A Faster Algorithm for Constrained Correlation Clustering

Nick Fischer
INSAIT, Sofia University "St. Kliment Ohridski", Bulgaria

Evangelos Kipouridis
Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Jonas Klausen
BARC, University of Copenhagen, Denmark

Mikkel Thorup
BARC, University of Copenhagen, Denmark

Abstract

In the Correlation Clustering problem we are given n nodes, and a preference for each pair of nodes indicating whether we prefer the two endpoints to be in the same cluster or not. The output is a clustering inducing the minimum number of violated preferences. In certain cases, however, the preference between some pairs may be too important to be violated. The constrained version of this problem specifies pairs of nodes that *must* be in the same cluster as well as pairs that *must not* be in the same cluster (hard constraints). The output clustering has to satisfy all hard constraints while minimizing the number of violated preferences.

Constrained Correlation Clustering is APX-Hard and has been approximated within a factor 3 by van Zuylen *et al.* [SODA '07]. Their algorithm is based on rounding an LP with $\Theta(n^3)$ constraints, resulting in an $\Omega(n^{3\omega})$ running time. In this work, using a more combinatorial approach, we show how to approximate this problem significantly faster at the cost of a slightly weaker approximation factor. In particular, our algorithm runs in $\widetilde{O}(n^3)$ time (notice that the input size is $\Theta(n^2)$) and approximates Constrained Correlation Clustering within a factor 16.

To achieve our result we need properties guaranteed by a particular influential algorithm for (unconstrained) Correlation Clustering, the CC-PIVOT algorithm. This algorithm chooses a *pivot* node u, creates a cluster containing u and all its preferred nodes, and recursively solves the rest of the problem. It is known that selecting pivots at random gives a 3-approximation. As a byproduct of our work, we provide a derandomization of the CC-PIVOT algorithm that still achieves the 3-approximation; furthermore, we show that there exist instances where no ordering of the pivots can give a $(3-\varepsilon)$-approximation, for any constant ε.

Finally, we introduce a node-weighted version of Correlation Clustering, which can be approximated within factor 3 using our insights on Constrained Correlation Clustering. As the general weighted version of Correlation Clustering would require a major breakthrough to approximate within a factor $o(\log n)$, Node-Weighted Correlation Clustering may be a practical alternative.

2012 ACM Subject Classification Theory of computation → Facility location and clustering

Keywords and phrases Clustering, Constrained Correlation Clustering, Approximation

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.32

Related Version *Full Version*: `https://arxiv.org/abs/2501.03154` [26]

Funding *Jonas Klausen* and *Mikkel Thorup* are part of BARC, Basic Algorithms Research Copenhagen, supported by VILLUM Foundation grants 16582 and 54451. This work is part of the project TIPEA that has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No. 850979).
Nick Fischer: Partially funded by the Ministry of Education and Science of Bulgaria's support for INSAIT, Sofia University "St. Kliment Ohridski" as part of the Bulgarian National Roadmap for Research Infrastructure. Parts of this work were done while the author was at Saarland University.

Acknowledgements We thank Lorenzo Beretta for his valuable suggestions on weighted sampling.

1 Introduction

Clustering is a fundamental task related to unsupervised learning, with many applications in machine learning and data mining. The goal of clustering is to partition a set of nodes into disjoint clusters, such that (ideally) all nodes within a cluster are similar, and nodes in different clusters are dissimilar. As no single definition best captures this abstract goal, a lot of different clustering objectives have been suggested.

Correlation Clustering is one of the most well studied such formulations for a multitude of reasons: Its definition is simple and natural, it does not need the number of clusters to be part of the input, and it has found success in many applications. Some few examples include automated labeling [1, 15], clustering ensembles [10], community detection [19, 36], disambiguation tasks [30], duplicate detection [5] and image segmentation [31, 40].

