Correlation Clustering with Asymmetric Classification Errors

Jafar Jafarov [1] **Sanchit Kalhan** [2] **Konstantin Makarychev** [2] **Yury Makarychev** [3]

Abstract

In the Correlation Clustering problem, we are given a weighted graph G with its edges labelled as "similar" or "dissimilar" by a binary classifier. The goal is to produce a clustering that minimizes the weight of "disagreements": the sum of the weights of "similar" edges across clusters and "dissimilar" edges within clusters. We study the correlation clustering problem under the following assumption: Every "similar" edge e has weight $\mathbf{w}_e \in [\alpha\mathbf{w}, \mathbf{w}]$ and every "dissimilar" edge e has weight $\mathbf{w}_e \geq \alpha\mathbf{w}$ (where $\alpha \leq 1$ and $\mathbf{w} > 0$ is a scaling parameter). We give a $(3 + 2\log_e(1/\alpha))$ approximation algorithm for this problem. This assumption captures well the scenario when classification errors are asymmetric. Additionally, we show an asymptotically matching Linear Programming integrality gap of $\Omega(\log 1/\alpha)$.

1. Introduction

In the Correlation Clustering problem, we are given a set of objects with pairwise similarity information. Our aim is to partition these objects into clusters that match this information as closely as possible. The pairwise information is represented as a weighted graph G whose edges are labelled as "positive/similar" and "negative/dissimilar" by a noisy binary classifier. The goal is to find a clustering $\mathcal{C}$ that minimizes the weight of edges disagreeing with this clustering: A positive edge is in disagreement with $\mathcal{C}$, if its endpoints belong to distinct clusters; and a negative edge is in disagreement with $\mathcal{C}$ if its endpoints belong to the same cluster. We call this objective the MinDisagree objective. The MinDisagree objective has been extensively studied in literature since it was introduced by Bansal, Blum, and Chawla (2004) (see e.g., (Charikar et al., 2003; Demaine et al., 2006; Ailon et al., 2008; Pan et al., 2015; Chawla et al., 2015)). There are currently two standard models for Correlation Clustering which we will refer to as (1) Correlation Clustering on Complete Graphs and (2) Correlation Clustering with Noisy Partial Information. In the former model, we assume that graph G is complete and all edge weights are the same i.e., G is unweighted. In the latter model, we do not make any assumptions on the graph G. Thus, edges can have arbitrary weights and some edges may be missing. These models are quite different from the computational perspective. For the first model, Ailon, Charikar, and Newman (2008) gave a 2.5 approximation algorithm. This approximation factor was later improved to 2.06 by Chawla, Makarychev, Schramm, and Yaroslavtsev (2015). For the second model, Charikar, Guruswami, and Wirth (2003) and Demaine, Emanuel, Fiat, and Immorlica (2006) gave an $O(\log n)$ approximation algorithm, they also showed that Correlation Clustering with Partial Noisy Information is as hard as the Multicut problem and, hence, $O(\log n)$ is likely to be the best possible approximation for this problem. In this paper, we show how to interpolate between these two models for Correlation Clustering.

We study the Correlation Clustering problem on complete graphs with edge weights. In our model, the weights on the edges are constrained such that the ratio of the lightest edge in the graph to the heaviest positive edge is at least $\alpha \leq 1$. Thus, if $\mathbf{w}$ is the weight of the heaviest positive edge in the graph, then each positive edge has weight in $[\alpha\mathbf{w}, \mathbf{w}]$ and each negative edge has weight greater than or equal to $\alpha\mathbf{w}$. We argue that this model – which we call Correlation Clustering with Asymmetric Classification Errors – is more adept at capturing the subtleties in real world instances than the two standard models. Indeed, the assumptions made by the Correlation Clustering on Complete Graphs model are too strong, since rarely do real world instances have equal edge weights. In contrast, in the Correlation Clustering with Noisy Partial Information model we can have edge weights that are arbitrarily small or large, an assumption which is too weak. In many real world instances, the edge weights lie in some range $[a, b]$ with $a, b > 0$. Our model captures a larger family of instances.

[1]University of Chicago, Chicago, Illinois, USA [2]Northwestern University, Evanston, Illinois, USA [3]Toyota Technological Institute at Chicago (TTIC), Chicago, Illinois, USA. Correspondence to: Jafar Jafarov <jafarov@uchicago.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

Furthermore, the nature of classification errors for objects that are similar and objects that are dissimilar is quite different. In many cases, a *positive* edge uv indicates that the classifier found some actual evidence that u and v are similar; while a negative edge simply means that the classifier could not find any such proof that u and v are similar, it does not mean that the objects u and v are necessarily dissimilar. In some other cases, a *negative* edge uv indicates that the classifier found some evidence that u and v are dissimilar; while a positive edge simply means that the classifier could not find any such proof. We discuss several examples below. Note that in the former case, a positive edge gives a substantially stronger signal than a negative edge and should have a higher weight; in the latter, it is the other way around: a negative edge gives a stronger signal than a positive edge and should have a higher weight. We make this statement more precise in Section 1.1.

The following examples show how the Correlation Clustering with Asymmetric Classification Errors model can help in capturing real world instances. Consider an example from the paper on Correlation Clustering by Pan, Papailiopoulos, Oymak, Recht, Ramchandran, and Jordan (2015). In their experiments, Pan et al. (2015) used several data sets including *dblp-2011* and *ENWiki-2013*[1]. In the graph *dblp-2011*, *each vertex represents a scientist and two vertices are connected with an edge if the corresponding authors have co-authored an article*. Thus, a positive edge with weight $\mathbf{w}^+$ between Alice and Bob in the Correlation Clustering instance indicates that Alice and Bob are coauthors, which strongly suggests that Alice and Bob work in similar areas of Computer Science. However, it is not true that all researchers working in some area of computer science have co-authored papers with each other. Thus, the negative edge that connects two scientists who do not have an article together does not deserve to have the same weight as a positive edge, and thus can be modeled as a negative edge with weight $\mathbf{w}^- < \mathbf{w}^+$.

Similarly, the vertices of the graph *ENWiki-2013* are Wikipedia pages. Two pages are connected with an edge if there is a link from one page to another. A link from one page to the other is a strong suggestion that the two pages are related and hence can be connected with a positive edge of weight $\mathbf{w}^+$, while it is not true that two similar Wikipedia pages necessarily should have a link from one to the other. Thus, it would be better to join such pages with a negative edge of weight $\mathbf{w}^- < \mathbf{w}^+$.

