constant probability our choices of S_i's will allow us to place the vertices in such a way that the decrease in the number of agreements with respect to an optimal clustering is $O(\varepsilon^2 n^2)$ per step, thus the algorithm outputs a solution that has $O(\varepsilon n^2)$ less agreements than any optimal solution.

3.2 Performance analysis of *MaxAg*(k,ε) algorithm.

Consider an arbitrary optimal k-clustering of the graph $D \equiv (D_1, \ldots, D_k)$. We consider the subsets of each cluster over our partition of vertices, defined as

$$\begin{aligned} \text{for } j = 1, \ldots k,\ D_j^i &\equiv D_j \cap V^i \\ D^i &\equiv (D_1^i, \ldots, D_k^i) \end{aligned}$$

[2]This is also similar in spirit, for example, to the PTAS for Min 2-sum clustering based on the PTAS for Metric Max CUT [14, 10].

```
Algorithm MaxAg(k, ε):
Input: A labeling $\mathcal{L} : \binom{n}{2} \to \{+,-\}$ of the edges of the complete graph on vertex set $V$.
Output: A $k$-clustering of the graph, i.e., a partition of $V$ into (at most) $k$ parts $V_1, V_2, \ldots, V_k$.

1. Construct an arbitrary partition of the graph into roughly equal parts, $(V^1, V^2, \ldots, V^m), m = \lceil \frac{4}{\varepsilon} \rceil$.
2. For $i = 1 \ldots m$, choose uniformly at random with replacement from $V \setminus V^i$, a subset $S^i$ of size
     $r = \Theta\left(\frac{1}{\varepsilon^2} \log \frac{k}{\varepsilon\delta}\right)$.
3. For each clustering of all the sets $S^i$ into $(S^i_1, \ldots, S^i_k)$ do
   (a) For $i = 1 \ldots m$ do the following
      (i) For each vertex $v \in V^i$ do
         (1) For $j = 1 \ldots k$, let
               $\beta_j(v) = |\Gamma^+(v) \cap S^i_j| + \sum_{l \neq j} |\Gamma^-(v) \cap S^i_l|$.
         (2) Place $v$ in cluster $\text{argmax}_j \beta_j(v)$.
   (b) If the current clustering has more agreements than the currently stored one, store it.
4. Output stored clustering.
```

Figure 1: ***MaxAg***(k, ε) algorithm

Let's also call the clustering output by our algorithm $W \equiv (W_1, \ldots, W_k)$ and define in the same fashion.

$$\begin{aligned} \text{for } j = 1, \ldots k,\ W^i_j &\equiv W_j \cap V^i \\ W^i &\equiv (W^i_1, \ldots, W^i_k) \end{aligned}$$

We will now define a sequence of *hybrid* clusterings, such that hybrid clustering H^i, for $i = 1, 2, \ldots, m+1$, consists of the vertices as clustered by our algorithm up to (not including) the i'th step and the rest of the vertices as clustered by D.

$$\begin{aligned} H^i &\equiv (H^i_1, \ldots, H^i_k) \\ \mathcal{H}^i &\equiv (\mathcal{H}^i_1, \ldots, \mathcal{H}^i_k) \\ \text{for } j = 1, \ldots k,\ H^i_j &\equiv (\cup_{l=1}^{i-1} W^l_j) \cup (\cup_{l=i}^{m} D^l_j) \\ \text{for } j = 1, \ldots k,\ \mathcal{H}^i_j &\equiv H^i_j \setminus V^i \end{aligned}$$

Since we are going through all possible clusterings of the random sample sets S^i, for the rest of the analysis consider the loop iteration when the clustering of each S^i exactly matches how it is clustered in $\mathcal{H}^i$, i.e., for $j = 1, 2, \ldots, k$, we have $S^i_j = S^i \cap \mathcal{H}^i_j$. Of course, taking the overall best clustering can only help us.

The following lemma captures the fact that our random sample with high probability gives us a good estimate on the number of agreements towards each cluster for most of the vertices considered.

Lemma 5 *For $i = 1 \ldots m$, with probability at least $1 - (\delta/4m)$ on the choice of S^i, for all but at most an $\varepsilon/8$ fraction of the vertices $v \in V^i$, the following holds for $j = 1, \ldots k$,*

$$\left| \frac{|\Gamma^+(v) \cap S^i_j|}{r} - \frac{|\Gamma^+(v) \cap \mathcal{H}^i_j|}{|V \setminus V^i|} \right| \leq \frac{\varepsilon}{32} \ . \tag{1}$$

(Note that if (1) above holds, then it also holds with $\Gamma^-(v)$ in place of $\Gamma^+(v)$.)