In Correlation Clustering we are given a graph $G = (V, E)$, and the output is a partition (clustering) $C = \{C_1, \dots, C_k\}$ of the vertex set V. We refer to the sets C_i of C as *clusters*. The goal is to minimize the number of edges between different clusters plus the number of non-edges inside of clusters. More formally, the goal is to minimize $|E \triangle E_C|$, the cardinality of the symmetric difference between E and E_C, where we define $E_C = \bigcup_{i=1}^{k} \binom{C_i}{2}$. In other words, the goal is to transform the input graph into a collection of cliques with the minimal number of edge insertions and deletions. An alternative description used by some authors is that we are given a *complete* graph where the edges are labeled either "+" (corresponding to the edges in our graph G) or "−" (corresponding to the non-edges in our graph G).

The problem is typically motivated as follows: Suppose that the input graph models the relationships between different entities which shall be grouped. An edge describes that we prefer its two endpoints to be clustered together, whereas a non-edge describes that we prefer them to be separated. In this formulation the cost of a correlation clustering is the number of violated preferences.

1.1 Previous Results

Correlation Clustering was initially introduced by Bansal, Blum, and Chawla [7], who proved that it is NP-Hard, and provided a deterministic constant-factor approximation, the constant being larger than 15,000. Subsequent improvements were based on rounding the natural LP: Charikar, Guruswami and Wirt gave a deterministic 4-approximation [17], Ailon, Charikar and Newman gave a randomized 2.5-approximation and proved that the problem is APX-Hard [3], while a deterministic 2.06-approximation was given by Chawla, Makarychev, Schramm and Yaroslavtsev [18]. The last result is near-optimal among algorithms rounding the natural LP, as its integrality gap is at least 2. In a breakthrough result by Cohen-Addad, Lee and Newman [23] a $(1.994 + \epsilon)$-approximation using the Sherali-Adams relaxation broke the 2 barrier. It was later improved to $1.73 + \epsilon$ [22] by Cohen-Addad, Lee, Li and Newman, and even to 1.437 by Cao *et al.* [13]. There is also a combinatorial 1.847-approximation (Cohen-Addad *et al.* [24]).

Given the importance of Correlation Clustering, research does not only focus on improving its approximation factor. Another important goal is efficient running times without big sacrifices on the approximation factor. As the natural LP has $\Theta(n^3)$ constraints, using a state-of-the-art LP solver requires time $\Omega(n^{3\omega}) = \Omega(n^{7.113})$.

In order to achieve efficient running times, an algorithm thus has to avoid solving the LP using an all-purpose LP-solver, or the even more expensive Sherali-Adams relaxation; such algorithms are usually called *combinatorial* algorithms[1]. Examples of such a direction can

[1] On a more informal note, combinatorial algorithms are often not only faster, but also provide deeper insights on a problem, compared to LP-based ones.

be seen in [3] where, along with their LP-based 2.5-approximation, the authors also design a combinatorial 3-approximation (the CC-PIVOT algorithm); despite its worse approximation, it enjoys the benefit of being faster. Similarly, much later than the 2.06-approximation [18], Veldt devised a faster combinatorial 6-approximation and a 4-approximation solving a less expensive LP [35].

Another important direction is the design of *deterministic* algorithms. For example, [3] posed as an open question the derandomization of CC-PIVOT. The question was (partially) answered affirmatively by [34]. Deterministic algorithms were also explicitly pursued in [35], and are a significant part of the technical contribution of [18].

Correlation Clustering has also been studied in different settings such as parameterized algorithms [28], sublinear and streaming algorithms [6, 12, 8, 9, 12, 16], massively parallel computation (MPC) algorithms [21, 14], and differentially private algorithms [11].