Consider now the multi-person tracking problem. The problem is modelled as a Correlation Clustering or closely related Lifted Multicut Problem (Tang et al., 2016; 2017) on a graph, whose vertices are people detections in video sequences. Two detections are connected with a positive or negative edge depending on whether the detected people have similar or dissimilar appearance (as well as some other information). In this case, a negative edge (u, v) is more informative since it signals that the classifier has identified body parts that do not match in detections u and v and thus the detected people are likely to be different (a positive edge (u, v) simply indicates that the classifier was not able to find non-matching body parts).

The Correlation Clustering with Asymmetric Classification Errors model captures the examples we discussed above. It is instructive to consider an important special case where all positive edges have weight $\mathbf{w}^+$ and all negative edges have weight $\mathbf{w}^-$ with $\mathbf{w}^+ \neq \mathbf{w}^-$. If we were to use the state of the art algorithm for Correlation Clustering on Complete Graphs on our instance for Correlation Clustering with Asymmetric Classification Errors (by completely ignoring edge weights and looking at the instance as an unweighted complete graph), we would get a $\Theta(\max(\mathbf{w}^+/\mathbf{w}^-, \mathbf{w}^-/\mathbf{w}^+))$ approximation to the MinDisagree objective. While if we were to use the state of the art algorithms for Correlation Clustering with Noisy Partial Information on our instance, we would get a $O(\log n)$ approximation to the MinDisagree objective.

Our Contributions. In this paper, we present an approximation algorithm for Correlation Clustering with Asymmetric Classification Errors. Our algorithm gives an approximation factor of $A = 3 + 2\log_e 1/\alpha$. Consider the scenario discussed above where all positive edges have weight $\mathbf{w}^+$ and all negative edges have weight $\mathbf{w}^-$. If $\mathbf{w}^+ \geq \mathbf{w}^-$, our algorithm gets a $(3 + 2\log_e \mathbf{w}^+/\mathbf{w}^-)$ approximation; if $\mathbf{w}^+ \leq \mathbf{w}^-$, our algorithm gets a 3-approximation.

Definition 1. *Correlation Clustering with Asymmetric Classification Errors is a variant of Correlation Clustering on a Complete Graph. We assume that the weight $\mathbf{w}_e$ of each positive edge lies in $[\alpha\mathbf{w}, \mathbf{w}]$ and the weight $\mathbf{w}_e$ of each negative edge lies in $[\alpha\mathbf{w}, \infty)$, where $\alpha \in (0, 1]$ and $\mathbf{w} > 0$.*

We note here that the assumption that the weight of positive edges is bounded from above is crucuial. Without this assumption (even if we require that negative weights are bounded from above and below), the LP gap is unbounded for every fixed α (this follows from the integrality gap example we present in Theorem 1.3).

The following is our main theorem.

Theorem 1.1. *There exists a polynomial time $A = 3 + 2\log_e 1/\alpha$ approximation algorithm for Correlation Clustering with Asymmetric Classification Errors.*

We also study a natural extension of our model to the case of complete bipartite graphs. That is, the positive edges across the biparition have a weight between $[\alpha\mathbf{w}, \mathbf{w}]$ and the negative edges across the bipartition have a weight of at least

[1]These data sets are published by (Boldi & Vigna, 2004; Boldi et al., 2011; 2004; 2014)

$\alpha\mathbf{w}$. We provide the details of this result in the full version. Note that the state-of-the-art approximation algorithm for Correlation Clustering on Unweighted Complete Bipartite Graphs has an approximation factor of 3 (see Chawla et al. (2015)).

Theorem 1.2. *There exists a polynomial time $A = 5 + 2\log_e 1/\alpha$ approximation algorithm for Correlation Clustering with Asymmetric Classification Errors on complete bipartite graphs.*

Our next result shows that this approximation ratio is likely best possible for LP-based algorithms. We show this by exhibiting an instance of Correlation Clustering with Asymmetric Classification Errors such that integrality gap for the natural LP for Correlation Clustering on this instance is $\Omega(\log 1/\alpha)$.

Theorem 1.3. *The natural Linear Programming relaxation for Correlation Clustering has an integrality gap of $\Omega(\log 1/\alpha)$ for instances of Correlation Clustering with Asymmetric Classification Errors.*

Moreover, we can show that if there is an $o(\log(1/\alpha))$-approximation algorithm whose running time is polynomial in both n and $1/\alpha$, then there is a $o(\log n)-$approximation algorithm for the general weighted case (and also for the MultiCut problem). However, we do not know if there is an $o(\log(1/\alpha))-$approximation algorithm for the problem whose running time is polynomial in n and exponential in $1/\alpha$. The existence of such an algorithm does not imply that there is an $o(\log n)-$approximation algorithm for the general weighted case (as far as we know).

We show a similar integraplity gap result for the Correlation Clustering with Asymmetric Classification Errors on complete bipartite graphs problem. Please find the details in the full version of the paper.

Theorem 1.4. *The natural Linear Programming relaxation for Correlation Clustering has an integrality gap of $\Omega(\log 1/\alpha)$ for instances of Correlation Clustering with Asymmetric Classification Errors on complete bipartite graphs.*

Throughout the paper, we denote the set of positive edges by E^+ and the set of negative edges by E^-. We denote an instance of the Correlation Clustering problem by $G = (V, E^+, E^-)$. We denote the weight of edge e by $\mathbf{w}_e$.

1.1. Ground Truth Model

In this section, we formalize the connection between asymmetric classification errors and asymmetric edge weights. For simplicity, we assume that each positive edge has a weight of $\mathbf{w}^+$ and each negative edge has a weight of $\mathbf{w}^-$. Consider a probabilistic model in which edge labels are assigned by a noisy classifier. Let $\mathcal{C}^* = (C_1^*, \dots C_T^*)$ be the ground truth clustering of the vertex set V. The classifier labels each edge within a cluster with a "+" edge with probability p^+ and as a "−" edge with probability $1 - p^+$; it labels each edge with endpoints in distinct clusters as a "−" edge with probability q^- and as a "+" edge with probability $1 - q^-$. Thus, $(1 - p^+)$ and $(1 - q^-)$ are the classification error probabilities. We assume that all classification errors are independent.

