Near Optimal LP Rounding Algorithm for Correlation Clustering on Complete and Complete k-partite Graphs

Shuchi Chawla
University of Wisconsin–Madison

Konstantin Makarychev
Microsoft Research

Tselil Schramm
UC Berkeley

Grigory Yaroslavtsev
University of Pennsylvania

ABSTRACT

We give new rounding schemes for the standard linear programming relaxation of the correlation clustering problem, achieving approximation factors almost matching the integrality gaps:

- For complete graphs our approximation is $2.06 - \varepsilon$, which almost matches the previously known integrality gap of 2.
- For complete k-partite graphs our approximation is 3. We also show a matching integrality gap.
- For complete graphs with edge weights satisfying triangle inequalities and probability constraints, our approximation is 1.5, and we show an integrality gap of 1.2.

Our results improve a long line of work on approximation algorithms for correlation clustering in complete graphs, previously culminating in a ratio of 2.5 for the complete case by Ailon, Charikar and Newman (JACM'08). In the weighted complete case satisfying triangle inequalities and probability constraints, the same authors give a 2-approximation; for the bipartite case, Ailon, Avigdor-Elgrabli, Liberty and van Zuylen give a 4-approximation (SICOMP'12).

1. INTRODUCTION

We study the *correlation clustering* problem – given inconsistent pairwise similarity/dissimilarity information over a set of objects, our goal is to partition the vertices into an *arbitrary* number of clusters that match this information as closely as possible. The task of clustering is made interesting by the fact that the similarity information is inherently noisy. For example, we may be asked to cluster u and v together, and v and w together, but u and w separately. In this case there is no clustering that matches the data exactly. The optimal clustering is the one that differs from the given constraints at the fewest possible number of pairs, and it may use anywhere between one and n clusters. In some contexts this problem is also known as *cluster editing*: given a graph between objects where every pair deemed similar is connected by an edge, add or remove the fewest number of edges so as to convert the graph into a collection of disjoint cliques.

STOC'15, June 14–17, 2015, Portland, Oregon, USA.

http://dx.doi.org/10.1145/2746539.2746604.

Correlation clustering is quite different from other common clustering objectives in that the given data is qualitative (similar versus dissimilar pairs) rather than quantitative (e.g. objects embedded in a metric space). As such, it applies to many different problems that arise in machine learning, biology, data mining and other areas. From a learning perspective, correlation clustering is essentially an agnostic learning problem: the goal is to fit a classifier from a certain concept class (namely all clusterings) as best as possible to noisy examples (namely the pairwise similarity information). The correlation clustering objective has been successfully employed for a number of learning problems, for example: coreference resolution [12, 13, 23], where the goal is to determine which references in a news article refer to the same object; cross-lingual link detection [26], where the goal is to find news articles in different languages that report on the same event; email clustering by topic or relevance; and image segmentation [29]. In biology, the problem of clustering gene expression patterns can be cast into the framework of correlation clustering [7, 5]. Another application, arising in data mining, is that of aggregating inconsistent clusterings taken from different sources [16]. In this setting, the cost of an aggregate clustering is the sum over pairs of objects of the fraction of clusterings that it differs from. This special case of correlation clustering is known as the *consensus clustering* problem.

In many of the above applications, we have access to a binary classifier that takes in pairs of objects and returns a "similar" or "dissimilar" label. We can interpret this information in the form of a complete graph with edges labeled "+" (denoting similarity) and "−" (denoting dissimilarity). The correlation clustering problem can then be restated as one of producing a clustering such that most of the "+" edges are within clusters and most "−" edges cross different clusters. In some cases, it may not be possible to compare all pairs of objects, leaving "missing" edges so that the underlying graph is not complete. However, correlation clustering on general graphs is equivalent to the multicut problem [14], and obtaining any constant factor approximation is Unique-Games hard [9].[1] Still, it is possible to get approximations for some practical cases with missing edges; for example, when the underlying graph is a complete k-partite graph. Ailon, Avigdor-Elgrabli, Liberty, and van Zuylen [2] give several applications of complete bipartite correlation clustering.

Since its introduction a decade ago by Bansal, Blum, and Chawla [6], correlation clustering has gained a lot of prominence within the theory and learning communities (see, e.g., the survey by Wirth [29], and references therein) and has become one of the textbook examples in the design of approximation algorithms

[1]The best known algorithm due to Demaine, Emanuel, Fiat, and Immorlica [14] gives an $O(\log n)$ approximation.

(Williamson and Shmoys [28] consider the correlation clustering problem with the maximization objective). Bansal et al. gave the first constant factor approximation algorithm for the problem on complete graphs. The factor has since then been improved several times, culminating in a factor of 2.5 for complete graphs due to Ailon, Charikar, and Newman [3], which relies on a natural LP formulation of the problem. On the other hand, the problem is known to be APX-hard [14], and the best known integrality gap of the LP is 2 [8], leaving a 20% margin for improvement.

1.1 Our Results

In this paper, we nearly close the gap between the approximation ratio and the integrality gap for complete graphs and complete k-partite graphs: For the correlation clustering problem on complete graphs, we obtain a $(2.06 - \varepsilon)$-approximation for some fixed ε, nearly matching an integrality gap of 2 [8]. For the correlation clustering problem on complete k-partite graphs, we obtain a 3-approximation and exhibit an integrality gap instance with a gap of 3. The previously best known algorithm for the bipartite variant of the problem due to Ailon et al. [2] gives a 4-approximation.

THEOREM 1. *There is a deterministic polynomial-time algorithm for the Correlation Clustering Problem that gives a $(2.06 - \varepsilon)$-approximation for complete graphs, where ε is some fixed constant smaller than 0.01, and a 3-approximation for complete k-partite graphs.*

We also study a weighted variant of the problem. In this variant, each edge has a positive weight λ^+_{uv} and a negative weight λ^-_{uv}, the goal is to minimize the total weight of violated edges (for details see the full version). We show that Weighted Correlated Clustering is equivalent to the unweighted Correlated Clustering on complete graphs if $\lambda^+_{uv} + \lambda^-_{uv} = 1$ for every $(u, v) \in E$. Gionis, Mannila and Tsaparas [17] introduce a natural special case of this problem in which the negative weights satisfy the triangle inequality (for every u, v, w it holds that $\lambda^-_{uv} \leq \lambda^-_{uw} + \lambda^-_{wv}$). For this problem – Weighted Correlation Clustering with Triangle Inequalities – we give a 1.5-approximation algorithm, improving on the 2-approximation of Ailon et al. [3]. Our proof of the last result is computer assisted.

The main technical contribution of our paper is an approach towards obtaining a tight rounding scheme given an LP solution. At a high level our algorithm is similar to that of Ailon et al. [3], but we perform the actual rounding decisions in a novel way, using carefully designed functions of the LP solution to get rounding probabilities for edges in the graph, allowing us to obtain a near-optimal approximation ratio. We emphasize that although we employ a lengthy and complicated analysis to prove that our rounding scheme achieves a $(2.06 - \varepsilon)$-approximation, our algorithm itself is very simple and runs in time $O(n^2)$ given the LP solution. The linear programming relaxation that we use has been studied very extensively and heuristic approaches have been developed for solving it [15].

We demonstrate our technique for correlation clustering in complete graphs, in complete k-partite graphs, and in the special case of weighted edges satisfying triangle inequality constraints. In each case, we obtain significant improvements over the previously best known results, and nearly or exactly match the integrality gap of the LP. When we are unable to match the integrality gap, we prove a lower bound on the ratio that may be achieved by any functions within our rounding scheme. Our results and a comparison with the previous work are summarized in Table 1.

1.2 Our Methodology

As is the case for many graph partitioning problems, the correlation clustering objective can be captured in the form of a linear program over variables that encode lengths of edges. A long edge signifies that its endpoints should be placed in different clusters, and a short edge signifies that its endpoints should be in the same cluster. For consistency, edge lengths must satisfy the triangle inequality.