PIVOT. The CC-PIVOT algorithm [3] is a very influential algorithm for Correlation Clustering. It provides a 3-approximation and is arguably the simplest constant factor approximation algorithm for Correlation Clustering. It simply selects a node uniformly at random, and creates a cluster $\mathcal{C}$ with this node and its neighbors in the (remaining) input graph. It then removes $\mathcal{C}$'s nodes and recurses on the remaining graph. Due to its simplicity, CC-PIVOT has inspired several other algorithms, such as algorithms for Correlation Clustering in the streaming model [6, 12, 8, 9, 12, 16] and algorithms for the more general Fitting Utrametrics problem [2, 20].

One can define a meta-algorithm based on the above, where we do not necessarily pick the pivots uniformly at random. Throughout this paper, we use the term PIVOT algorithm to refer to an instantiation of the (Meta-)Algorithm 1[2]. Obviously CC-PIVOT is an instantiation of PIVOT, where the pivots are selected uniformly at random.

Algorithm 1 The PIVOT meta-algorithm. CC-PIVOT is an instantiation of PIVOT where pivots are selected uniformly at random.

```
procedure PIVOT($G = (V, E)$)
1   $C \leftarrow \emptyset$
2   while $V \neq \emptyset$ do
3       Pick a pivot node $u$
        /* an instantiation of PIVOT() only needs to specify how the
           pivot is selected in each iteration                          */
4       Add a cluster containing $u$ and all its neighbors to $C$
5       Remove $u$, its neighbors and all their incident edges from $G$
6   return $C$
```

The paper that introduced CC-PIVOT [3] posed as an open question the derandomization of the algorithm. The question was partially answered in the affirmative by [34]. Unfortunately there are two drawbacks with this algorithm. First, it requires solving the natural LP, which makes its running time equal to the pre-existing (better) 2.5-approximation. Second, this

[2] This is not to be confused with the more general pivoting paradigm for Correlation Clustering algorithms. In that design paradigm, the cluster we create for each pivot is not necessarily the full set of remaining nodes with which the pivot prefers to be clustered, but can be decided in any other way (e.g. randomly, based on a probability distribution related to an LP or more general hierarchies such as the Sherali-Adams hierarchy).

algorithm does not only derandomize the order in which pivots are selected, but also decides the cluster of each pivot based on an auxiliary graph (dictated by the LP) rather than based on the original graph. Therefore it is not an instantiation of PIVOT.

Weighted Correlation Clustering. In the weighted version of Correlation Clustering, we are also given a weight for each preference. The final cost is then the sum of weights of the violated preferences. An $O(\log n)$-approximation for weighted Correlation Clustering is known by Demaine, Emanuel, Fiat and Immorlica [25]. In the same paper they show that the problem is equivalent to the Multicut problem, meaning that an $o(\log n)$-approximation would require a major breakthrough. As efficiently approximating the general weighted version seems out of reach, research has focused on special cases for which constant-factor approximations are possible [32, 33].

Constrained Correlation Clustering. Constrained Correlation Clustering is an interesting variant of Correlation Clustering capturing the idea of critical pairs of nodes. To address these situations, Constrained Correlation Clustering introduces hard constraints in addition to the pairwise preferences. A clustering is valid if it satisfies all hard constraints, and the goal is to find a valid clustering of minimal cost. We can phrase Constrained Correlation Clustering as a weighted instance of Correlation Clustering: Simply give infinite weight to pairs associated with a hard constraint and weight 1 to all other pairs.

To the best of our knowledge, the only known solution to Constrained Correlation Clustering is given in the work of van Zuylen and Williamson who designed a deterministic 3-approximation [34]. The running time of this algorithm is $O(n^{3\omega})$, where $\omega < 2.3719$ is the matrix-multiplication exponent. Using the current best bound for ω, this is $\Omega(n^{7.113})$.

1.2 Our Contribution

Our main result is the following theorem. It improves the $\Omega(n^{7.113})$ running time of the state-of-the-art algorithm for Constrained Correlation Clustering while still providing a constant (but larger than 3) approximation factor[3].