Proof: Consider an arbitrary vertex $v \in V^i$ and the randomly chosen set $S^i = \{u_1, \dots, u_r\}$. For each $j \in \{1, \dots, k\}$, we define the random variables

$$\text{for } l = 1, \dots r,\ \alpha_j^l = \begin{cases} 1, & \text{if } u_l \in \Gamma^+(v) \cap S_j^i; \\ 0, & \textit{otherwise.} \end{cases}$$

Clearly $\sum_{l=1}^r \alpha_j^l = |\Gamma^+(v) \cap S_j^i|$ and $Pr[\alpha_j^l = 1] = \frac{|\Gamma^+(v) \cap \mathcal{H}_j^i|}{|V \backslash V^i|}$.
Using an additive Chernoff bound we get that

$$Pr\left[\left|\frac{|\Gamma^+(v) \cap S_j^i|}{r} - \frac{|\Gamma^+(v) \cap \mathcal{H}_j^i|}{|V \backslash V^i|}\right| > \frac{\varepsilon}{32}\right] < 2 \cdot \exp(-2(\tfrac{\varepsilon}{32})^2 r) < \frac{\varepsilon\delta}{32mk}$$

Defining a random variable to count the number of vertices not satisfying inequality(1) and using Markov's inequality we get that for that particular j, inequality(1) holds for all but a fraction $\varepsilon/8$ of vertices $v \in V^i$, with probability at least $1 - (\delta/4mk)$. Using a probability union bound the lemma easily follows. ∎

We define $\mathsf{agree}(A)$ to be equal to the number of agreements induced by k-clustering A. Now consider the placement of V^i vertices in clusters $W_1^i, \dots, W_k^i$ as performed by the algorithm during step i. We will examine the number of agreements compared to the placement of the same vertices under H^i (placement under the optimal clustering), more specifically we will bound the difference in the number of agreements induced by placing vertices differently than H^i. The following lemma formalizes this concept.

Lemma 6 *For $i = 0, \dots m$, we have* $\mathsf{agree}(H^{i+1}) \geq \mathsf{agree}(D) - i \cdot \frac{1}{8}\varepsilon^2 n^2$

Proof: Observe that $H^1 \equiv D$ and $H^{m+1} \equiv W$. The only vertices placed differently between H^{i+1} and H^i are the vertices in V^i. Suppose that our algorithm places $v \in V^i$ in cluster x and v is placed in cluster x' under H^i. For each vertex v the number of agreements towards clusters other than x, x' remains the same, therefore we will focus on the number of agreements towards these two clusters and the number of agreements within V^i.

The number of agreements we could lose by thus misplacing v is

$$\mathsf{diff}_{xx'}(v) \quad = \quad |\Gamma^+(v) \cap \mathcal{H}_{x'}^i| - |\Gamma^+(v) \cap \mathcal{H}_x^i| + |\Gamma^-(v) \cap \mathcal{H}_x^i| - |\Gamma^-(v) \cap \mathcal{H}_{x'}^i|$$

Since our algorithm chose cluster x, by construction

$$|\Gamma^+(v) \cap S_x^i| + |\Gamma^-(v) \cap S_{x'}^i| \geq |\Gamma^+(v) \cap S_{x'}^i| + |\Gamma^-(v) \cap S_x^i| \tag{2}$$

If inequality (1) holds for vertex v, using it for $\Gamma^+(v)$ and $\Gamma^-(v)$ in both clusters x,x', we obtain bounds on the difference of agreements between our random sample's clusters $S_x^i, S_{x'}^i$ and the hybrid clusters $\mathcal{H}_x^i, \mathcal{H}_{x'}^i$. Combining with inequality (2) we get that $\mathsf{diff}_{xx'}(v)$ is at most $\frac{1}{8}\varepsilon n$. Therefore the total decrease in the number of agreements by this type of vertices is at most $\frac{1}{8}\varepsilon n|V^i| \leq \frac{1}{8}\varepsilon\frac{n^2}{m}$.

By Lemma 5 there are at most $(\varepsilon/8)|V^i|$ vertices in V^i for which inequality (1) doesn't hold. The total number of agreements originating from these vertices is at most $\frac{1}{8}\varepsilon|V^i|n \leq \frac{1}{8}\varepsilon\frac{n^2}{m}$. Finally, the total number of agreements from within V_i is at most $|V^i|^2 \leq \frac{1}{4}\varepsilon\frac{n^2}{m}$.