A natural approach to rounding this relaxation is to interpret each edge's length, x_{uv}, as the probability with which it should be cut. The challenge is to ensure consistency. For example, consider a triangle with two positive edges and one negative edge where the negative edge has LP length $\frac{1}{2}$. If we first cut the negative edge with probability $\frac{1}{2}$, then in order to return a consistent clustering we are forced to cut one of the positive edges. In this way, an independent decision to cut or not cut one edge may force a decision on a different edge, resulting in "collateral damage." Ailon, Charikar, and Newman [3] give a simple rounding algorithm and a charging scheme that cleanly bounds the cost of this collateral damage. Their algorithm picks a random vertex w in the graph and rounds every edge (w, u) incident on this "pivot" with probability equal to the length of the edge; vertices u corresponding to the edges (w, u) that are not cut by this procedure form w's cluster; this cluster is then removed from the graph, and the algorithm recurses on the remaining graph. This approach gives the best previously known approximation ratio for the correlation clustering problem on complete graphs, a factor of 2.5.

Our main technical contribution is a more subtle treatment of the probability of cutting an edge: rather than cutting edge (u, v) with probability x_{uv}, we cut (u, v) with probability given by some function $f(x_{uv})$ (this idea was previously used by Ailon in his algorithm for ranking aggregation [1]). In a departure from all of the other LP-rounding algorithms for correlation clustering, we use different rounding functions for positive and negative edges. Though it may at first be surprising that the latter distinction can be helpful, we remark that positive and negative constraints do not behave symmetrically. For example, in a triangle with two positive edges and one negative edge, the negative edge forces an inconsistency: any clustering of this triangle must violate at least one constraint. On the other hand, in a triangle with two negative edges and one positive edge, there is a valid clustering that does not violate any constraints. Thus, we see that the positive and negative edges behave differently, and we prove that rounding positive and negative edges of the same LP length with different probabilities gives a correspondingly better approximation ratio.

In some cases, this distinction leads to results that run counter to our intuition. One might expect that negative edges should be cut with higher probability than positive edges. However, this is not always the case. It turns out that it helps to cut long positive edges with probability 1, because we can charge them to the LP. On the other hand, it pays to be careful about cutting long negative edges, because this might cause too much collateral damage to other edges.

Our methodology for selecting the rounding functions f^+ and f^- is interesting in its own right. By regarding the cost of the algorithm on each kind of triangle as a polynomial in x_{uv}, x_{vw}, x_{uw} and in $f^{\pm}(x_{uv})$, $f^{\pm}(x_{uv})$, $f^{\pm}(x_{uv})$, then characterizing these multivariate polynomials, we are able to obtain analytic upper and lower bounds on f^+ and f^-. While this does not force our choices of f^+ or f^-, it suggests natural candidate functions that can then be further analyzed. This same worst-case polynomial identification and bounding approach yields lower bounds on the best possible approximation ratio that can be achieved by a similar algorithm.

Approximation Algorithms for Correlation Clustering				
	PREVIOUS FACTOR	OUR FACTOR	INTEGRALITY GAP	LIMITATION
Complete	$\approx 10^4$ [6], 4 [8], 2.5 [3, 27]	≈ 2.06, Thm 1	2 [8]	2.025 ($\star$)
Triangle Inequality	3 [17], 2 [3, 27]	1.5 ($\star$)	1.2 ($\star$)	1.5 ($\star$)
Bipartite	11 [5],4 [2]	3, Thm 1	3 ($\star$)	—
K-partite	—	3, Thm 1	3 ($\star$)	—

Table 1: Previous and our approximation factors, integrality gaps and limitations of our approach. A ($\star$) marks a result that we have deffered to the full version.

1.3 Other Related Work

As mentioned above, there has been a series of works giving constant factor approximations for correlation clustering in complete graphs [6, 8, 3]. A modified version of the Ailon et al. [3] 3-approximation algorithm can be used as a basis for parallel algorithms [11]. Van Zuylen et al. [27] showed that the 2.5-approximation of Ailon et al. can be derandomized without any loss in approximation factor. Correlation clustering on complete bipartite graphs was first studied by Amit [5], who presents an 11-approximation. This was subsequently improved to a 4-approximation by Ailon et al. [2]. For complete graphs with weights satisfying triangle inequalities a 3-approximation was obtained by Gionis et al. [17] and a 2-approximation by Ailon et al. [3]. These prior works are summarized in Table 1. In general graphs, the problem can be approximated to within a factor of $O(\log n)$, and because it is equivalent to the multicut problem, this is suspected to be the best possible [8, 14].

Bansal et al. [6] also studied an alternative version of the problem in which the objective is to approximately maximize the number of edges that the clustering gets correct: that is, the number of "+" edges inside clusters and "−" edges going across clusters. They noted that this "MaxAgree" version can be trivially approximated to within a factor of 2 in arbitrary weighted graphs, and presented a polynomial time approximation scheme for the version on complete graphs. Subsequently, Swamy [24] and Charikar, Guruswami and Wirth [8] developed an improved SDP-based approximation for the MaxAgree problem on arbitrary weighted graphs. MaxAgree is known to be hard to approximate within a factor of $80/79$ for both the unweighted (complete graph) and weighted versions [8, 25].

A number of other variants of the correlation clustering problem have been studied. Giotis and Guruswami [18] and Karpinski et al. [19] studied the variant where the solution is stipulated to contain only a few (constant number of) clusters, and under that constraint presented polynomial time approximation schemes. Mathieu, Sankur, and Schudy [21] studied the online version of the problem and give an $O(n)$-competitive algorithm, which is also the best possible within constant factors. Mathieu and Schudy [22] introduced a semi-random model, in which every graph is generated as follows: start with an arbitrary planted partitioning of the graph into clusters, set labels consistent with the planted partitioning on all edges, then independently pick every edge into a random subset of *corrupted* edges with probability p, and let the adversary arbitrarily change labels on the corrupted edges. Mathieu and Schudy [22] showed how to get $1 + o(1)$ approximation under very mild assumptions on p. Their result was extended to other semi-random models in [20, 10]. The problem was studied in another interesting stochastic model in [4].

1.4 Organization

In Section 2, we formally describe and generalize the algorithm and analytical framework of Ailon et al. [3] to accomodate our algorithm and analysis, and lay the groundwork for our analysis with observations about algorithms within this framework. In Section 3, we give the analysis that leads to our choices of rounding functions. Section 4 gives an overview of the analysis of the triples for complete correlation clustering, as well as pictoral proofs of the main theorem. Appendix A contains the analytical proof for the k-partite correlation clustering algorithm.

The full version contains all further results, including the analytical proof of the $(2.06 - \varepsilon)$-approximation for the complete case, the proof of a lower bound of 2.025 on the approximation ratio of any algorithm within our framework, new integrality gaps for the k-partite case and the weighted case with triangle inequalities, our results for the weighted versions of the problem, and a derandomization of our algorithm.

2. APPROXIMATION ALGORITHM

In this section, we present an approximation algorithm for the Correlation Clustering Problem that works both for complete graphs and complete k-partite graphs. For the weighted case of the problem, we refer the reader to the full version. We denote the set of positive edges by E^+ and the set of negative edges by E^-. The algorithm is based on the approach of Ailon et al. [3]. It iteratively finds clusters and removes them from the graph. Once all vertices are clustered, the algorithm outputs all found clusters and terminates.