▶ **Theorem 1** (Constrained Correlation Clustering). *There is a deterministic algorithm for Constrained Correlation Clustering computing a 16-approximation in time $\widetilde{O}(n^3)$.*

We first show how to obtain this result, but with a randomized algorithm that holds with high probability, instead of a deterministic one. In order to do so, we perform a (deterministic) preprocessing step and then use the CC-PIVOT algorithm. Of course CC-PIVOT alone, without the preprocessing step, would not output a clustering respecting the hard constraints. Its properties however (and more generally the properties of PIVOT algorithms) are crucial; we are not aware of any other algorithm that we could use instead and still satisfy all the hard constraints of Constrained Correlation Clustering after our preprocessing step.

To obtain our deterministic algorithm we derandomize the CC-PIVOT algorithm.

▶ **Theorem 2** (Deterministic PIVOT). *There are the following deterministic PIVOT algorithms for Correlation Clustering:*

- *A combinatorial $(3 + \epsilon)$-approximation, for any constant $\epsilon > 0$, in time $\widetilde{O}(n^3)$.*
- *A non-combinatorial 3-approximation in time $\widetilde{O}(n^5)$.*

[3] We write $\widetilde{O}(T)$ to suppress polylogarithmic factors, i.e., $\widetilde{O}(T) = T(\log T)^{O(1)}$.

We note that the final approximation of our algorithm for Constrained Correlation Clustering depends on the approximation of the applied PIVOT algorithm. If it was possible to select the order of the pivots in a way that guarantees a better approximation, this would immediately improve the approximation of our Constrained Correlation Clustering algorithm. For this reason, we study lower bounds for PIVOT; currently, we know of instances for which selecting the pivots at random doesn't give a better-than-3-approximation in expectation [3]; however, for these particular instances there *does* exist a way to choose the pivots that gives better approximations. Ideally, we want a lower bound applying for any order of the pivots (such as the lower bound for the generalized PIVOT solving the Ultrametric Violation Distance problem in [20]). We show that our algorithm is optimal, as there exist instances where no ordering of the pivots will yield a better-than-3-approximation.

▶ **Theorem 3** (PIVOT Lower Bound). *There is no constant $\epsilon > 0$ for which there exists a PIVOT algorithm for Correlation Clustering with approximation factor $3 - \epsilon$.*

We also introduce the Node-Weighted Correlation Clustering problem, which is related to (but incomparable, due to their asymmetric assignment of weights) a family of problems introduced in [36]. As weighted Correlation Clustering is equivalent to Multicut, improving over the current $\Theta(\log n)$-approximation seems out of reach. The advantage of our alternative type of weighted Correlation Clustering is that it is natural and approximable within a constant factor.

In Node-Weighted Correlation Clustering we assign weights to the nodes, rather than to pairs of nodes. Violating the preference between nodes u, v with weights ω_u and ω_v incurs cost $\omega_u \cdot \omega_v$. We provide three algorithms computing (almost-)3-approximations for Node-Weighted Correlation Clustering:

▶ **Theorem 4** (Node-Weighted Correlation Clustering, Deterministic). *There are the following deterministic algorithms for Node-Weighted Correlation Clustering:*

- *A combinatorial $(3 + \epsilon)$-approximation, for any constant $\epsilon > 0$, in time $\widetilde{O}(n^3)$.*
- *A non-combinatorial 3-approximation in time $O(n^{7.116})$.*

▶ **Theorem 5** (Node-Weighted Correlation Clustering, Randomized). *There is a randomized combinatorial algorithm for Node-Weighted Correlation Clustering computing an expected 3-approximation in time $O(n + m)$ with high probability $1 - 1/\operatorname{poly}(n)$.*

1.3 Overview of Our Techniques

Constrained Correlation Clustering. We obtain a faster algorithm for Constrained Correlation Clustering by

1. modifying the input graph using a subroutine aware of the hard-constraints, and
2. applying a PIVOT algorithm on this modified graph.