We note that similar models have been previously studied by (Bansal et al., 2004; Elsner & Schudy, 2009; Mathieu & Schudy, 2010; Ailon et al., 2013; Makarychev et al., 2015) and others. However, the standard assumption in such models was that the error probabilities, $(1 - p^+)$ and $(1 - q^-)$, are less than a half; that is, $p^+ > 1/2$ and $q^- > 1/2$. Here, we investigate two cases (i) when $p^+ < 1/2 < q^-$ and (ii) when $q^- < 1/2 < p^+$. We assume that $p^+ + q^- > 1$, which means that the classifier is more likely to connect similar objects with a "+" than dissimilar objects or, equivalently, that the classifier is more likely to connect dissimilar objects with a "−" than similar objects. For instance, consider a classifier that looks for evidence that the objects are similar: if it finds some evidence, it adds a positive edge; otherwise, it adds a negative edge (as described in our examples *dblp-2011* and *ENWiki-2013* in the Introduction). Say, the classifier detects a similarity between two objects in the same ground truth cluster with a probability of only 30% and incorrectly detects similarity between two objects in different ground truth clusters with a probability of 10%. Then, it will add a *negative* edge between two similar objects with probability 70%! While this scenario is not captured by the standard assumption, it is captured by case (i) (here, $p^+ = 0.3 < 1/2 < q^- = 0.9$ and $p^+ + q^- > 1$).

Consider a clustering $\mathcal{C}$ of the vertices. Denote the sets of positive edges and negative edges with both endpoints in the same cluster by $\text{In}^+(\mathcal{C})$ and $\text{In}^-(\mathcal{C})$, respectively, and the sets of positive edges and negative edges with endpoints in different clusters by $\text{Out}^+(\mathcal{C})$ and $\text{Out}^-(\mathcal{C})$, respectively. Then, the log-likelihood function of the clustering $\mathcal{C}$ is,

$$\begin{aligned}
\ell(G;\mathcal{C}) &= \log\Big(\prod_{(u,v)\in \text{In}^+(\mathcal{C})} p^+ \times \prod_{(u,v)\in \text{In}^-(\mathcal{C})}(1-p^+) \\
&\quad \times \prod_{(u,v)\in \text{Out}^+(\mathcal{C})}(1-q^-) \times \prod_{(u,v)\in \text{Out}^-(\mathcal{C})} q^-\Big) \\
&= \log\Big((p^+)^{|\text{In}^+(\mathcal{C})|}(1-p^+)^{|\text{In}^-(\mathcal{C})|} \\
&\quad \cdot (1-q^-)^{|\text{Out}^+(\mathcal{C})|}(q^-)^{|\text{Out}^-(\mathcal{C})|}\Big) \\
&= |\text{In}^+(\mathcal{C})|\log p^+ + |\text{In}^-(\mathcal{C})|\log(1-p^+) \\
&\quad + |\text{Out}^+(\mathcal{C})|\log(1-q^-) + |\text{Out}^-(\mathcal{C})|\log q^-
\end{aligned}$$

$$= \underbrace{\left(|E^+| \log p^+ + |E^-| \log q^-\right)}_{\text{constant expression}}$$
$$- \underbrace{\left(|\text{Out}^+(\mathcal{C})| \log \frac{p^+}{1-q^-} + |\text{In}^-(\mathcal{C})| \log \frac{q^-}{1-p^+}\right)}_{\text{MinDisagree objective}}.$$

Let $\mathbf{w}^+ = \log \frac{p^+}{1-q^-}$ and $\mathbf{w}^- = \log \frac{q^-}{1-p^+}$. Then, the negative term – $\left(|\text{Out}^+(\mathcal{C})| \log \frac{p^+}{1-q^-} + |\text{In}^-(\mathcal{C})| \log \frac{q^-}{1-p^+}\right)$ – equals $\mathbf{w}^+|\text{Out}^+(\mathcal{C})| + \mathbf{w}^-|\text{In}^-(\mathcal{C})|$. Note that $|\text{Out}^+(\mathcal{C})|$ is the number of positive edges disagreeing with $\mathcal{C}$ and $|\text{In}^-(\mathcal{C})|$ is the number of negative edges disagreeing with $\mathcal{C}$.

Now observe that the first term in the expression above – $\left(|E^+| \log p^+ + |E^-| \log q^-\right)$ – does not depend on $\mathcal{C}$. It only depends on the instance $G = (V, E^+, E^-)$. Thus, maximizing the log-likelihood function over $\mathcal{C}$ is equivalent to minimizing the following objective

$$\mathbf{w}^+(\#\text{ disagreeing "+"edges}) + \mathbf{w}^-(\#\text{ disagreeing "}-\text{"edges}).$$

Note that we have $\mathbf{w}^+ > \mathbf{w}^-$ when $p^+ < 1/2 < q^-$ (case (i) above); in this case, a "+" edge gives a stronger signal than a "−" edge. Similarly, we have $\mathbf{w}^- > \mathbf{w}^+$ when $q^- < 1/2 < p^+$ (case (ii) above); in this case, a "−" edge gives a stronger signal than a "+" edge.

2. Algorithm

In this section, we present an approximation algorithm for Correlation Clustering with Asymmetric Classification Errors. The algorithm first solves a standard LP relaxation and assigns every edge a length of x_{uv} (see Section 2.1). Then, one by one it creates new clusters and removes them from the graph. The algorithm creates a cluster C as follows. It picks a random vertex p, called a pivot, among yet unassigned vertices and a random number $R \in [0, 1]$. Then, it adds the pivot p and all vertices u with $f(x_{pu}) \leq R$ to C, where $f : [0, 1] \to [0, 1]$ is a properly chosen function, which we define below. We give a pseudo-code for this algorithm in Algorithm 1.

Our algorithm resembles the LP-based correlation clustering algorithms by Ailon et al. (2008) and Chawla et al. (2015). However, a crucial difference between our algorithm and above mentioned algorithms is that our algorithm uses a "dependant" rounding. That is, if for two edges pv_1 and pv_2, we have $f(x_{pv_1}) \leq R$ and $f(x_{pv_2}) \leq R$ at some step t of the algorithm then both v_1 and v_2 are added to the new cluster S_t. The algorithms by Ailon et al. (2008) and Chawla et al. (2015) make decisions on whether to add v_1 to S_t and v_2 to S_t, independently. Also, the choice of the function f is quite different from the functions used by Chawla et al. (2015). In fact, it is influenced by the paper by Garg, Vazirani, and Yannakakis (1996).