Initially, the algorithm marks all vertices as active. At step t, it picks a random pivot w among active vertices, and then adds each active vertex u to S_t with probability $(1 - p_{uw})$ independently of other vertices. Then, the algorithm removes the cluster S_t from the graph and marks all vertices in S_t as inactive. The probability $(1 - p_{uw})$ depends on the LP solution and the type of the pair (u, w): we set $p_{uw} = f^+(x_{uw})$, if (u, w) is a positive edge; $p_{uw} = f^-(x_{uw})$, if (u, w) is a negative edge; and $p_{uw} = f^\circ(x_{uw})$, if there is no edge between u and w. Here, x_{uv} is the LP variable corresponding to the pair (u, v) (we describe the LP in a moment) and f^+, f^-, and f° are special functions which we will define later. Below we give pseudo-code for the algorithm.

Input: Graph G, LP solution $\{x_{uv}\}_{u,v \in V}$.
Output: Partitioning of vertices into disjoint sets.

- Let $V_0 = V$ be the set of active vertices; let $t = 0$.
- while ($V_t \neq \varnothing$)
  - Pick a pivot $w_t \in V_t$ uniformly at random.
  - For each vertex $u \in V_t$, set p_{uw}:

$$p_{uw} = \begin{cases} f^+(x_{uw}), \text{ if } (u,v) \text{ is a positive edge;} \\ f^-(x_{uw}), \text{ if } (u,v) \text{ is a negative edge;} \\ f^\circ(x_{uw}), \text{ if no edge between } u \text{ and } v. \end{cases} \quad (1)$$

  - For each vertex $u \in V_t$, add u to S_t with probability $(1 - p_{uw})$ independently of all other vertices.

– Remove S_t from V_t i.e., let $V_{t+1} = V_t \setminus S_t$. Let $t = t + 1$.

- Output $S_0, \ldots, S_T$, where T is the index of the last iteration of the algorithm.

This algorithm is probabilistic. We show how to derandomize it in the full version. The main new ingredient of our algorithm is a procedure for picking the probabilities p_{uw}. To compute these probabilities, we use the standard LP relaxation introduced by [8]. We first formulate an integer program for Correlation Clustering. For every pair of vertices u and v, we have a variable $x_{uv} \in \{0, 1\}$ that is equal to the distance between u and v in the "multicut metric": $x_{uv} = 0$ if u and v are in the same cluster; and $x_{uv} = 1$ if u and v are in different clusters. Variables x_{uv} satisfy the triangle inequality constraints (3). They are also symmetric, i.e. $x_{uv} = x_{vu}$. Instead of writing the constraint $x_{uv} = x_{vu}$, we have only one variable for each edge (u, v). We refer to this variable as x_{uv} or x_{vu}. The IP objective is to minimize the number of violated constraints. We write it as follows: $\min \sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$. Note that a term x_{uv} in the first sum equals 1 if and only if the corresponding positive edge (u, v) is cut; and a term $(1 - x_{uv})$ in the second sum equals 1 if and only if the corresponding negative edge (u, v) is contracted. Thus, this integer program is exactly equivalent to the Correlation Clustering Problem. We relax the integrality constraints $x_{uv} \in \{0, 1\}$ and obtain the following LP.

$$\min \sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv}) \tag{2}$$

$$x_{uv} + x_{vw} \geq x_{uw} \qquad \text{for all } u, v, w \in V \tag{3}$$

$$x_{uu} = 0 \qquad \text{for all } u \in V \tag{4}$$

$$x_{uv} \in [0, 1] \qquad \text{for all } u, v \in V \tag{5}$$

The approximation ratio of the algorithm depends on the set of functions $\{f^+, f^-, f^\circ\}$ we use for rounding. We explain how we pick these functions in Section 3. In Appendix A, we analyze the specific rounding functions that give improved approximation guarantees for complete k-partite graphs, and in the full version we analyze the rounding functions that give improved approximation guarantees for complete graphs (a pictorial proof is given in Section 4). We prove the following theorems.

THEOREM 2. *For complete graphs, the approximation algorithm with rounding functions*

$$f^+(x) = \begin{cases} 0, & \text{if } x < a \\ \left(\frac{x-a}{b-a}\right)^2, & \text{if } x \in [a, b] \\ 1 & \text{if } x \geq b \end{cases}, \qquad f^-(x) = x,$$

gives a $(2.06 - \varepsilon)$-approximation for $a = 0.19$ and $b = 0.5095$, and a constant ε with $0 < \varepsilon < 0.01$.

The proof of Theorem 2 is given in the full version.

THEOREM 3 (SEE SECTION A). *For complete k-partite graphs, the approximation algorithm with rounding functions*

$$f_3^+(x) = \begin{cases} 0, & \text{if } x < \frac{1}{3} \\ 1 & \text{if } x \geq \frac{1}{3} \end{cases},$$

$$f_3^-(x) = x,$$

$$f_3^\circ(x) = \begin{cases} \frac{3}{2}x & \text{if } x \leq \frac{2}{3} \\ 1 & \text{if } x > \frac{2}{3} \end{cases},$$

gives a 3-approximation.

Note that the integrality gap for complete graphs is 2 [8], and the integrality gap for complete k-partite graphs is 3 (see the full version for a proof). So the algorithm for complete k-partite graphs optimally rounds the LP; the algorithm for complete graphs nearly optimally rounds the LP.

2.1 Analysis

In this section, we prove a general statement – Lemma 4 – that asserts that the approximation ratio of the algorithm is at most α if a certain condition (depending on α) holds for every triple of vertices $u, v, w \in V$. We shall assume that $f(0) = 0$ for each $f \in \{f^+, f^-, f^\circ\}$, and, particularly, that the algorithm always puts the pivot w_t in the set S_t.

Consider step t of the algorithm. At this step, the algorithm finds and removes set S_t from the graph. Observe, that if $u \in S_t$ or $v \in S_t$, then the constraint (u, v) is either violated or satisfied right after step t. Specifically, if (u, v) is a positive edge, then the constraint (u, v) is violated if exactly one of the vertices – u or v – is in S_t. If (u, v) is a negative constraint, then (u, v) is violated if both u and v are in S_t. Denote the number of violated constraints at step t by ALG_t. Then,

$$ALG_t = \sum_{\substack{(u,v)\in E^+ \\ u,v\in V_t}} (\mathbb{1}(u \in S_t; v \notin S_t) + \mathbb{1}(u \notin S_t; v \in S_t)) + \sum_{\substack{(u,v)\in E^- \\ u,v\in V_t}} \mathbb{1}(u \in S_t; v \in S_t).$$

Here $\mathbb{1}(\mathcal{E})$ denotes the indicator function of the event $\mathcal{E}$. We want to charge the cost of constraints violated at step t to the LP cost of edges removed at step t. The LP cost of edges removed at this step equals

$$LP_t = \sum_{\substack{(u,v)\in E^+ \\ u,v\in V_t}} \mathbb{1}(u \in S_t \text{ or } v \in S_t) x_{uv} + \sum_{\substack{(u,v)\in E^- \\ u,v\in V_t}} \mathbb{1}(u \in S_t \text{ or } v \in S_t)(1 - x_{uv}).$$

Note, that $\sum_{t=0}^{T} LP_t = LP$, since every edge is removed from the graph exactly once (compare the expression above with the objective function (2)). If we show that $\mathbb{E}[ALG_t] \leq \alpha \mathbb{E}[LP_t]$ for all t, then we will immediately get an upper bound on the expected total cost of the clustering:

$$\mathbb{E}[ALG] = \mathbb{E}\Big[\sum_{t=0}^{T} ALG_t\Big] \leq \alpha \mathbb{E}\Big[\sum_{t=0}^{T} LP_t\Big] = \alpha LP.$$

Here, we use that $X_s = \sum_{t=0}^{s} \left(\alpha LP_t - ALG_t\right)$ is a submartingale (i.e., $\mathbb{E}[X_{s+1} \mid X_s] \geq X_s$) , and T is a stopping time.