In fact, no matter what PIVOT algorithm is used, the output clustering respects all hard constraints when the algorithm is applied on the modified graph.

To motivate this two-step procedure, we note that inputs exist where *no* PIVOT algorithm, if applied to the unmodified graph, would respect the hard constraints. One such example is the cycle on four vertices, with two vertex-disjoint edges made into hard constraints.

The solution of [34] is similar to ours, as it also modifies the graph before applying a Correlation Clustering algorithm. However, both their initial modification and the following Correlation Clustering algorithm require solving the standard LP, which is expensive ($\Omega(n^{7.113})$ time). In our case both steps are implemented with deterministic and combinatorial algorithms which brings the running time down to $\widetilde{O}(n^3)$.

For the first step, our algorithm carefully modifies the input graph so that on one hand the optimal cost is not significantly changed, and on the other hand any PIVOT algorithm on the transformed graph returns a clustering that respects all hard constraints. For the second step, we use a deterministic combinatorial PIVOT algorithm.

Concerning the effect of modifying the graph, roughly speaking we get that the final approximation factor is $(2+\sqrt{5})\cdot\alpha+3$, where α is the approximation factor of the PIVOT algorithm we use. Plugging in $\alpha = 3+\epsilon$ from Theorem 2 we get the first combinatorial constant-factor approximation for Constrained Correlation Clustering in $\widetilde{O}(n^3)$ time.

Node-Weighted Correlation Clustering. We generalize the deterministic combinatorial techniques from before to the Node-Weighted Correlation Clustering problem. In addition, we also provide a very efficient randomized algorithm for the problem. It relies on a weighted random sampling technique.

One way to view the algorithm is to reduce Node-Weighted Correlation Clustering to an instance of Constrained Correlation Clustering, with the caveat that the new instance's size depends on the weights (and can thus even be exponential). Each node u is replaced by a set of nodes of size related to u's weight and these nodes have constraints forcing them to be in the same cluster.

We show that we can simulate a simple randomized PIVOT algorithm on that instance, where instead of sampling uniformly at random, we sample with probabilities proportional to the weights. Assuming polynomial weights, we can achieve this in linear time. To do so, we design an efficient data structure supporting such sampling and removal of elements.

It is easy to implement such a data structure using any balanced binary search tree, but the time for constructing it and applying all operations would be $O(n\log n)$. Using a non-trivial combination of the Alias Method [38, 37] and Rejection Sampling, we achieve a linear bound.

Due to space constraints the presentation of our algorithms for Node-Weighted Correlation Clustering is deferred to the full version of the paper [26].

Deterministic PIVOT algorithms. Our algorithms are based on a simple framework by van Zuylen and Williamson [34]. In this framework we assign a nonnegative "charge" to each pair of nodes. Using these charges, a PIVOT algorithm decides which pivot to choose next. The approximation factor depends on the total charge (as compared with the cost of an optimal clustering), and the minimum charge assigned to any bad triplet (an induced subgraph $K_{1,2}$).

The reason why these bad triplets play an important role is that for any bad triplet, any clustering needs to pay at least 1. To see this, let uvw be a bad triplet with uv being the only missing edge. For a clustering to pay 0, it must be the case that both uw and vw are together. However, this would imply that uv are also together although they prefer not to.

Our combinatorial $(3+\epsilon)$-approximation uses the multiplicative weights update method, which can be intuitively described as follows: We start with a tiny charge on all pairs. Then we repeatedly find a bad triplet uvw with currently minimal charge (more precisely: for which the sum of the charges of uv, vw, wu is minimal), and scale the involved charges by $1+\epsilon$. One can prove that this eventually results in an almost-optimal distribution of charges, up to rescaling.

For this purpose it suffices to show that the total assigned charge is not large compared to the cost of the optimal correlation clustering. We do so by observing that our algorithm $(1+\epsilon)$-approximates the covering LP of Figure 1, which we refer to as the *charging LP*.