Overall the number of agreements that we could lose in one step of the algorithm is at most $\frac{1}{2}\varepsilon\frac{n^2}{m} \le \frac{1}{8}\varepsilon^2 n^2$. The lemma follows by induction. ■

The approximation guarantee of Theorem 4 easily follows from Lemma 6. We need to go through all possible k-clusterings of our random sample sets, a total of k^{mr} loop iterations. The inner loop (over i) runs m times, and each of those iterations can be implemented in $O(nr)$ time. The claimed running time bound of our algorithm thus follows. ■

4 PTAS for minimizing disagreements with k clusters

This section is devoted to the proof of the following theorem, which is our main result in this paper.

Theorem 7 (Main) *For every $k \ge 2$, there is a PTAS for* MinDisAgree$[k]$.

The algorithm for MinDisAgree$[k]$ will use the approximation scheme for MaxAgree$[k]$ as a subroutine. The latter already provides a very good approximation for the number of disagreements unless this number is very small. So in the analysis, the main work is for the case when the optimum clustering is right on most of the edges.

4.1 Idea behind the algorithm.

The case of 2-clusters turns out to be lot simpler and we use it to first illustrate the basic idea. By the PTAS for maximization, we only need to focus on the case when the optimum clustering has only $\mathsf{OPT} = \gamma n^2$ disagreements for some small $\gamma > 0$. We draw a random sample S and try all partitions of it, and focus on the run when we guess the right partition $S = S_1 \cup S_2$, namely the way some fixed optimal clustering $\mathcal{D}$ partitions S. Since the optimum has a very large number of agreements, there must exist a set A of size at least $(1-O(\gamma))n$ such that each node in A has a clear choice of which side it prefers to be on. Moreover, for each node in A, we can find out its choice correctly (with high probability) based on edges connecting it to nodes in the sample S. Therefore, we can find a clustering which agrees with $\mathcal{D}$ on a set A of at least $1 - O(\gamma)$ fraction of the nodes. We can then go through this clustering, and for each node in parallel, switch it to the other side if that improves the solution to produce the final clustering. Nodes in A won't get switched and will remain clustered exactly as in the optimum $\mathcal{D}$. The number of extra disagreements compared to $\mathcal{D}$ on edges amongst nodes in $V \setminus A$ is obviously at most the number of those edges which is $O(\gamma^2 n^2)$. For edges connecting a node $u \in V \setminus A$ to nodes in A, since we placed u on the "better" side, and A is placed exactly as in $\mathcal{D}$ in the final clustering, we can have at most $O(\gamma n)$ extra disagreements per node compared to $\mathcal{D}$ (this is the error introduced by the edges to the misplaced nodes in $V \setminus A$). Therefore we get a clustering with at most $\mathsf{OPT} + O(\gamma^2 n^2) = (1 + O(\gamma))\mathsf{OPT}$ disagreements.

Our k-clustering algorithm for $k > 2$ uses a similar high-level approach, but is more complicated. The main thing which breaks down compared to the $k = 2$ case is the following. For two clusters, if $\mathcal{D}$ has agreements on a large, i.e. $(1 - O(\gamma))$, fraction of edges incident on a node u (i.e. if $u \in A$ in the above notation), then we are guaranteed to place u exactly as in $\mathcal{D}$ based on the sample S (when we guess its correct clustering), since the other option will have much poorer agreement. This is not the case when $k > 2$, and one can get a large number of agreements by placing a node in say one of two possible clusters. Therefore, it does not seem possible to argue that each node in A is correctly placed, and then to use this to finish off the clustering.

However, what we *can* show is that nodes in A that are incorrectly placed, call this set B, must be in small clusters of $\mathcal{D}$, and thus are few in number. Moreover, every node in A that falls in one of the large clusters that we produce, is guaranteed to be correctly placed. (These facts are the content of Lemma 10.) The nodes in B still need to be clustered, and even a small additional number of mistakes per node in clustering them is more than we can afford. We get around this predicament by noting that nodes in B and $A \backslash B$ are in different sets of clusters in $\mathcal{D}$. It follows that we can cluster B recursively in new clusters (and we are making progress because B is clustered using fewer than k clusters). The actual algorithm must also deal with nodes outside A, and in particular decide which of these nodes are recursively clustered along with B.