Let $e.cost_w(u, v)$ be the conditional probability of violating the constraint (u, v) given that the pivot w_t is w (assuming $u, v, w \in$

V_t, and $u \neq v$). We say that $e.cost_w(u,v)$ is the expected cost of the constraint (u,v) given pivot w. Similarly, let $e.lp_w(u,v)$ be the conditional probability of removing the edge (u,v) given the pivot is w multiplied by the LP cost of the edge (u,v). (The LP cost equals x_{uv} for positive edges; and $(1-x_{uv})$ for negative edges.) That is,

$$e.cost_w(u,v) = \\ = \begin{cases} p_{uw}(1-p_{vw}) + (1-p_{uw})p_{vw}, & \text{if } (u,v) \in E^+; \\ (1-p_{uw})(1-p_{vw}), & \text{if } (u,v) \in E^-; \\ 0, & \text{if } (u,v) \notin E; \end{cases} \quad (6)$$

and

$$e.lp_w(u,v) = \\ = \begin{cases} (1-p_{uw}p_{vw})x_{uv}, & \text{if } (u,v) \in E^+; \\ (1-p_{uw}p_{vw})(1-x_{uv}), & \text{if } (u,v) \in E^-; \\ 0, & \text{if } (u,v) \notin E. \end{cases} \quad (7)$$

The expressions above do not depend on the set of active vertices V_t. The cut probabilities p_{uw} and p_{vw} are defined by the algorithm (see equation (1)). Note that $e.cost_u(u,v)$, $e.cost_v(u,v)$, $e.lp_u(u,v)$, and $e.lp_v(u,v)$ are well defined. We also formally define $e.cost_w(u,u)$ and $e.lp_w(u,u)$ using formulas (6) and (7). In the analysis of the algorithm for complete graphs, we assume that each vertex u has a positive self-loop, thus $e.cost_w(u,u) = 2(1-p_{uw})p_{uw}$, $e.lp_w(u,u) = 0$.

We now write $\mathbb{E}[ALG_t]$ and $\mathbb{E}[LP_t]$ in terms of $e.cost$ and $e.lp$:

$$\mathbb{E}[ALG_t \mid V_t] = \sum_{\substack{(u,v)\in E \\ u,v\in V_t}} \Big(\frac{1}{|V_t|} \sum_{w\in V_t} e.cost_w(u,v)\Big) \\ = \frac{1}{2|V_t|} \sum_{u,v,w\in V_t : u\neq v} e.cost_w(u,v);$$

$$\mathbb{E}[LP_t \mid V_t] = \sum_{\substack{(u,v)\in E \\ u,v\in V_t}} \Big(\frac{1}{|V_t|} \sum_{w\in V_t} e.lp_w(u,v)\Big) \\ = \frac{1}{2|V_t|} \sum_{u,v,w\in V_t : u\neq v} e.lp_w(u,v).$$

We divided the expressions on the right hand side by 2, because, in the sum, we count every $e.cost_w(u,v)$ and $e.lp_w(u,v)$ twice (e.g., the first sum contains the terms $e.cost_w(u,v)$ and $e.cost_w(v,u)$). We now add terms $e.cost_w(u,u)$ to the first sum and terms $e.lp_w(u,u)$ to the second sum. Then, we group all terms containing u, v, and w together. Note that $e.cost_w(u,u) \geq 0$ and $e.lp_w(u,u) = 0$. We get

$$\mathbb{E}[ALG_t \mid V_t] \leq \quad (8)$$
$$\leq \frac{1}{6|V_t|} \sum_{u,v,w\in V_t} \underbrace{e.cost_w(u,v) + e.cost_v(w,u) + e.cost_u(v,w)}_{ALG(uvw)};$$

and

$$\mathbb{E}[LP_t \mid V_t] = \quad (9)$$
$$= \frac{1}{6|V_t|} \sum_{u,v,w\in V_t} \underbrace{e.lp_w(u,v) + e.lp_v(w,u) + e.lp_u(v,w)}_{LP(uvw)}.$$

We denote each term in the first sum by $ALG(uvw)$ and each term in the second sum by $LP(uvw)$. Observe, that if $ALG(uvw) \leq \alpha LP(uvw)$ for all $u,v,w \in V$, then $\mathbb{E}[ALG_t] \leq \alpha\mathbb{E}[LP_t]$, and, hence, $\mathbb{E}[ALG] \leq \alpha LP$. We thus obtain the following lemma.

LEMMA 4. *Fix a set of functions $\{f^+, f^-, f^\circ\}$ with $f^+(0) = f^-(0) = f^\circ(0) = 0$. If $ALG(uvw) \leq \alpha LP(uvw)$ for every $u,v,w \in V$ (see (6), (7), (8), and (9) for definitions), then the expected number of violated constraints at step t is bounded by α times the expected LP volume removed at step t:*

$$\mathbb{E}[ALG_t] \leq \alpha\mathbb{E}[LP_t].$$

Consequently, the expected cost of the clustering returned by the algorithm is upper bounded by αLP.

2.2 Triple-Based Analysis

To finish analysis we need to show that $ALG(uvw) \leq \alpha LP(uvw)$ for every triple of vertices $u,v,w \in V$. We analyze our choice of functions f for complete k-partite graphs in Appendix A; the analysis of our choice of f for complete graphs is deferred to the full version. We show that functions from Theorem 2 satisfy the conditions of Lemma 4 with $\alpha = 2.06$ for complete graphs; and functions from Theorem 3 satisfy the conditions of Lemma 4 with $\alpha = 3$ for complete k-partite graphs. To show that $ALG(uvw) \leq \alpha LP(uvw)$ for every triangle uvw, we consider all triangles uvw with LP values satisfying triangle inequalities with all possible types of edges: positive, negative, and "missing" or "neutral" edges. For brevity, we refer to triangles as (s_{uv}, s_{vw}, s_{uw}) where each s is one of the symbols "+", "−" or "∅". For example, a $(+,-,\varnothing)$-triangle is a triangle having two edges: a positive edge and a negative edge; the third edge is missing.

The analysis of the functions f requires considering many cases. We show that $ALG(uvw) \leq \alpha LP(uvw)$ for all (s_1,s_2,s_3)-triangles with edge lengths (x,y,z) satisfying triangle inequalities that may possibly appear in each type of graph (we use that there are no neutral edges in a complete graph, and that no triangle in a complete k-partite graph contains exactly two neutral edges). In other words, we fix types of edges for every triangle and then consider a function of edge lengths x_{uv},x_{vw}, x_{uw}, and cut probabilities p_{uv}, p_{vw}, p_{uw} formally defined using algebraic expressions (6), (7), (8), and (9):

$$\mathcal{C}(x_{uv}, x_{vw}, x_{uw}, p_{uv}, p_{vw}, p_{uw}) = \alpha LP(uvw) - ALG(uvw).$$

Note, that this function is defined even for those triangles that are not present in our graph. We show that

$$\mathcal{C}(x_{uv}, x_{vw}, x_{uw}, f^{uv}(x_{uv}), f^{vw}(x_{vw}), f^{uw}(x_{uw})) \geq 0,$$

for all $x_{uv}, x_{vw}, x_{uw} \in [0,1]$ satisfying the triangle inequality. Here, f^{uv}, f^{vw} and f^{uw} are rounding functions for the edges (u,v), (v,w) and (u,w) respectively. We first prove that for many rounding functions f, and, particularly, for rounding functions we use in this paper, it is sufficient to verify the inequality $\mathcal{C} \geq 0$ only for those x_{uv}, x_{vw}, and x_{uw} for which the triangle inequality is tight, and a few corner cases.