Our faster deterministic non-combinatorial algorithm solves the charging LP using an LP solver tailored to covering LPs [4, 39]. An improved solver for covering LPs would directly improve the running time of this algorithm.

Figure 1 The primal and dual LP relaxations for Correlation Clustering, which we refer to as the *charging LP*. $T(G)$ is the set of all bad triplets in G.

$$\begin{aligned} \min \quad & \sum_{uv\in\binom{V}{2}} x_{uv} \\ \text{s.t.} \quad & x_{uv}+x_{vw}+x_{wu} \geq 1 \quad \forall uvw \in T(G), \\ & x_{uv} \geq 0 \quad \forall uv \in \binom{V}{2} \end{aligned}$$

$$\begin{aligned} \max \quad & \sum_{uvw\in T(G)} y_{uvw} \\ \text{s.t.} \quad & \sum_{w:uvw\in T(G)} y_{uvw} \leq 1 \quad \forall uv \in \binom{V}{2}, \\ & y_{uvw} \geq 0 \quad \forall uvw \in T(G) \end{aligned}$$

Lower Bound. Our lower bound is obtained by taking a complete graph K_n for some even number of vertices n, and removing a perfect matching. Each vertex in the resulting graph is adjacent to all but one other vertex and so *any* PIVOT algorithm will partition the vertices into a large cluster of $n-1$ vertices and a singleton cluster. A non PIVOT algorithm, however, is free to create just a single cluster of size n, at much lower cost. The ratio between these solutions tends to 3 with increasing n.

We note that in [3] the authors proved that CC-PIVOT's analysis is tight. That is, its expected approximation factor is not better than 3. However, their lower bound construction (a complete graph K_n minus one edge) only works for CC-PIVOT, not for PIVOT algorithms in general.

1.4 Open Problems

We finally raise some open questions.

1. Can we improve the approximation factor of Constrained Correlation Clustering from 16 to 3 while keeping the running time at $\widetilde{O}(n^3)$?
2. We measure the performance of a PIVOT algorithm by comparing it to the best correlation clustering obtained by *any* algorithm. But as Theorem 3 proves, there is no PIVOT algorithm with an approximation factor better than 3. If we instead compare the output to the best correlation clustering obtained by a *PIVOT algorithm*, can we get better guarantees (perhaps even an exact algorithm in polynomial time)?
3. In the Node-Weighted Correlation Clustering problem, we studied the natural objective of minimizing the total cost $\omega_v \cdot \omega_u$ of all violated preferences uv. Are there specific applications of this problem? Can we achieve similar for other cost functions such as $\omega_v + \omega_u$?

2 Preliminaries

We denote the set $\{1, \dots, n\}$ by $[n]$. We denote all subsets of size k of a set A by $\binom{A}{k}$. The symmetric difference between two sets A, B is denoted by $A \triangle B$. We write $\operatorname{poly}(n) = n^{O(1)}$ and $\widetilde{O}(n) = n(\log n)^{O(1)}$.

In this paper all graphs $G = (V, E)$ are undirected and unweighted. We typically set $n = |V|$ and $m = |E|$. For two disjoint subsets $U_1, U_2 \subseteq V$, we denote the set of edges with one endpoint in U_1 and the other in U_2 by $E(U_1, U_2)$. The subgraph of G induced by vertex-set U_1 is denoted by $G[U_1]$. For vertices u, v, w we often abbreviate the (unordered) set $\{ u, v \}$ by uv and similarly write uvw for $\{ u, v, w \}$. We say that uvw is a *bad triplet* in G if the induced subgraph $G[uvw]$ contains exactly two edges (i.e., is isomorphic to $K_{1,2}$). Let $T(G)$ denote the set of bad triplets in G. We say that the edge set E_C of a clustering $C = \{ C_1, \dots, C_k \}$ of V is the set of pairs with both endpoints in the same set in C. More formally, $E_C = \bigcup_{i=1}^{k} \binom{C_i}{2}$.