With this intuition in place, we now proceed to the formal specification of the algorithm that gives a factor $(1+\varepsilon)$ approximation for MinDisAgree[k] in Figure 2. We will use a small enough absolute constant c_1 in the algorithm; the choice $c_1 = 1/20$ will work.

```
Algorithm MinDisAg($k, \varepsilon$):
Input: A labeling $\mathcal{L} : \binom{n}{2} \to \{+,-\}$ of the edges of the complete graph on vertex set $V = \{1, 2, \ldots, n\}$
Output: A $k$-clustering of the graph, i.e., a partition of $V$ into (at most) $k$ parts $V_1, V_2, \ldots, V_k$.

0. If $k = 1$, return the obvious 1-clustering.
1. Run the PTAS for MaxAgree[$k$] from previous section on input $\mathcal{L}$ with accuracy $\frac{\varepsilon^2 c_1^2}{32k^4}$.
   Let ClusMax be the $k$-clustering returned.
2. Set $\beta = \frac{c_1 \varepsilon}{16k^2}$. Pick a sample $S \subseteq V$ by drawing $\frac{5\log n}{\beta^2}$ vertices u.a.r with replacement.
3. ClusVal = 0; /* Keeps track of value of best clustering found so far*/
4. For each partition $\tilde{S}$ of $S$ as $S_1 \cup S_2 \cup \cdots \cup S_k$, perform the following steps:
   (a) Initialize the clusters $C_i \equiv S_i$ for $1 \le i \le k$.
   (b) For each $u \in V \setminus S$
      (i) For each $i = 1, 2, \ldots, k$, compute $\mathsf{pval}^{\tilde{S}}(u, i)$, defined to be $1/|S|$ times the number of
          agreements on edges connecting $u$ to nodes in $S$ if $u$ is placed in cluster $i$ along with $S_i$.
      (ii) Let $j_u = \arg\max_i \mathsf{pval}^{\tilde{S}}(u, i)$, and $\mathsf{val}^{\tilde{S}}(u) \stackrel{\text{def}}{=} \mathsf{pval}^{\tilde{S}}(u, j_u)$.
      (iii) Place $u$ in cluster $C_{j_u}$, i.e., $C_{j_u} \equiv C_{j_u} \cup \{u\}$.
   (c) Compute the set of large and small clusters as
          $\mathsf{Large} \equiv \{j \mid 1 \le j \le k,\ |C_j| \ge \frac{n}{2k}\}$, and $\mathsf{Small} \equiv \{1, 2, \ldots, k\} \setminus \mathsf{Large}$.
       Let $l = |\mathsf{Large}|$ and $s = k - l = |\mathsf{Small}|$. /* Note that $s < k$. */
   (d) Cluster $W \stackrel{\text{def}}{=} \bigcup_{j \in \mathsf{Small}} C_j$ into $s$ clusters using recursive call to algorithm MinDisAg($s, \varepsilon/3$).
         Let the clustering output by the recursive call be $W \equiv W_1' \cup W_2' \cup \cdots \cup W_s'$
         (where some of the $W_i'$'s may be empty)
   (e) Let $\mathcal{C}$ be the clustering comprising of the $k$ clusters $\{C_j\}_{j \in \mathsf{Large}}$ and $\{W_i'\}_{1 \le i \le s}$.
         If the number of agreements of $\mathcal{C}$ is at least ClusVal, update ClusVal to this value, and
         update ClusMin $\equiv \mathcal{C}$.
5. Output the better of the two clusterings ClusMax and ClusMin.
```

Figure 2: ***MinDisAg***(k, ε) algorithm

4.2 Performance analysis of the algorithm.

We now analyze the approximation guarantee of the above algorithm. We need some notation. Let $\mathcal{A} \equiv A_1 \cup A_2 \cup \cdots A_k$ be any k-clustering of the nodes in V. Define the function $\mathsf{val}^{\mathcal{A}} : V \to [0,1]$ as follows: $\mathsf{val}^{\mathcal{A}}(u)$ equals the fraction of edges incident upon node u whose labels agree with clustering $\mathcal{A}$ (i.e., we count negative edges that are cut by $\mathcal{A}$ and positive edges that lie within the same A_i for some i). Also define $\mathsf{disagr}(\mathcal{A})$ to be the number of disagreements of $\mathcal{A}$ w.r.t. labeling L. (Clearly $\mathsf{disagr}(\mathcal{A}) = \frac{n-1}{2}\sum_{u\in V}(1 - \mathsf{val}^{\mathcal{A}}(u))$.) For a node $u \in V$ and $1 \le i \le k$, let $\mathcal{A}^{(u,i)}$ denote the clustering obtained from $\mathcal{A}$ by moving u to A_i and leaving all other nodes untouched. We define the function $\mathsf{pval}^{\mathcal{A}} : V \times \{1,2,\ldots,k\} \to [0,1]$ as follows: $\mathsf{pval}^{\mathcal{A}}(u,i)$ equals the fraction of edges incident upon u that agree with the clustering $\mathcal{A}^{(u,i)}$.

In the following, we fix $\mathcal{D}$ to be any optimal k-clustering that partitions V as $V \equiv D_1 \cup D_2 \cup \cdots \cup D_k$. Let γ be defined to be $\mathsf{disagr}(\mathcal{D})/n^2$ so that the clustering $\mathcal{D}$ has γn^2 disagreements w.r.t. the input labeling L.