Algorithm 1 Approximation Algorithm

input An instance of Correlation Clustering with Asymmetric Weights $G = (V, E^+, E^-, \mathbf{w}_e)$.
  Initialize $t = 0$ and $V_t = V$.
  while $V_t \neq \varnothing$ **do**
    Pick a random pivot $p_t \in V_t$.
    Choose a radius R uniformly at random in $[0, 1]$.
    Create a new cluster S_t; add the pivot p_t to S_t.
    for all $u \in V_t$ **do**
      if $f(x_{p_t u}) \leq R$ **then**
        Add u to S_t.
      end if
    end for
    Let $V_{t+1} = V_t \setminus S_t$ and $t = t + 1$.
  end while
output clustering $\mathcal{S} = (S_0, \ldots, S_{t-1})$.

2.1. Linear Programming Relaxation

In this section, we describe a standard linear programming (LP) relaxation for Correlation Clustering which was introduced by Charikar, Guruswami, and Wirth (2003). We first give an integer programming formulation of the Correlation Clustering problem. For every pair of vertices u and v, the integer program (IP) has a variable $x_{uv} \in \{0, 1\}$, which indicates whether u and v belong to the same cluster:

- $x_{uv} = 0$, if u and v belong to the same cluster; and
- $x_{uv} = 1$, otherwise.

We require that $x_{uv} = x_{vu}$, $x_{uu} = 0$ and all x_{uv} satisfy the triangle inequality. That is, $x_{uv} + x_{vw} \geq x_{uw}$.

Every feasible IP solution x defines a partitioning $\mathcal{S} = (S_1, \ldots, S_T)$ in which two vertices u and v belong to the same cluster if and only if $x_{uv} = 0$. A positive edge uv is in disagreement with this partitioning if and only if $x_{uv} = 1$; a negative edge uv is in disagreement with this partitioning if and only if $x_{uv} = 0$. Thus, the cost of the partitioning is given by the following linear function:

$$\sum_{uv \in E^+} \mathbf{w}_{uv} x_{uv} + \sum_{uv \in E^-} \mathbf{w}_{uv}(1 - x_{uv}).$$

We now replace all integrality constraints $x_{uv} \in \{0, 1\}$ in the integer program with linear constraints $x_{uv} \in [0, 1]$. The obtained linear program is given in Figure 1. In the paper, we refer to each variable x_{uv} as the length of the edge uv.

$$\min \sum_{uv \in E^+} \mathbf{w}_{uv} x_{uv} + \sum_{uv \in E^-} \mathbf{w}_{uv}(1 - x_{uv}).$$

subject to

$$\begin{aligned} x_{uw} &\leq x_{uv} + x_{vw} && \text{for all } u, v, w \in V \\ x_{uv} &= x_{vu} && \text{for all } u, v \in V \\ x_{uu} &= 0 && \text{for all } u \in V \\ x_{uv} &\in [0, 1] && \text{for all } u, v \in V \end{aligned}$$

Figure 1. LP relaxation

Algorithm 2 One iteration of Algorithm 1 on triangle uvw

Pick a random pivot $p \in \{u, v, w\}$.
Choose a random radius R with the uniform distribution in $[0, 1]$.
Create a new cluster S. Insert p in S.
for all $a \in \{u, v, w\} \setminus \{p\}$ **do**
  if $f(x_{pa}) \leq R$ **then**
    Add a to S .
  end if
end for

3. Analysis of the Algorithm

The analysis of our algorithm follows the general approach proposed by Ailon, Charikar, and Newman (2008). Ailon et al. (2008) observed that in order to get upper bounds on the approximation factors of their algorithms, it is sufficient to consider how these algorithms behave on triplets of vertices. Below, we present their method adapted to our settings. Then, we will use Theorem 3.1 to analyze our algorithm.

3.1. General Approach: Triple-Based Analysis

Consider an instance of Correlation Clustering $G = (V, E^+, E^-)$ on three vertices u, v, w. Suppose that the edges uv, vw, and uw have signs $\sigma_{uv}, \sigma_{vw}, \sigma_{uw} \in \{\pm\}$, respectively. We shall call this instance a triangle (u, v, w) and refer to the vector of signs $\sigma = (\sigma_{vw}, \sigma_{uw}, \sigma_{uv})$ as the signature of the triangle (u, v, w).

Let us now assign arbitrary lengths x_{uv}, x_{vw}, and x_{uw} satisfying the triangle inequality to the edges uv, vw, and uw and run one iteration of our algorithm on the triangle uvw (see Algorithm 2).

We say that a positive edge uv is in disagreement with S if $u \in S$ and $v \notin S$ or $u \notin S$ and $v \in S$. Similarly, a negative edge uv is in disagreement with S if $u, v \in S$. Let $cost(u, v \mid w)$ be the probability that the edge (u, v) is in disagreement with S given that w is the pivot.

$$cost(u, v \mid w) = \begin{cases} \Pr[\mathcal{K} \mid p = w], & \text{if } \sigma_{uv} = \text{“+”}; \\ \Pr[\mathcal{L} \mid p = w], & \text{if } \sigma_{uv} = \text{“−”}. \end{cases}$$

where $\mathcal{K}$ denotes the event of $u \in S, v \notin S$ or $u \notin S, v \in S$ and $\mathcal{L}$ denotes the event of $u \in S, v \in S$. Let $lp(u, v \mid w)$ be the LP contribution of the edge (u, v) times the probability of it being removed, conditioned on w being the pivot.

$$lp(u, v \mid w) = \begin{cases} x_{uv} \cdot \Pr[\mathcal{M} \mid p = w], & \text{if } \sigma_{uv} = \text{“+”}; \\ (1 - x_{uv}) \cdot \Pr[\mathcal{M} \mid p = w], & \text{if } \sigma_{uv} = \text{“−”}. \end{cases}$$

where $\mathcal{M}$ denotes the event of $u \in S$ or $v \in S$. We now define two functions $ALG^\sigma(x, y, z)$ and $LP^\sigma(x, y, z)$. To this end, construct a triangle (u, v, w) with signature σ edge lengths x, y, z (where $x_{vw} = x$, $x_{uw} = y$, $x_{uv} = z$). Then,

$$\begin{aligned} ALG^\sigma(x, y, z) = {} & \mathbf{w}_{uv} \cdot cost(u, v \mid w) \\ & + \mathbf{w}_{uw} \cdot cost(u, w \mid v) \\ & + \mathbf{w}_{vw} \cdot cost(v, w \mid u); \end{aligned}$$

$$\begin{aligned} LP^\sigma(x, y, z) = {} & \mathbf{w}_{uv} \cdot lp(u, v \mid w) \\ & + \mathbf{w}_{uw} \cdot lp(u, w \mid v) \\ & + \mathbf{w}_{vw} \cdot lp(v, w \mid u). \end{aligned}$$

We will use the following theorem from the paper by Chawla, Makarychev, Schramm, and Yaroslavtsev (2015) (Lemma 4) to analyze our algorithm. This theorem was first proved by Ailon, Charikar, and Newman (2008) but it was not stated in this form in their paper.