We now formally define the problems of interest.

▶ **Definition 6** (Correlation Clustering). *Given a graph $G = (V, E)$, output a clustering $C = \{ C_1, \dots, C_k \}$ of V with edge set E_C minimizing $|E \triangle E_C|$.*

An algorithm for Correlation Clustering is said to be a PIVOT algorithm if it is an instantiation of Algorithm 1 (Page 3). That is, an algorithm which, based on some criterion, picks an unclustered node u (the *pivot*), creates a cluster containing u and its unclustered neighbors in (V, E), and repeats the process until all nodes are clustered. In particular, the algorithm may not modify the graph in other ways before choosing a pivot.

The constrained version of Correlation Clustering is defined as follows.

▶ **Definition 7** (Constrained Correlation Clustering). *Given a graph $G = (V, E)$, a set of friendly pairs $F \subseteq \binom{V}{2}$ and a set of hostile pairs $H \subseteq \binom{V}{2}$, compute a clustering $C = \{ C_1, \dots, C_k \}$ of V with edge set E_C such that no pair $uv \in F$ has u, v in different clusters and no pair $uv \in H$ has u, v in the same cluster. The clustering C shall minimize $|E \triangle E_C|$.*

We also introduce Node-Weighted Correlation Clustering, a new related problem that may be of independent interest.

▶ **Definition 8** (Node-Weighted Correlation Clustering). *Given a graph $G = (V, E)$ and positive weights $\{ \omega_u \}_{u \in V}$ on the nodes, compute a clustering $C = \{ C_1, \dots, C_k \}$ of V with edge set E_C minimizing*

$$\sum_{uv \in E \triangle E_C} \omega_u \cdot \omega_v \, .$$

For simplicity, we assume that the weights are bounded by $\operatorname{poly}(n)$, and thereby fit into a constant number of word RAM cells of size $w = \Theta(\log n)$. We remark that our randomized algorithm would be a polynomial (but not linear) time one if we allowed the weights to be of exponential size.

The Node-Weighted Correlation Clustering problem clearly generalizes Correlation Clustering since we pay $w(u) \cdot w(v)$ (instead of 1) for each pair uv violating a preference.

3 Combinatorial Algorithms for Constrained Correlation Clustering

Let us fix the following notation: A connected component in (V, F) is a *supernode*. The set of supernodes partitions V and is denoted by SN. Given a node u, we let $s(u)$ be the unique supernode containing u. Two supernodes U, W are *hostile* if there exists a hostile pair uw with $u \in U, w \in W$. Two supernodes U, W are *connected* if $|E(U, W)| \geq 1$. Two supernodes U, W are *β-connected* if $|E(U, W)| \geq \beta \cdot |U| \cdot |W|$.

The first step of our combinatorial approach is to transform the graph G into a more manageable form G', see procedure Transform of Algorithm 2. The high-level idea is that in G':

1. If uv is a friendly pair, then u and v are connected and have the same neighborhood.
2. If uv is a hostile pair, then u and v are not connected and have no common neighbor.
3. An $O(1)$-approximation of the G' instance is also an $O(1)$-approximation of the G instance.

As was already noticed in [34], Properties 1 and 2 imply that a PIVOT algorithm on G' gives a clustering satisfying the hard constraints. Along with Property 3 and our deterministic combinatorial PIVOT algorithm for Correlation Clustering in Theorem 2, we prove Theorem 1. Properties 1 and 2 (related to correctness) and the running time ($\widetilde{O}(n^3)$) of our algorithm are relatively straightforward to prove. Due to space constraints, their proofs can be found in the full version of the paper [26]. In this section we instead focus on the most technically challenging part, the approximation guarantee.