LEMMA 5. *Suppose that f^+ is a monotonically non-decreasing piecewise convex function; f^- is a monotonically non-decreasing piecewise concave function; and f° is a monotonically non-decreasing function. Let A^+ be the set of endpoints of convex pieces of f^+, and A^- be the set of endpoints of the concave pieces of f^-. (Note that $\{0,1\} \subset A^+$ and $\{0,1\} \subset A^-$.) If the conditions of Lemma 4 are violated for some (s_{uv}, s_{vw}, s_{uw})-triangle with edge lengths $(x^*_{uv}, x^*_{vw}, x^*_{uw})$ i.e.,*

$\alpha LP(uvw) - ALG(uvw) < 0$, *then there exists possibly another triangle* (x_{uv}, x_{vw}, x_{uw}) *(satisfying triangle inequalities) for which* $\alpha LP(uvw) - ALG(uvw) < 0$ *such that either*

1. *the triangle inequality is tight for* (x_{uv}, x_{vw}, x_{uw})*; or*
2. *the lengths of all positive edges of the triangle belong to* A^+*; the lengths of all negative edges of the triangle belong to* A^-*; the lengths of all neutral edges belong to* $\{0, 1\}$.

PROOF. We show that the function

$$\mathcal{C}(x_{uv}, x_{vw}, x_{uw}, f^{uv}(x_{uv}), f^{vw}(x_{vw}), f^{uw}(x_{uw}))$$

has the global minimum over the region satisfying triangle inequalities at a point (x_{uv}, x_{vw}, x_{uw}) satisfying (1) or (2). By slightly perturbing functions f, we may assume that these functions are strictly increasing.[2] Suppose that $\mathcal{C}$ has a minimum at point x. Consider one of the edges, say, x_{uv} whose length does not lie in the corresponding set A^+, A^- or $\{0, 1\}$. Let $p_{uv} = f^{uv}(x_{uv})$; and let $g^{uv}(p_{uv}) = (f^{uv})^{-1}(p_{uv})$. Note that the function $\mathcal{C}$ is a multilinear polynomial of x's and p's; in particular, it is linear in p_{uv} and x_{uv}. Moreover, it does not have monomials containing both x_{uv} and p_{uv}. This follows from the formal definition of $\mathcal{C}$. Let us fix all variables except for x_{uv} and p_{uv}. We now express x_{uv} as a function of p_{uv}: $x_{uv} = g^{uv}(p_{uv})$. The function g^{uv} is locally concave if (u, v) is a positive edge; and it is locally convex if (u, v) is a negative edge. Here we use that x_{uv} is not in A^+ or A^-. Observe that the only term containing x_{uv} in $\mathcal{C}$ comes from the expression for $LP(uvw)$. Thus, the coefficient of x_{uv} is positive if (u, v) is a positive edge; and it is negative if (u, v) is a negative edge. The function $\mathcal{C}$ does not depend on x_{uv} if (u, v) is a neutral edge (of course, $\mathcal{C}$ may depend on p_{uv}). So the function $\mathcal{C}(g^{uv}(p_{uv}), x_{vw}, x_{uw}, p_{uw}, f^{vw}(x_{vw}), f^{uw}(x_{uw}))$ is a concave function of p_{uw} (when x_{vw} and x_{uw} are fixed). Therefore, if we slightly decrease or increase p_{uw} the value of $\mathcal{C}$ will decrease. But we assumed that $\mathcal{C}$ has the global minimum at x. Hence, x lies on the boundary of the region constrained by the triangle inequality. This concludes the proof. □

We leave the analysis for the complete case in the full version, and include the considerably simpler analysis for the k-partite case in Appendix A.

3. CHOOSING ROUNDING FUNCTIONS

We now discuss how we came up with the rounding functions f for complete graphs. We need to find functions f^+ and f^- that satisfy the conditions of Lemma 4, i.e., such that for all triangles uvw, $\alpha LP(uvw) - ALG(uvw) \geq 0$. (For brevity, we shall drop the argument uvw of the ALG and LP functions from now on.) Lemma 5 suggests that we only need to care about triangles for which the triangle inequality is tight. We focus on some special families of such triangles and obtain lower and upper bounds of f^+ and f^-. Then, we find functions satisfying these constraints. Later, we formally prove that the functions we found indeed give $\alpha = 2.06$ approximation. We sketch the proof in Section 4 and give a detailed proof in the full version.

We first consider a $(+, -, -)$-triangle with edge lengths $(0, x, x)$.

LEMMA 6. *For a* $(+, -, -)$*-triangle with edge lengths* $(0, x, x)$,

$$\frac{ALG}{LP} = \frac{1 - f^-(x)^2}{1 - x}.$$

[2]Formally, we consider a sequence of strictly monotone functions uniformly converging to our rounding functions.

PROOF. We simply calculate ALG and LP using (6) and (7). □

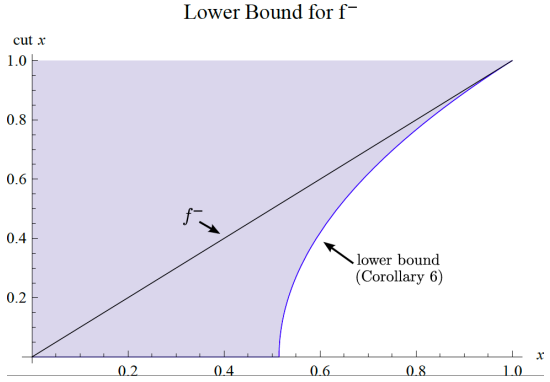
COROLLARY 7. *Any function* f^- *that achieves an* α*-approximation on all* $(+, -, -)$*-triangles satisfies*

$$f^-(x) \geq \sqrt{1 - \alpha(1 - x)}$$

for all $x \in [0, 1]$.

CLAIM 8. *The function* $f^-(x) = x$ *does not violate the conditions of Corollary 7 for* $\alpha = 2.06$.

Fixing $\alpha = 2.06$, the function we select is restricted to the blue region below.



Thus we take $f^-(x) = x$, as this choice is an easy candidate for the analysis. Now, we bound f^+ using the tight case for the linear rounding, a $(+, +, -)$-triangle with edge lengths $(x, x, 2x)$.

LEMMA 9. *Any function* f^+ *that achieves an* α*-approximation ratio on all* $(+, +, -)$*-triangles has*

$$f^+(x) \geq$$

$$\geq \frac{4x - 2\alpha x^2 - \sqrt{(2\alpha x^2 - 4x)^2 - (1 - \alpha + 4x)(1 + \alpha - 2\alpha x)}}{(1 + \alpha - 2\alpha x)},$$

for $x \in [0, 1/2]$*, if* $f^-(x) = x$.

PROOF. Again, we calculate $\alpha LP - ALG$ using (6) and (7). This yields a quadratic function in $f^+(x)$:

$$\alpha LP - ALG = -1 + \alpha - 4x - 4x(-2 + \alpha x) f^+(x) - (1 + \alpha - 2\alpha x) f^+(x)^2.$$

Solving for $\alpha LP - ALG \geq 0$ in terms of $f^+(x)$, we get our result. □

This lower bound on f^+ is necessary for approximating $(+, +, -)$-triangles well, but choosing a large f^+ has consequences for the approximation ratio of $(+, +, +)$-triangles. We use a $(+, +, +)$-triangle with edge lengths $(x, x, 0)$ to obtain an upper bound on f^+.

LEMMA 10. *Any function* f^+ *that achieves an* α*-approximation ratio on all* $(+, +, +)$*-triangles satisfies for all* $x \in [0, 1]$,

$$f^+(x) \leq 1 - \sqrt{1 - \alpha x}.$$

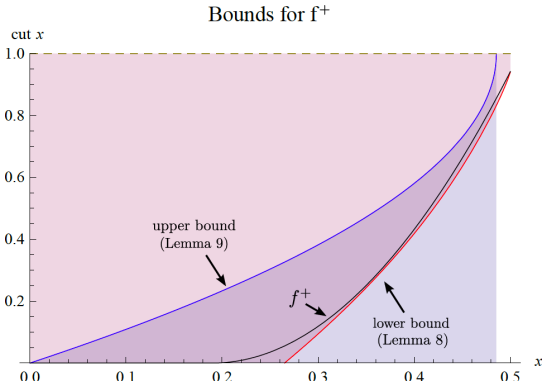
PROOF. We compute $\alpha LP - ALG$ using (6) and (7) as before. This yields a different quadratic function in $f^+(x)$:

$$\alpha LP - ALG = 2(\alpha x - 2f^+(x) + f^+(x)^2).$$

Solving for $\alpha LP - ALG \geq 0$ in terms of $f^+(x)$, we get our result. □

The bounds from Lemma 10 and Lemma 9 give a restricted region in which $f^+(x)$ may be for $x \in [0, \frac{1}{2}]$ and $x \in [0, \frac{1}{\alpha}]$ to get an α-approximation. We chose f^+ so that it would violate neither constraint, and also be easy to analyze.