Call a sample S of nodes, each drawn uniformly at random with replacement, to be α-good if the nodes in S are distinct[3] and for each $u \in V$ and $i \in \{1,2,\ldots,k\}$,

$$|\mathsf{pval}^{\tilde{S}}(u,i) - \mathsf{pval}^{\mathcal{D}}(u,i)| \le \alpha \tag{3}$$

for the partition $\tilde{S}$ of S as $\cup_{i=1}^{k} S_i$ with $S_i = S \cap D_i$ (where $\mathsf{pval}^{\tilde{S}}(\cdot,\cdot)$ is as defined in the algorithm). The following lemma follows by a standard Chernoff and union bound argument similar to Lemma 5.[4]

Lemma 8 *The sample S picked in Step 2 is β-good with high probability (at least $1 - O(1/\sqrt{n})$).*

Therefore, in what follows we assume that the sample S is β-good. In the rest of the discussion, we focus on the run of the algorithm for the partition $\tilde{S}$ of S that agrees with the optimal partition $\mathcal{D}$, i.e., $S_i = S \cap D_i$. (All lemmas stated apply for this run of the algorithm, though we don't make this explicit in the statements.) Let $(C_1, C_2, \ldots, C_k)$ be the clusters produced by the algorithm at end of Step 4(c) on this run. Let's begin with the following simple observation.

Lemma 9 *Suppose a node $u \in D_s$ is placed in cluster C_r at the end of Step 4(b) for $r \ne s$, $1 \le r, s \le k$. Then $\mathsf{pval}^{\mathcal{D}}(u,r) \ge \mathsf{pval}^{\mathcal{D}}(u,s) - 2\beta = \mathsf{val}^{\mathcal{D}}(u) - 2\beta$.*

Proof: Note that since $u \in D_s$, $\mathsf{val}^{\mathcal{D}}(u) = \mathsf{pval}^{\mathcal{D}}(u,s)$. By the β-goodness of S (recall Inequality (3)), $\mathsf{pval}^{\tilde{S}}(u,s) \ge \mathsf{pval}^{\mathcal{D}}(u,s) - \beta$. Since we chose to place u in C_r instead of C_s, we must have $\mathsf{pval}^{\tilde{S}}(u,r) \ge \mathsf{pval}^{\tilde{S}}(u,s)$. By the β-goodness of S again, we have $\mathsf{pval}^{\mathcal{D}}(u,r) \ge \mathsf{pval}^{\tilde{S}}(u,r) - \beta$. Combining these three inequalities gives us the claim of the lemma. ■

Define the set of nodes of low value in the optimal clustering $\mathcal{D}$ as $T_{\mathsf{low}} \overset{\text{def}}{=} \{u \mid \mathsf{val}^{\mathcal{D}}(u) \le 1 - c_1/k^2\}$. The total number of disagreements is at least the number of disagreements induced by these low valued nodes, therefore

$$|T_{\mathsf{low}}| \le \frac{2k^2 \mathsf{disagr}(\mathcal{D})}{(n-1)c_1} = \frac{2k^2\gamma n^2}{(n-1)c_1} \le \frac{4k^2\gamma n}{c_1} \ . \tag{4}$$

[3]Note that in the algorithm we draw elements of the sample with replacement, but for the analysis, we can pretend that S consists of distinct elements, since this happens with high probability.

[4]Since our sample size is $\Omega(\log n)$ as opposed to $O(1)$ that was used in Lemma 5, we can actually ensure (3) holds for *every* vertex w.h.p.

The following key lemma asserts that the large clusters produced in Step 4(c) are basically correct.

Lemma 10 *Suppose $\gamma \le \frac{c_1}{16k^3}$. Let $\mathsf{Large} \subseteq \{1, 2, \ldots, k\}$ be the set of large clusters as in Step 4(c) of the algorithm. Then for each $i \in \mathsf{Large}$, $C_i \setminus T_{\mathsf{low}} \equiv D_i \setminus T_{\mathsf{low}}$, that is w.r.t. nodes of large value, C_i precisely agrees with the optimal cluster D_i.*