Theorem 3.1 (see (Ailon et al., 2008) and (Chawla et al., 2015))**.** *Consider a function f with $f(0) = 0$. If for all signatures $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ (where each $\sigma_i \in \{\pm\}$) and edge lengths x, y, and z satisfying the triangle inequality, we have $ALG^\sigma(x, y, z) \leq \rho LP^\sigma(x, y, z)$, then the approximation factor of the algorithm is at most ρ.*

3.2. Analysis of the Approximation Algorithm

Proof of Theorem 1.1. Without loss of generality we assume that the scaling parameter $\mathbf{w}$ is 1. We consider two cases $\alpha \leq 0.169$ and $\alpha \geq 0.169$. To simplify the exposition, here we consider the more interesting case of $\alpha \leq 0.169$; we consider the other case in the full version of the paper. Define $f(x)$ as follows (see Figure 2),

$$f(x) = \begin{cases} 1 - e^{-Ax}, & \text{if } 0 \leq x < \frac{1}{2} - \frac{1}{2A}; \\ 1, & \text{otherwise;} \end{cases}$$

where $A = 3 + 2 \log_e 1/\alpha$.

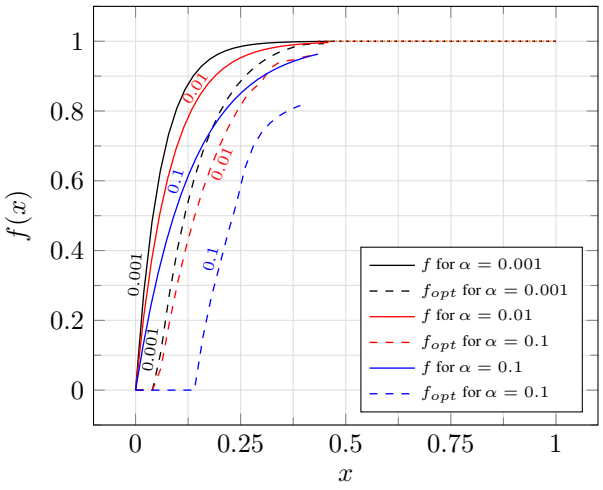

Figure 2. This plot shows functions $f(x)$ used in the proof of Theorem 1.1 for $\alpha \in \{0.001, 0.01, 0.1\}$. Additionally, it shows optimal functions $f_{opt}(x)$ (see Section 4 for details). Note that every function $f(x)$, including $f_{opt}(x)$, has a discontinuity at point $\tau = 1/2 - 1/2A$; for $x \geq \tau$, $f(x) = 1$.

Our analysis of the algorithm relies on Theorem 3.1. We will show that for every triangle (u_1, u_2, u_3) with edge lengths (x_1, x_2, x_3) (satisfying the triangle inequality) and signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$, we have

$$ALG^{\sigma}(x_1, x_2, x_3) \leq A \cdot LP^{\sigma}(x_1, x_2, x_3). \qquad (1)$$

Therefore, by Theorem 3.1, our algorithm gives an A-approximation.

Without loss of generality, we assume that $x_1 \leq x_2 \leq x_3$. When $i \in \{1, 2, 3\}$ is fixed, we will denote the other two elements of $\{1, 2, 3\}$ by k and j, so that $j < k$. For $i \in \{1, 2, 3\}$, let $e_i = (u_j, u_k)$ (the edge opposite to u_i), $w_i = \mathbf{w}_{e_i}$, $x_i = x_{u_j u_k}$, $y_i = f(x_i)$, and $t_i = A \cdot lp(u_j, u_k|u_i) - cost(u_j, u_k|u_i)$. Observe that (1) is equivalent to

$$\sum_{i=1}^{3} w_i t_i \geq 0. \qquad (2)$$

Now express t_i's in terms of x_i's and y_i's.

Claim 3.2. *For every $i \in \{1, 2, 3\}$, we have*

$$t_i = \begin{cases} A(1 - y_j)x_i - (y_k - y_j), & \text{if } \sigma_i = \text{“+”} \\ A(1 - y_j)(1 - x_i) - (1 - y_k), & \text{if } \sigma_i = \text{“−”} \end{cases}$$

Proof. First assume that $\sigma_i =$ “+”. Then

$$\begin{aligned} t_i &= A \cdot lp(u_j, u_k|u_i) - cost(u_j, u_k|u_i) \\ &= Ax_{u_j u_k} \cdot \Pr[u_j \in S \text{ or } u_k \in S \mid p = u_i] \\ &- \Pr[u_j \in S, u_k \notin S \text{ or } u_j \notin S, u_k \in S \mid p = u_i] \\ &= Ax_i \cdot \Pr[f(x_k) \leq R \text{ or } f(x_j) \leq R] \\ &- \Pr[f(x_k) \leq R < f(x_j) \text{ or } f(x_j) \leq R < f(x_k)] \\ &= Ax_i(1 - y_j) - (y_k - y_j), \end{aligned}$$

where we used that $y_k = f(x_k) \geq f(x_j) = y_j$ (since $x_k \geq x_j$ and $f(x)$ is non-decreasing). Now assume that $\sigma_i =$ “−”. Similarly, we have

$$\begin{aligned} t_i &= A \cdot lp(u_j, u_k|u_i) - cost(u_j, u_k|u_i) \\ &= A(1 - x_{u_j u_k}) \cdot \Pr[u_j \in S \text{ or } u_k \in S \mid p = u_i] \\ &- \Pr[u_j \in S, u_k \in S \mid p = w] \\ &= A(1 - x_i) \cdot \Pr[f(x_k) \leq R \text{ or } f(x_j) \leq R] \\ &- \Pr[f(x_k) \leq R, f(x_j) \leq R] \\ &= A(1 - x_i) \cdot (1 - y_j) - (1 - y_k). \end{aligned}$$

□

Let us say that edge e_i *pays for itself* if $t_i \geq 0$. Note that if all edges e_1, e_2, e_3 pay for themselves then (2) holds and we are done. We now show that negative edges pay for themselves.