CLAIM 11. *Functions f^+ and f^- from Theorem 2 do not violate the conditions in Lemma 10 or Lemma 9 for an $\alpha = 2.06$ approximation when $a = 0.19, b = 0.5095$.*



The parameters a, b were chosen computationally within these analytic bounds so as to minimize α.

4. PICTORIAL PROOFS

Here we give a *pictorial proof* of our main result, Theorem 2. This proof serves as an illustration for an analytical proof we present in the full version.

To prove Theorem 2, we use the framework presented in Section 2.2, bounding the approximation ratio of each triangle for every set of LP weights permitted by the constraints. Lemma 5 allows us to consider only triangles for which the triangle inequality is tight, that is, triangles of the form $(x, y, x + y)$. Figure 1, Figure 2, Figure 3, and Figure 4 on the following page are plots of the polynomials $2.06 \cdot LP(uvw) - ALG(uvw)$ when the triangle inequality is tight; the fact that each of these polynomials is positive in the range of possible LP weights proves Theorem 2.

The analytical proof in the full version proceeds by showing that this difference polynomial is positive for all possible LP weights. In the first two cases, we take partial derivatives to find the worst triangle lengths in terms of a single variable, then bound the roots of the polynomials; in the latter two cases we are able to provide a factorization for the polynomial that is positive term-by-term. For the complete argument, see the full version.

Acknowledgements

Part of this work was done when the first, third, and fourth authors were visiting Microsoft Research. Shuchi Chawla is supported in part by NSF Award CCF-1320854. Tselil Schramm is supported by an NSF Graduate Research Fellowship, Grant No. DGE 1106400. Grigory Yaroslavtsev is supported by an ICERM Institute Postdoctoral Fellowship at Brown and a Warren Center Postdoctoral Fellowship at Penn.

5. REFERENCES

[1] N. Ailon. Aggregation of partial rankings, p-ratings and top-m lists. *Algorithmica*, 57(2):284–300, 2010.

[2] N. Ailon, N. Avigdor-Elgrabli, E. Liberty, and A. van Zuylen. Improved approximation algorithms for bipartite correlation clustering. *SIAM J. Comput.*, 41(5):1110–1121, 2012.

[3] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5), 2008.

[4] N. Ailon and E. Liberty. Correlation clustering revisited: The "true" cost of error minimization problems. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part I*, ICALP '09, pages 24–36, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] N. Amit. *The bicluster graph editing problem*. PhD thesis, Tel Aviv University, 2004.

[6] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

[7] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 2014/11/02 1999.

[8] M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005.

[9] S. Chawla, R. Krauthgamer, R. Kumar, Y. Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. *computational complexity*, 15(2):94–114, 2006.

[10] Y. Chen, S. Sanghavi, and H. Xu. Clustering sparse graphs. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2204–2212. Curran Associates, Inc., 2012.

[11] F. Chierichetti, N. N. Dalvi, and R. Kumar. Correlation clustering in mapreduce. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 641–650, 2014.

[12] W. Cohen and J. Richman. Learning to match and cluster entity names. In *In ACM SIGIR-2001 Workshop on Mathematical/Formal Methods in Information Retrieval*, 2001.

[13] W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 475–480, New York, NY, USA, 2002. ACM.

[14] E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci.*, 361(2-3):172–187, 2006.

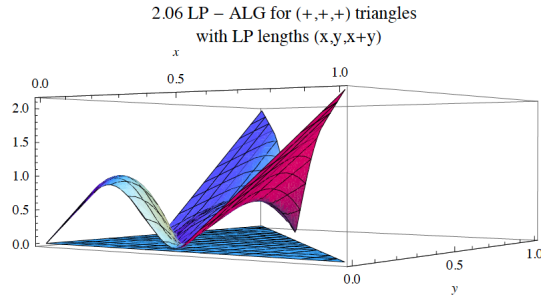
[15] N. Downing, P. J. Stuckey, and A. Wirth. Improved consensus clustering via linear programming. In *Computer Science 2010, Thirty-Third Australasian Computer Science Conference (ACSC 2010), Brisbane, Australia, January 18-22, 2010, Proceedings*, pages 61–70, 2010.

[16] V. Filkov. Integrating microarray data by consensus clustering. In *In Proceedings of International Conference on Tools with Artificial Intelligence (ICTAI*, pages 418–425, 2003.

[17] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *TKDD*, 1(1), 2007.

[18] I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '06, pages 1167–1176, Philadelphia, PA, USA, 2006. Society for Industrial and Applied Mathematics.

[19] M. Karpinski and W. Schudy. Linear time approximation schemes for the gale-berlekamp game and related



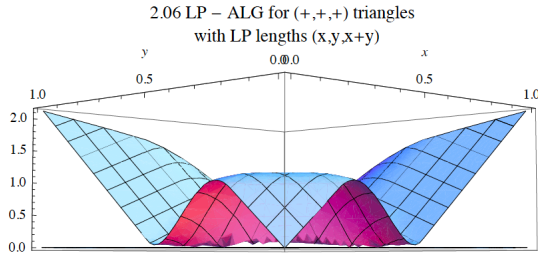
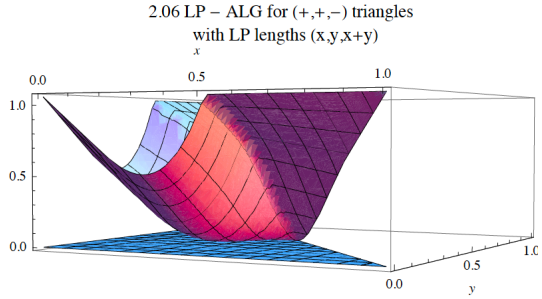


Figure 1: The difference between $\alpha \cdot LP(uvw) - ALG(uvw)$ for $(+,+,+)$-triangles with tight triangle inequality constraints.



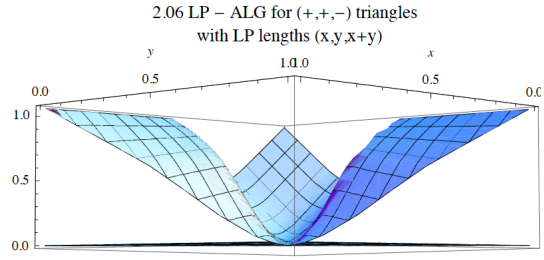


Figure 2: The difference between $\alpha \cdot LP(uvw) - ALG(uvw)$ for $(+,+,-)$-triangles with tight triangle inequality constraints.

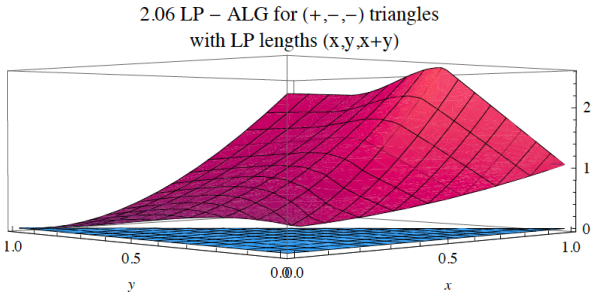


Figure 3: The difference between $\alpha \cdot LP(uvw) - ALG(uvw)$ for $(+,-,-)$-triangles with tight triangle inequality constraints.