Proof: Let $i \in \mathsf{Large}$ be arbitrary. We will first prove the inclusion $C_i \setminus T_{\mathsf{low}} \subseteq D_i \setminus T_{\mathsf{low}}$. Suppose this is not the case and there exists $u \in C_i \setminus (D_i \cup T_{\mathsf{low}})$. Let $u \in D_j$ for some $j \neq i$. Since $u \notin T_{\mathsf{low}}$, we have $\mathsf{val}^{\mathcal{D}}(u) \ge 1 - c_1/k^2$, which implies $\mathsf{pval}^{\mathcal{D}}(u, j) \ge 1 - c_1/k^2$. By Lemma 9, this gives $\mathsf{pval}^{\mathcal{D}}(u, i) \ge 1 - c_1/k^2 - 2\beta$. Therefore we have

$$2(1 - c_1/k^2 - \beta) \quad \le \quad \mathsf{pval}^{\mathcal{D}}(u, i) + \mathsf{pval}^{\mathcal{D}}(u, j) \le 2 - \frac{|D_i| + |D_j| - 1}{n}$$

where the last step follows from the simple but powerful observation that each edge connecting u to a vertex in $D_i \cup D_j$ is correctly classified in exactly one of the two placements of u in the i'th and j'th clusters (when leaving every other vertex as in clustering $\mathcal{D}$). We conclude that both

$$|D_i|, |D_j| \le 2(\frac{c_1}{k^2} + \beta)n + 1 \ . \tag{5}$$

What we have shown is that if $u \in C_i \setminus (D_i \cup T_{\mathsf{low}})$, then $u \in D_j$ for some j with $|D_j| \le 2(c_1/k^2 + \beta)n + 1$. It follows that $|C_i \setminus (D_i \cup T_{\mathsf{low}})| \le 2(c_1/k + \beta k)n + k$. Therefore,

$$|D_i| \quad \ge \quad |C_i| - |T_{\mathsf{low}}| - 2(\frac{c_1}{k} + \beta k)n - k \ge \frac{n}{2k} - \frac{4k^2\gamma n}{c_1} - 2(\frac{c_1}{k} + \beta k)n - k > 2(\frac{c_1}{k^2} + \beta)n + 1$$

where the last step follows since $\gamma \le \frac{c_1}{16k^3}$, $k \ge 2$, $c_1 = 1/20$, and β is tiny. This contradicts (5), and so we conclude $C_i \setminus T_{\mathsf{low}} \subseteq D_i \setminus T_{\mathsf{low}}$.

Now for the other inclusion $D_i \setminus T_{\mathsf{low}} \subseteq C_i \setminus T_{\mathsf{low}}$. If a node $v \in D_i \setminus (C_i \cup T_{\mathsf{low}})$ is placed in C_q for $q \neq i$, then a similar argument to how we concluded (5) establishes $|D_i| \le 2(\frac{c_1}{k^2} + \beta)n + 1$, which is impossible since we have shown $D_i \supseteq C_i \setminus T_{\mathsf{low}}$, and hence $|D_i| \ge |C_i| - |T_{\mathsf{low}}| \ge \frac{n}{2k} - \frac{4k^2\gamma n}{c_1} > 2(\frac{c_1}{k^2} + \beta)n + 1$, where the last step follows using $\gamma \le \frac{c_1}{16k^3}$ and $k \ge 2$ for the choice $c_1 = 1/20$. ■

The next lemma states that there is a clustering which is very close to optimum which agrees exactly with our large clusters. This will enable us to find a near-optimal clustering by recursing on the small clusters to recluster them as needed, exactly as our algorithm does.

Lemma 11 *Assume $\gamma \le \frac{c_1}{16k^3}$. There exists a clustering $\mathcal{F}$ that partitions V as $V \equiv F_1 \cup F_2 \cup \cdots F_k$ that satisfies the following:*

(i) $F_i \equiv C_i$ for every $i \in \mathsf{Large}$

(ii) The number of disagreements of the clustering $\mathcal{F}$ is at most $\mathsf{disagr}(\mathcal{F}) \le \gamma n^2 \Big(1 + \frac{4k^2}{c_1}(\beta + \frac{2k^2\gamma}{c_1})\Big)$

Proof: Suppose $w \in T_{\mathsf{low}}$ is such that $w \in C_r$, $w \in D_s$ with $r \neq s$. Consider the clustering formed from $\mathcal{D}$ by performing the following in parallel for each $w \in T_{\mathsf{low}}$: If $w \in C_r$ and $w \in D_s$ for some $r \neq s$, move w to D_r. Let $\mathcal{F} \equiv F_1 \cup \cdots \cup F_k$ be the resulting clustering. By construction $F_i \cap T_{\mathsf{low}} \equiv C_i \cap T_{\mathsf{low}}$ for all i, $1 \le i \le k$. Since we only move nodes in T_{low}, clearly $F_i \setminus T_{\mathsf{low}} \equiv D_i \setminus T_{\mathsf{low}}$

for $1 \leq i \leq k$. By Lemma 10, $C_i \setminus T_{\mathsf{low}} \equiv D_i \setminus T_{\mathsf{low}}$ for $i \in \mathsf{Large}$. Combining all these equalities we conclude that $F_i \equiv C_i$ for each $i \in \mathsf{Large}$.