Claim 3.3. *If $\sigma_i =$ “−”, then $t_i \geq 0$.*

Proof. We need to show that $A(1 - y_j)(1 - x_i) \geq 1 - y_k$. First, if $x_k \geq \frac{1}{2} - \frac{1}{2A}$ then $y_k = 1$ and the inequality trivially holds. Suppose that $x_k < \frac{1}{2} - \frac{1}{2A}$. Then $A > \frac{1}{1-2x_k} \geq \frac{1}{1-x_k-x_j} \geq \frac{1}{1-x_i}$ (here, we used the triangle inequality $x_k + x_j \geq x_i$). Thus

$$A(1 - y_j)(1 - x_i) \geq A(1 - y_k)(1 - x_i) \geq 1 - y_k.$$

□

Positive edges do not necessarily pay for themselves. However, if $x_3 < \frac{1}{2} - \frac{1}{2A}$, then all edges pay for themselves.

Claim 3.4. *Suppose that $x_3 < \frac{1}{2} - \frac{1}{2A}$. Then $t_i \geq 0$ for every i.*

Proof. Since $x_3 < \frac{1}{2} - \frac{1}{2A}$, for every $i \in \{1, 2, 3\}$ we have $x_i < \frac{1}{2} - \frac{1}{2A}$ and thus $y_i = f(x_i) = 1 - e^{-Ax_i}$.

Let us now fix i and prove that $t_i \geq 0$. If $\sigma_i =$ “−”, then, by Claim 3.3, $t_i \geq 0$ and we are done. So we assume that $\sigma_i =$ “+”. Then,

$$\begin{aligned} y_k - y_j = e^{-Ax_j} - e^{-Ax_k} &= e^{-Ax_j}\left(1 - e^{-A(x_k - x_j)}\right) \\ &\leq e^{-Ax_j} A(x_k - x_j) \leq e^{-Ax_j} Ax_i = A(1 - y_j)x_i \end{aligned}$$

where the first inequality follows from the inequality $1 - e^x \leq -x$, and the second inequality follows from the triangle inequality. Thus, $t_i = A(1 - y_j)x_i - (y_k - y_j) \geq 0$. □

We conclude that if $x_3 < 1/2 - \frac{1}{2A}$, then (2) holds and we are done. The case $x_3 < 1/2 - \frac{1}{2A}$ is the most interesting case in the analysis; the rest of the proof is more technical. As a side note, let us point out that Theorem 1.1 has dependence $A = 3 + 2\log_e 1/\alpha$ because (i) $f(x)$ must be equal to $C - e^{-Ax}$ or a slower growing function so that Claim 3.4 holds (ii) Theorem 3.1 requires that $f(0) = 0$, and finally (iii) we will need below that $1 - f\left(\frac{1}{2} - \frac{3}{2A}\right) \le \alpha$.

From now on, we assume that $x_3 \ge \frac{1}{2} - \frac{1}{2A}$. We show that positive edges of length at least $1/A$ pay for themselves.

Claim 3.5. *If $\sigma_i =$ "+" and $x_i \ge 1/A$, then $t_i \ge 0$.*

Proof. We have, $t_i = A(1-y_j)x_i - (y_k - y_j) \ge (1 - y_j) - (y_k - y_j) = 1 - y_k \ge 0$. □

Now we prove that we may assume that $\sigma_i =$ "+" if $x_i < 1/2 - 1/(2A)$.

Claim 3.6. *Suppose that $x_i < \frac{1}{2} - \frac{1}{2A}$. If (2) holds for σ with $\sigma_i =$ "+", then (2) also holds for σ' obtained from σ by changing the sign of σ_i to "−".*

Proof. To prove the claim, we show that the value of t_i is greater for σ' than for σ. That is, $A(1-y_j)x_i - (y_k - y_j) < A(1-y_j)(1-x_i) - (1-y_k)$. (Note that the values of t_j and t_k do not depend on σ_i and thus do not change if we replace σ with σ'). Since f is non-decreasing, $y_k \ge y_j$.

$$x_i < \frac{1}{2} - \frac{1}{2A} = \frac{1}{2} + \frac{1}{2A} - \frac{1}{A} \le \frac{1}{2} + \frac{1}{2A} - \frac{(1-y_k)}{A(1-y_j)}.$$

Thus, $2A(1-y_j)x_i < A(1-y_j) + 1 - y_j - 2(1-y_k)$. Therefore, $A(1-y_j)x_i - (y_k - y_j) < A(1-y_j)(1-x_i) - (1-y_k)$, as required. □

Observe that if $x_1 \ge \frac{1}{A}$, then all $x_i \ge \frac{1}{A}$ and thus, by Claims 3.3 and 3.5, all $t_i \ge 0$ and we are done. Similarly, if $x_2 \ge \frac{1}{2} - \frac{1}{2A} \ge \frac{1}{A}$ (since $A \ge 3$), then $t_2 \ge 0$ and $t_3 \ge 0$; additionally, $y_2 = y_3 = 1$, thus $t_1 = 0$ and we are done. Therefore, we will assume below that

$$x_1 < \frac{1}{A}, \quad x_2 < \frac{1}{2} - \frac{1}{2A}, \quad x_3 \ge \frac{1}{2} - \frac{1}{2A}$$

(the last assumption was made above). By Claim 3.6, we may also assume that $\sigma_1 =$ "+" and $\sigma_2 =$ "+". Recall that we assumed that $\alpha < 0.169$. In this regime, $A > 5$ and, therefore, $x_2 \ge x_3 - x_1 \ge (\frac{1}{2} - \frac{1}{2A}) - \frac{1}{A} > \frac{1}{A}$ and $x_3 \ge \frac{1}{2} - \frac{1}{2A} > \frac{1}{A}$. Thus, by Claims 3.3 and 3.5, $t_2 \ge 0$ and $t_3 \ge 0$ (edges e_2 and e_3 pay for themselves). If $t_1 \ge 0$, we are done. So we will assume below that $t_1 < 0$. Then,

$$w_1 t_1 + w_2 t_2 + w_3 t_3 \ge 1 \cdot t_1 + \alpha t_2 + \alpha t_3 \qquad (3)$$

(recall that we assume that e_1 is a positive edge and thus $w_1 \le 1$).

Since $x_3 \ge \frac{1}{2} - \frac{1}{2A}$, we have $y_3 = 1$. Now we separately consider two possible signatures $\sigma =$ ("+", "+", "+") and $\sigma =$ ("+", "+", "−").