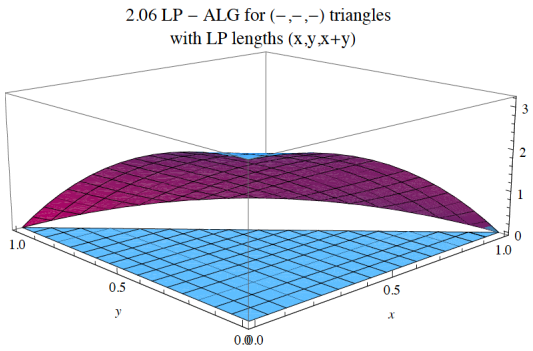


Figure 4: The difference between $\alpha \cdot LP(uvw) - ALG(uvw)$ for $(-,-,-)$-triangles with tight triangle inequality constraints.

minimization problems. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 313–322, New York, NY, USA, 2009. ACM.

[20] K. Makarychev, Y. Makarychev, and A. Vijayaraghavan. Algorithms for semi-random correlation clustering. *arXiv preprint arXiv:1406.5667*, 2014.

[21] C. Mathieu, O. Sankur, and W. Schudy. Online Correlation Clustering. In J.-Y. Marion and T. Schwentick, editors, *27th International Symposium on Theoretical Aspects of Computer Science*, volume 5 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 573–584, Dagstuhl, Germany, 2010. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[22] C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 712–728, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[23] A. Mccallum and B. Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *In NIPS*, pages 905–912. MIT Press, 2003.

[24] C. Swamy. Correlation clustering: Maximizing agreements via semidefinite programming. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 526–527, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[25] J. Tan. A note on the inapproximability of correlation clustering. *Inf. Process. Lett.*, 108(5):331–335, Nov. 2008.

[26] J. Van Gael and X. Zhu. Correlation clustering for crosslingual link detection. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 1744–1749, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[27] A. van Zuylen, R. Hegde, K. Jain, and D. P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 405–414, 2007.

[28] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

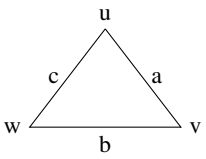
[29] A. Wirth. Correlation clustering. In C. Sammut and G. Webb, editors, *Encyclopedia of Machine Learning*, pages 227–231. Springer US, 2010.

APPENDIX

A. ANALYSIS OF THE K-PARTITE CASE

PROOF PROOF OF THEOREM 3. By Lemma 4, it suffices to show that $ALG(uvw) \leq LP(uvw)$ for every triangle uvw in the graph. There are 7 possible types of triangles in a complete k-partite graph: $(+,+,+)$-triangles, $(+,+,-)$-triangles, $(+,-,-)$-triangles, and $(-,-,-)$-triangles may exist when each vertex is a member of different partitions; $(+,+,\varnothing)$-triangles, $(+,-,\varnothing)$-triangles, and $(-,-,\varnothing)$-triangles can occur when two vertices are members of the same partition and the third vertex is a member of a different partition. When all three vertices belong to one partition, all edges are neutral, and so none contribute to the cost.

We adopt the convention that the triangles have LP lengths (a,b,c), with vertex u opposite edge b, vertex v opposite edge c, and vertex w opposite edge a (see figure below). Costs incurred when vertex $x \in \{u,v,w\}$ is a pivot will be enclosed in square brackets with subscript x, as in $[cost]_x$. Instead of writing, $ALG(uvw)$ and $LP(uvw)$, we write ALG and LP.



$(+,-,\varnothing)$-Triangles: We first analyze the most interesting case: what happens if our triangle has edge labels $(+,-,\varnothing)$. Consider a $(+,-,\varnothing)$-triangle with side lengths (a,b,c) respectively. We have

$$\begin{aligned} ALG &= [(1-p_{uv})(1-p_{uw})]_u + [(1-p_{uw})p_{vw} + (1-p_{vw})p_{uw}]_w \\ &= 1 - f_3^+(a) + f_3^-(b) + f_3^+(a)f_3^\circ(c) - 2f_3^-(b)f_3^\circ(c); \end{aligned}$$

$$\begin{aligned} LP &= [(1-b)\cdot(1-p_{uw}p_{uv})]_u + [a\cdot(1-p_{uw}p_{vw})]_w \\ &= (1-b)(1-f_3^+(a)f_3^\circ(c)) + a(1-f_3^-(b)f_3^\circ(c)). \end{aligned}$$

Since f_3^+ and f_3° are piecewise functions, we consider four cases.

1. If $a < 1/3$, $c < 2/3$, then $LP = 1 - b + a - \frac{3}{2}abc$, and $ALG = 1 + b - 3bc$. So,

$$\begin{aligned} 3\cdot LP - ALG &= 2 + 3a + b(3c - 4 - \frac{9}{2}ac) \\ &\geq 2 + 3a + (a+c)(3c - 4 - \frac{9}{2}ac) \\ &= (2-a) + ac(3 - \frac{9}{2}a - \frac{9}{2}c) + (3c^2 - 4c). \end{aligned}$$

The inequality above follows from the triangle inequality constraints $b \leq a + c$. We now bound each term separately using the assumptions $a < 1/3$ and $c < 2/3$: We have $(2-a) > 5/3$, $(3 - 9a/2 - 9c/2) \geq -3/2$, and $(3c^2 - 4c) \geq -4/3$. Thus,

$$3\cdot LP - ALG \geq \tfrac{1}{3} - \tfrac{3}{2}ac \geq 0.$$

as desired.

2. If $a < \frac{1}{3}$, $c \geq \frac{2}{3}$, then $LP = 1 - b + a - ab$, and $ALG = 1 - b$. Because $a - ab > 0$, $ALG \leq LP$, as desired.

3. If $a \geq \frac{1}{3}$, $c < \frac{2}{3}$, then $LP = 1 - b - \frac{3}{2}c + \frac{3}{2}bc + a - \frac{3}{2}abc$ and $ALG = b + \frac{3}{2}c - 3bc$. We have

$$\begin{aligned} 3\cdot LP - ALG &= 3 + 3a - 4(b+c) + \tfrac{15}{2}bc - \tfrac{9}{2}abc \\ &\geq 4 - 4(b+c) + \tfrac{15}{2}bc - \tfrac{3}{2}bc \\ &= 4(1-b)(1-c), \end{aligned}$$

where in the second inequality we applied $a \geq 1/3$. Thus, $ALG \leq 3\cdot LP$.

4. If $a \geq \frac{1}{3}$, $c \geq \frac{2}{3}$, then $LP = 0$ and $ALG = 0$ as well, so $ALG \leq 3\cdot LP$.

We now bound the expected costs of the 6 other triangles.

$(+,+,+)$-Triangles: If $a,b,c < \frac{1}{3}$ or $a,b,c \geq \frac{1}{3}$, then either no edges are cut and the algorithm makes no mistakes, or all edges are cut and no edge is ever charged unsafely, so $ALG = 0$. The remaining cases are $a,b < \frac{1}{3} \leq c$ and $a < \frac{1}{3} \leq b,c$. In both cases,

$$\begin{aligned} ALG = {} & [f_3^+(c)(1-f_3^+(a)) + f_3^+(a)(1-f_3^+(c))]_u \\ & + [f_3^+(a)(1-f_3^+(b)) + f_3^+(b)(1-f_3^+(a))]_v \\ & + [f_3^+(b)(1-f_3^+(c)) + f_3^+(c)(1-f_3^+(b))]_w, \end{aligned}$$

$$LP = [b(1 - f_3^+(c)f_3^+(a))]_u + [c(1 - f_3^+(a)f_3^+(b))]_v + [a(1 - f_3^+(b)f_3^+(c))]_w.$$

When $a, b < \frac{1}{3} \leq c$, $ALG = 2$, $LP = a + b + c \geq 2c \geq \frac{2}{3}$, where we apply the LP triangle inequality constraints. When $a < \frac{1}{3} \leq b, c$, $ALG = 2$, $LP = c + b \geq \frac{2}{3}$. Therefore $3 \cdot LP \geq ALG$ as desired.