Now the only extra edges that the clustering $\mathcal{F}$ can get wrong compared to $\mathcal{D}$ are those incident upon nodes in T_{low}, and therefore

$$\mathsf{disagr}(\mathcal{F}) - \mathsf{disagr}(\mathcal{D}) \leq (n-1) \sum_{w \in T_{\mathsf{low}}} (\mathsf{val}^{\mathcal{D}}(w) - \mathsf{val}^{\mathcal{F}}(w)) \tag{6}$$

If a node w belongs to the same cluster in $\mathcal{F}$ and $\mathcal{D}$ (i.e., we did not move it), then since no node outside T_{low} is moved in obtaining $\mathcal{F}$ from $\mathcal{D}$, we have

$$\mathsf{val}^{\mathcal{F}}(w) \geq \mathsf{val}^{\mathcal{D}}(w) - |T_{\mathsf{low}}|/(n-1) \ . \tag{7}$$

If we moved a node $w \in T_{\mathsf{low}}$ from D_s to D_r, then by Lemma 9 we have $\mathsf{pval}^{\mathcal{D}}(w, r) \geq \mathsf{val}^{\mathcal{D}}(w) - 2\beta$. Therefore for such a node w

$$\mathsf{val}^{\mathcal{F}}(w) \quad \geq \quad \mathsf{pval}^{\mathcal{D}}(w, r) - |T_{\mathsf{low}}|/(n-1) \geq \mathsf{val}^{\mathcal{D}}(w) - 2\beta - |T_{\mathsf{low}}|/(n-1) \ . \tag{8}$$

Combining (6), (7) and (8), we can conclude $\mathsf{disagr}(\mathcal{F}) - \mathsf{disagr}(\mathcal{D}) \leq (n-1)|T_{\mathsf{low}}|(2\beta + \frac{|T_{\mathsf{low}}|}{n-1})$. The claim now follows using the upper bound on $|T_{\mathsf{low}}|$ from (4) (and using $n^2/(n-1)^2 \leq 2$). ■

Lemma 12 *If the optimal clustering $\mathcal{D}$ has γn^2 disagreements for $\gamma \leq \frac{c_1}{16k^3}$, then the clustering* $\mathsf{ClusMin}$ *found by the algorithm has at most $\gamma n^2(1+\varepsilon/3)(1 + 4k^2\beta/c_1 + 8k^4\gamma/c_1^2)$ disagreements.*

Proof: We note that when restricted to the set of all edges except those entirely within W, the set of agreements of the clustering $\mathcal{C}$ in Step 4(e) coincides precisely with that of $\mathcal{F}$. Let n_1 be the number of disagreements of $\mathcal{F}$ on edges that lie within W and let n_2 be the number of disagreements on all other edges. Since W is clustered recursively, we have the number of disagreements in $\mathcal{C}$ is at most $n_2 + n_1(1+\varepsilon/3) \leq (n_1+n_2)(1+\varepsilon/3)$. The claim follows from the bound on $n_1 + n_2$ from Lemma 11, Part (ii). ■

Theorem 13 *For every $\varepsilon > 0$, algorithm* **MinDisAg**(k, ε) *delivers a clustering with number of disagreements within a factor $(1+\varepsilon)$ of the optimum.*

Proof: Let $\mathsf{OPT} = \gamma n^2$ be the number of disagreements of an optimal clustering. The solution $\mathsf{ClusMax}$ returned by the maximization algorithm has at most $\mathsf{OPT} + \frac{\varepsilon^2 c_1^2 n^2}{32k^4} = \gamma n^2\Big(1 + \frac{\varepsilon^2 c_1^2}{32k^4\gamma}\Big)$ disagreements. The solution $\mathsf{ClusMin}$ has at most $\gamma n^2(1+\varepsilon/3)(1+4k^2\beta/c_1+8k^4\gamma/c_1^2))$ disagreements. If $\gamma > \frac{\varepsilon c_1^2}{32k^4}$, the former is within $(1+\varepsilon)$ of the optimal. If $\gamma \leq \frac{\varepsilon c_1^2}{32k^4}$ (which also satisfies the requirement $\gamma \leq c_1/16k^3$ we had in Lemma 12), the latter clustering $\mathsf{ClusMin}$ achieves approximation ratio $(1+\varepsilon/3)(1+\varepsilon/2) \leq (1+\varepsilon)$ (recall that $\beta \leq \frac{\varepsilon c_1}{16k^2}$). Thus the better of these two solutions is always an $(1+\varepsilon)$ approximation. ■

To conclude Theorem 7, we examine the running time of **MinDisAg**. Step 4 will be run for $k^{|S|} = n^{O(k^4/\varepsilon^2)}$ iterations. During each iteration, the placement of vertices is done in $O(n \log n)$ time. Finally, observe that there is always at least one large cluster, therefore the recursive call is always done on at most $(k-1)$ clusters. It follows that the running time of **MinDisAg**(k, ε) can be described from the recurrence $T(k, \varepsilon) \leq n^{O(k^4/\varepsilon^2)}(n \log n + \cdot T(k-1, \varepsilon/3))$ from which we derive that the total running time is bounded by $n^{O(9^k/\varepsilon^2)} \log n$.