First, assume that $\sigma =$ ("+", "+", "+"). Because of (3), to prove (2), it is sufficient to show

$$(1-y_2) + \alpha(1-y_1) + \alpha(y_2 - y_1) \le \\ \le A(1-y_2)x_1 + \alpha A(1-y_1)x_2 + \alpha A(1-y_1)x_3.$$

Note that $x_2 \ge x_3 - x_1 \ge \frac{1}{2} - \frac{1}{2A} - \frac{1}{A} = \frac{1}{2} - \frac{3}{2A}$. Therefore,

$$1 - y_2 \le 1 - \left(1 - e^{-A\left(\frac{1}{2} - \frac{3}{2A}\right)}\right) = e^{-\frac{3}{2} - \log_e \frac{1}{\alpha} + \frac{3}{2}} \\ = e^{-\log_e \frac{1}{\alpha}} = \alpha.$$

Thus, $(1-y_2) + \alpha(1-y_1) + \alpha(y_2 - y_1) \le \alpha y_2 + 2\alpha(1-y_1)$. To finish the analysis of the case $\sigma =$ ("+", "+", "+"), it is sufficient to show that

$$\alpha y_2 + 2\alpha(1-y_1) \le A(1-y_2)x_1 + \alpha A(1-y_1)x_2 \\ + \alpha A(1-y_1)x_3.$$

This inequality immediately follows from the following claim (we simply need to add up (4) and (5) and multiply the result by α).

Claim 3.7. *For $c = 0.224$, we have*

$$(2-c)(1-y_1) \le A(1-y_1)x_2; \text{ and} \qquad (4)$$
$$y_2 + c(1-y_1) \le A(1-y_1)x_3. \qquad (5)$$

Proof. Since $c \ge 2 - \log_e \frac{1}{0.169} \ge 2 - \log_e \frac{1}{\alpha}$ (recall that $\alpha \le 0.169$), we have $2 - c \le \log_e \frac{1}{\alpha} = \frac{A}{2} - \frac{3}{2} \le Ax_2$. Therefore, (4) holds.

We also have, $c \le 0.169 + \log_e \frac{1}{0.169} + 1 - e \le \alpha + \log_e \frac{1}{\alpha} + 1 - e$. Thus, $e - \alpha \le \frac{A}{2} - \frac{1}{2} - c \le Ax_3 - c$. Therefore,

$$e^{-1}(Ax_3 - c) \ge 1 - \alpha e^{-1} \\ = 1 - e^{-A\left(\frac{1}{2} - \frac{1}{2A}\right)} \ge y_2, \qquad (6)$$

where we used that $x_2 < \frac{1}{2} - \frac{1}{2A}$ and $y_2 = f(x_2) = 1 - e^{-Ax_2}$. Observe that from inequalities (6) and $x_1 < \frac{1}{A}$ it follows that

$$y_2 \le \left(1 - f\left(\frac{1}{A}\right)\right)(Ax_3 - c) \le (1-y_1)(Ax_3 - c),$$

which implies (5). □

Now, assume that $\sigma =$ ("+", "+", "−"). We need to prove the following inequality,

$$(1-y_2) + \alpha(1 - y_1 + 1 - y_2) \le \\ \le A(1-y_2)x_1 + \alpha A(1-y_1)(x_2 + 1 - x_3). \qquad (7)$$

As before,

$$(1-y_2)+\alpha(1-y_1+1-y_2)\le \\ \le \alpha+\alpha(1-y_1+1-y_2)\le \alpha+2\alpha(1-y_1). \quad (8)$$

On the other hand,

$$\begin{aligned} A(1-y_2)x_1+\alpha A(1-y_1)(x_2+1-x_3)\ge \quad (9)\\ \ge \alpha A(1-y_1)(1-x_1+x_1+x_2-x_3)\\ \ge \alpha A(1-y_1)(1-x_1)\\ \ge \alpha A(1-y_1)\left(1-\frac{1}{A}\right)=\alpha(1-y_1)(A-1)\end{aligned}$$

where the second inequality is due to the triangle inequality and the third inequality is due to $x_1<\frac{1}{A}$. Finally, observe that $1\le 2e^{-1}\log_e\frac{1}{\alpha}=e^{-1}(A-3)\le(1-y_1)(A-3)$. We get,

$$\alpha(1-y_1)(A-1)\ge\alpha+2\alpha(1-y_1). \quad (10)$$

Combining (8), (9), and (10), we get (7).

This concludes the case analysis and the proof of Theorem 1.1. □

4. Better approximation for values of α appearing in practice

We note that the choice of function $f(x)$ in Theorem 1.1 is somewhat suboptimal. However, for every $\alpha\in(0,1]$, we can compute the optimal function $f_{opt}(x)$ (with high precision) using linear programming. Using this function f_{opt}, we can achieve an approximation factor A_{opt} better than the approximation factor $A_{thm}=3+2\log_e 1/\alpha$ guaranteed by Theorem 1.1.[2] While asymptotically $A_{thm}/A_{opt}\to 1$ as $\alpha\to 0$, A_{opt} is noticeably better than A_{thm} for many values of α that are likely to appear in practice (say, for $\alpha\in(10^{-8},0.1)$). We list approximation factors A_{thm} and A_{opt} for several values of α in Table 1; we also plot the dependence of A_{thm} and A_{opt} on α in Figure 3.

Table 1. Approximation factors A_{thm} and A_{opt} for different α-s.

$\log_e 1/\alpha$	$1/\alpha$	A_{thm}	A_{opt}
0	1	3	3
1.61	5	6.22	4.32
2.30	10	7.61	4.63
3.91	50	10.82	6.07
4.61	100	12.21	6.78
6.21	500	15.43	8.69
6.91	1000	16.82	9.62
8.52	5 000	20.03	11.9
10	22 026.5	23	14.2
15	3.3×10^6	33	22.6
20	4.9×10^8	43	31.3

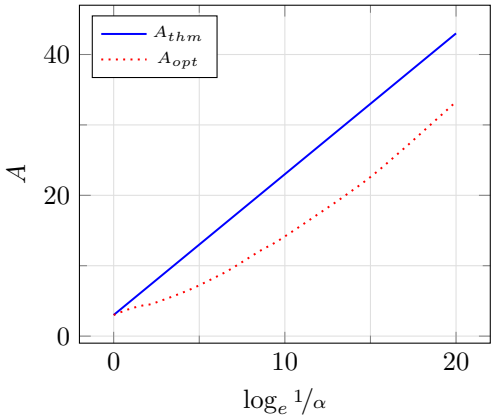

Figure 3. Plots of approximation factors A_{thm} and A_{opt}.