$(-,-,-)$**-Triangles:** In this case, the function f_3^- is the only one participating in the costs; the costs are

$$\begin{aligned} ALG &= [(1 - f_3^-(a))(1 - f_3^-(c))]_u + [(1 - f_3^-(a))(1 - f_3^-(b))]_v \\ &\quad + [(1 - f_3^-(c))(1 - f_3^-(b))]_w \\ &= 3 - 2a - 2b - 2c + ab + bc + ac, \end{aligned}$$

and

$$\begin{aligned} LP &= [(1 - b)(1 - f_3^-(a)f_3^-(c))]_u + [(1 - c)(1 - f_3^-(a)f_3^-(b))]_v \\ &\quad + [(1 - a)(1 - f_3^-(c))f_3^-(b))]_w \\ &= 3 - a - b - c - ab - ac - bc + 3abc. \end{aligned}$$

We verify that the 3-approximation holds:

$$\begin{aligned} 3 \cdot LP - ALG &= 6 - a - b - c - 4ab - 4ac - 4bc + 9abc \\ &= 6 - a - b - 4ab - c(4a(1 - b) \\ &\quad + 4b(1 - a) + (1 - ab)) \\ &\geq 5 - 5a - 5b + 5ab = 5(1 - a)(1 - b), \end{aligned}$$

where in the third inequality we have applied $c \leq 1$. Thus, $3 \cdot LP \geq ALG$ as desired.

$(+,+,-)$**-Triangles:** Here, we must verify the cases $a, b < \frac{1}{3}$, $a < \frac{1}{3} \leq b$, and $\frac{1}{3} \leq a, b$. In all cases,

$$\begin{aligned} ALG &= [f_3^+(a)(1 - f_3^-(c)) + f_3^-(c)(1 - f_3^+(a))]_u \\ &\quad + [(1 - f_3^+(a))(1 - f_3^+(b))]_v \\ &\quad + [f_3^+(b)(1 - f_3^-(c)) + f_3^-(c)(1 - f_3^+(b))]_w, \\ LP &= [b(1 - f_3^+(a)f_3^-(c))]_u + [(1 - c)(1 - f_3^+(a)f_3^+(b))]_v \\ &\quad + [a(1 - f_3^+(b)f_3^-(c))]_w. \end{aligned}$$

When $a, b < \frac{1}{3}$, we have $ALG = 1 + 2c$, $LP = 1 - c + b + a \geq 1$, where we have applied the triangle inequality constraint to bound the LP. Since $c \leq 1$, we have $3 \cdot LP \geq ALG$.

When $a < \frac{1}{3} \leq b$, we have $ALG = 1$, $LP = 1 + a + b - c - ac \geq 1 - ac \geq \frac{2}{3}$, where we have applied the triangle inequality constraint and the fact that $ac \leq \frac{1}{3}$.

When $\frac{1}{3} \leq a, b$, we have $ALG = 2(1 - c)$, $LP = b(1 - c) + a(1-c) \geq \frac{2}{3}(1-c)$, where we have applied the fact that $a+b \geq \frac{2}{3}$. Thus, we have $3 \cdot LP \geq ALG$ in this case as well.

$(+,-,-)$**-Triangles:** Here we must verify two cases: $a < \frac{1}{3}$ and $a \geq \frac{1}{3}$. In both cases, the costs are

$$\begin{aligned} ALG &= [(1 - f_3^+(a))(1 - f_3^-(c))]_u + [(1 - f_3^+(a))(1 - f_3^-(b))]_v \\ &\quad + [f_3^-(c)(1 - f_3^-(b)) + f_3^-(b)(1 - f_3^-(c))]_w, \\ LP &= [(1 - b)(1 - f_3^+(a)f_3^-(c))]_u + [(1 - c)(1 - f_3^+(a)f_3^-(b))]_v \\ &\quad + [a(1 - f_3^-(b)f_3^-(c))]_w. \end{aligned}$$

When $a < \frac{1}{3}$, $ALG = 2 - 2bc$, $LP = 2 - b - c + a - abc$. We calculate,

$$\begin{aligned} 3 \cdot LP - ALG &= 4 - 3b - 3c + 3a + 2bc - 3abc \\ &= 4(1 - b)(1 - c) + b + c - 2bc + 3a(1 - bc) \\ &= (1 - bc) + 3(1 - b)(1 - c) + 3a(1 - bc) \geq 0, \end{aligned}$$

as desired.

When $a \geq \frac{1}{3}$, $ALG = c + b - 2bc$, $LP = 2 - 2b - 2c + 2bc + a(1 - bc)$. Again we verify,

$$\begin{aligned} 3 \cdot LP - ALG &= 6 - 7b - 7c + 8bc + 3a(1 - bc) \\ &\geq 7 - 7b - 7c + 7bc \\ &= 7(1 - b)(1 - c) \geq 0, \end{aligned}$$

where to get the inequality we applied the assumption $a \geq \frac{1}{3}$.

$(+,+,\varnothing)$**-Triangles:** Here, when $c \geq \frac{2}{3}$, the rounding is deterministic: if $a < \frac{1}{3} \leq b$, $ALG = 1$, $LP = b \geq \frac{1}{3}$; if $\frac{1}{3} \leq a, b$, the cost to the algorithm is zero. Crucially, it is not possible to have both $a, b < \frac{1}{3}$, because $a + b \geq c \geq \frac{2}{3}$.

Now we deal with the cases where $c < \frac{2}{3}$. The costs are

$$\begin{aligned} ALG &= [f_3^\circ(c)(1 - f_3^+(a)) + f_3^+(a)(1 - f_3^\circ(c))]_u \\ &\quad + [f_3^\circ(c)(1 - f_3^+(b)) + f_3^+(b)(1 - f_3^\circ(c))]_w \\ LP &= [b(1 - f_3^\circ(c)f_3^+(a))]_u + [a(1 - f_3^\circ(c)f_3^+(b))]_w \end{aligned}$$

When $a, b < \frac{1}{3}$, $ALG = 3c$, and $LP = a + b \geq c$, where we applied the triangle inequality.

When $a < \frac{1}{3} \leq b$, we have $ALG = 1$, $LP = b + a(1 - c) \geq \frac{1}{3}$, by our assumption on b.

For $\frac{1}{3} \leq a, b$, $ALG = 2(1 - c)$ and $LP = (b + a)(1 - c) \geq \frac{2}{3}(1 - c)$. Therefore, $3 \cdot LP \geq ALG$ holds in all cases.

$(-,-,\varnothing)$**-Triangles:** Here, when $c \geq \frac{2}{3}$, the algorithm never incurs cost on unsafe edges. Therefore, we must simply calculate the costs when $c < \frac{2}{3}$:

$$\begin{aligned} ALG &= [(1 - f_3^\circ(c))(1 - f_3^-(a))]_u + [(1 - f_3^\circ(c))(1 - f_3^-(b))]_w \\ &= (1 - \tfrac{3}{2}c)(2 - a - b), \\ LP &= [(1 - b)(1 - f_3^\circ(c)f_3^-(a))]_u + [(1 - a)(1 - f_3^\circ(c)f_3^-(b))]_w \\ &= (1 - b)(1 - \tfrac{3}{2}ac) + (1 - a)(1 - \tfrac{3}{2}bc) \end{aligned}$$

We verify that $3 \cdot LP \geq ALG$:

$$LP - ALG = 3c(1 - b)(1 - a) \geq 0.$$

Thus the approximation holds.

This concludes the case analysis for k-partite complete graphs. □