5 Complexity on general graphs

So far, we have discussed the MaxAgree[k] and MinDisAgree[k] problems on complete graphs. In this section, we note some results on the complexity of these problems when the graph can be arbitrary. As we will see, the problems become much harder in this case.

Theorem 14 *There is a polynomial time factor* 0.878 *approximation algorithm for* MaxAgree[2] *on general graphs. For every $k \geq 3$, there is a polynomial time factor* 0.7666 *approximation algorithm for* MaxAgree[k] *on general graphs.*

Proof: The bound for 2-clusters case follows from the Goemans-Williamson algorithm for Max CUT modified in the obvious way to account for the positive edges. The bound for $k \geq 3$ is obtained by Swamy [18] who also notes that slightly better bounds are possible for $3 \leq k \leq 5$. ■

We note that in light of the recent hardness result for Max CUT [15], the above guarantee for MaxAgree[2] is likely the best possible.

Theorem 15 *There is a polynomial time $O(\sqrt{\log n})$ approximation algorithm for* MinDisAgree[2] *on general graphs. For $k \geq 3$,* MinDisAgree[k] *on general graphs cannot be approximated within any finite factor.*

Proof: The bound for 2-clustering follows by the simple observation that MinDisAgree[2] on general graphs reduces to Min 2CNF Deletion, i.e., given an instance of 2SAT, determining the minimum number of clauses that have to be deleted to make it satisfiable. The latter problem admits an $O(\sqrt{\log n})$ approximation algorithm [1]. The result on MinDisAgree[k] for $k \geq 3$ follows by a reduction from k-coloring. When $k \geq 3$, it is NP-hard to tell if a graph is k-colorable, and thus even given an instance of MinDisAgree[k] with only negative edges, it is NP-hard to determine if the optimum number of disagreements is zero or positive. ■

References

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. $O(\sqrt{\log n})$ approximation algorithms for Min Uncut, Min 2CNF deletion, and directed cut problems. In *Proceedings of the 37th ACM Symposium on Theory of Computing (STOC)*, pages 573–581, 2005.

[2] N. Ailon, M. Charikar, and A. Newman. Aggregating Inconsistent Information: Ranking and Clustering. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 684–693, 2005.

[3] N. Alon, K. Makarychev, Y. Makarychev, and A. Naor. Quadratic forms on graphs. In *Proceedings of the 37th ACM Symposium on Theory of Computing (STOC)*, pages 486–493, 2005.

[4] S. Arora, E. Berger, E. Hazan, G. Kindler, and S. Safra. On non-approximability for quadratic programs. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science (FOCS)*, 2005.

[5] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning, Special Issue on Clustering*, 56:89–113, 2004.

[6] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *J Comp. Biol.*, 6:281–97, 1999.

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

[8] M. Charikar and A. Wirth. Maximizing quadratic programs: extending Grothendieck's inequality. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 54–60, 2004.

[9] W. Fernandez de la Vega, M. Karpinski, C. Kenyon, and Y. Rabani. Approximation schemes for clustering problems. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 50–58, 2003.

[10] W. Fernandez de la Vega and C. Kenyon. A randomized approximation scheme for metric max-cut. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 468–471, 1998.

[11] E. Demaine and N. Immorlica. Correlation clustering with partial information. In *Proc. of 6th APPROX*, pages 1–13, 2003.

[12] D. Emanuel and A. Fiat. Correlation clustering—minimizing disagreements on arbitrary weighted graphs. In *Proc. of 11th ESA*, pages 208–20, 2003.

[13] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, July 1998.

[14] P. Indyk. A sublinear-time approximation scheme for clustering in metric spaces. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 154–159, 1999.

[15] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for Max Cut and other 2-variable CSPs. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 146–154, 2004.

[16] A. Nemirovski, C. Roos, and T. Terlaky. On maximization of quadratic form over intersection of ellipsoids with common center. *Mathematical Programming*, 86(3):463–473, 1999.

[17] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. In *Proc. of 28th Workshop on Graph Theory (WG)*, pages 379–90, 2002.

[18] C. Swamy. Correlation Clustering: Maximizing agreements via semidefinite programming. In *Proc. of 15th SODA*, pages 519–20, 2004.