5. Integrality Gap

In this section, we give a $\Theta(\log 1/\alpha)$ integrality gap example for the LP relaxation presented in Section 2.1. Notice that in the example each positive edge has a weight of $\mathbf{w}^+$ and each negative edge has a weight of $\mathbf{w}^-$ with $\mathbf{w}^+\ge\mathbf{w}^-$.

Proof of Theorem 1.3. Consider a 3 regular expander $G=(V,E)$ on $n=\Theta((\alpha^2\log^2\alpha)^{-1})$ vertices. We say that two vertices u and v are similar if $(u,v)\in E$; otherwise u and v are dissimilar. That is, the set of positive edges E^+ is E and the set of negative edges E^- is $V\times V\setminus E$. Let $\mathbf{w}^+=1$ and $\mathbf{w}^-=\alpha$.

Lemma 5.1. *The integrality gap of the Correlation Clustering instance $G_{cc}=(V,E^+,E^-)$ described above is $\Theta(\log 1/\alpha)$.*

Proof. Let $d(u,v)$ be the shortest path distance in G. Let $\varepsilon=2/\log_3 n$. We define a feasible metric LP solution as follows: $x_{uv}=\min(\varepsilon d(u,v),1)$.

Let LP^+ be the LP cost of positive edges, and LP^- be the LP cost of negative edges. The LP cost of every positive edge is ε since $d(u,v)=1$ for $(u,v)\in E$. There are $3n/2$ positive edges in G_{cc}. Thus, $LP_+<3n/\log_3 n$. We now estimate LP^-. For every vertex u, the number of vertices v at distance less than t is upper bounded by 3^t because G is a 3-regular graph. Thus, the number of vertices v at distance less than $1/2\log_3 n$ is upper bounded by $\sqrt{n}$. Observe that the LP cost of a negative edge (u,v) (which is equal to $\alpha(1-x_{uv})$) is positive if and only if $d(u,v)<1/2\log_3 n$. Therefore, the number of negative edges with a positive LP cost incident on any vertex u is at most $\sqrt{n}$.

[2] It is also possible to slightly modify Algorithm 1 so that it gets approximation A_{opt} without explicitly computing f. We omit the details here.

Consequently, the LP cost of all negative edges is upper bounded by $\alpha n^{\frac{3}{2}} = \Theta(n/\log 1/\alpha)$. Hence,

$$LP \leq \Theta(n/\log 1/\alpha) + 3n/\log_3 n = \Theta(n/\log 1/\alpha).$$

Here, we used that $\log n = \Theta(\log 1/\alpha)$.

We now lower bound the cost of the optimal (integral) solution. Consider an optimal solution. There are two possible cases.

1. No cluster contains 90% of the vertices. Then a constant fraction of positive edges in the expander G are cut and, therefore, the cost of the optimal clustering is at least $\Theta(n)$.
2. One of the clusters contains at least 90% of all vertices. Then all negative edges in that cluster are in disagreement with the clustering. There are at least $\binom{0.9n}{2} - m = \Theta(n^2)$ such edges. Their cost is at least $\Omega(\alpha n^2)$.

We conclude that the cost of the optimal solution is at least $\Theta(n)$ and, thus, the integrality gap is $\Theta(\log(1/\alpha))$. □

We note that in this example $\log(1/\alpha) = \Theta(\log n)$. However, it is easy to construct an integrality gap example where $\log(1/\alpha) \ll \Theta(\log n)$. To do so, we pick the integrality gap example constructed above and create $k \gg n$ disjoint copies of it. To make the graph complete, we add negative edges with (fractional) LP value equal to 1 to connect each copy to every other copy of the graph. The new graph has $kn \gg n$ vertices. However, the integrality gap remains the same, $\Theta(\log 1/\alpha)$. □

Acknowledgements

Jafar Jafarov and Yury Makarychev were supported in part by NSF CCF-1718820 and NSF TRIPODS CCF-1934843. Sanchit Kalhan and Konstantin Makarychev were supported in part by NSF TRIPODS CCF-1934931.

References

Ailon, N., Charikar, M., and Newman, A. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM (JACM)*, 55(5):23, 2008.

Ailon, N., Chen, Y., and Xu, H. Breaking the small cluster barrier of graph clustering. In *International Conference on Machine Learning*, pp. 995–1003, 2013.

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

Boldi, P. and Vigna, S. The WebGraph framework I: Compression techniques. In *Proc. of the Thirteenth International World Wide Web Conference*, pp. 595–601, 2004.

Boldi, P., Codenotti, B., Santini, M., and Vigna, S. Ubicrawler: A scalable fully distributed web crawler. *Software: Practice & Experience*, 34(8):711–726, 2004.

Boldi, P., Rosa, M., Santini, M., and Vigna, S. Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks. In *Proceedings of the International Conference on World Wide Web*, pp. 587–596, 2011.

Boldi, P., Marino, A., Santini, M., and Vigna, S. BUbiNG: Massive crawling for the masses. In *Proceedings of the Companion Publication of the International Conference on World Wide Web*, pp. 227–228, 2014.

Charikar, M., Guruswami, V., and Wirth, A. Clustering with qualitative information. In *IEEE Symposium on Foundations of Computer Science*. Citeseer, 2003.

Chawla, S., Makarychev, K., Schramm, T., and Yaroslavtsev, G. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. In *Proceedings of the Symposium on Theory of Computing*, pp. 219–228, 2015.

Demaine, E. D., Emanuel, D., Fiat, A., and Immorlica, N. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187, 2006.

Elsner, M. and Schudy, W. Bounding and comparing methods for correlation clustering beyond ilp. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pp. 19–27. Association for Computational Linguistics, 2009.

Garg, N., Vazirani, V. V., and Yannakakis, M. Approximate max-flow min-(multi) cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.

Makarychev, K., Makarychev, Y., and Vijayaraghavan, A. Correlation clustering with noisy partial information. In *Conference on Learning Theory*, pp. 1321–1342, 2015.

Mathieu, C. and Schudy, W. Correlation clustering with noisy input. In *Proceedings of the Symposium on Discrete Algorithms*, pp. 712–728, 2010.

Pan, X., Papailiopoulos, D., Oymak, S., Recht, B., Ramchandran, K., and Jordan, M. I. Parallel correlation clustering on big graphs. In *Advances in Neural Information Processing Systems*, pp. 82–90, 2015.

Tang, S., Andres, B., Andriluka, M., and Schiele, B. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision*, pp. 100–111, 2016.

Tang, S., Andriluka, M., Andres, B., and Schiele, B. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 3539–3548, 2017.