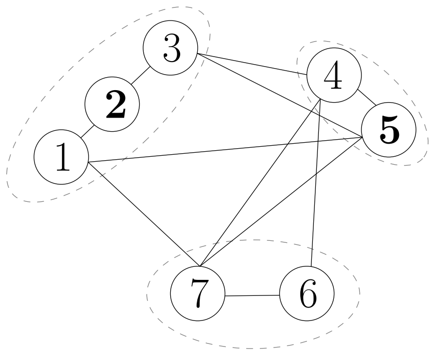
Our algorithm works as follows (see also Figure 2): If some supernode is hostile to itself, then it outputs that no clustering satisfies the hard constraints. Else, starting from the edge set E, it adds all edges within each supernode. Then it drops all edges between hostile supernodes. Subsequently, it repeatedly detects hostile supernodes that are connected with the same supernode, and drops one edge from each such connection. Finally, for each β-connected pair of supernodes, it connects all their nodes if $\beta > \frac{3-\sqrt{5}}{2}$, and disconnects them otherwise[4].

From a high-level view, the first two modifications are directly related to the hard constraints: If u_1, u_2 are friendly and u_2, u_3 are friendly, then any valid clustering has u_1, u_3 in the same cluster, even if a preference discourages it. Similarly, if u_1, u_2 are friendly, u_3, u_4 are friendly, but u_1, u_3 are hostile, then any valid clustering has u_2, u_4 in different clusters, even if a preference discourages it. Our first two modifications simply make the preferences consistent with the hard constraints.

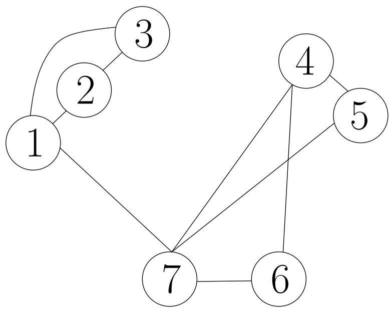
The third modification guarantees that hostile supernodes share no common neighbor. A PIVOT algorithm will thus never put their nodes in the same cluster, as the hostility constraints require. Concerning the cost, notice that if hostile supernodes U_1, U_2 are connected with supernode U_3, then no valid clustering can put all three of them in the same cluster. Therefore we always need to pay either for the connections between U_1 and U_3, or for the connections between U_2 and U_3.

Finally, after the rounding step, for each pair of supernodes U_1, U_2, the edge set $E(U_1, U_2)$ is either empty or the full set of size $|U_1| \cdot |U_2|$. This ensures that a PIVOT algorithm always puts all nodes of a supernode in the same cluster, thus also obeying the friendliness constraints. Concerning the cost of the rounded instance, a case analysis shows that it is always within a constant factor of the cost of the instance before rounding.

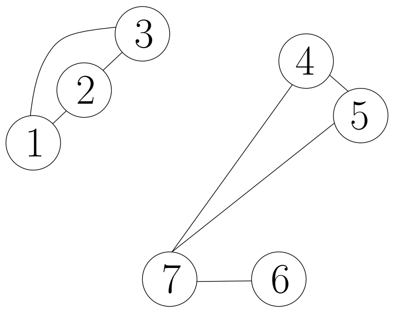
[4] The constant $\frac{3-\sqrt{5}}{2}$ optimizes the approximation factor. The natural choice of 0.5 would still give a constant approximation factor, albeit slightly worse.



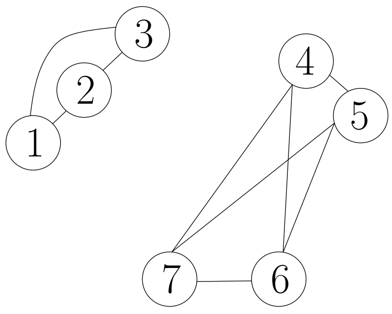
(a) The original graph. The set of friendly pairs is $F = \{\{1,2\},\{2,3\},\{4,5\},\{6,7\}\}$, and the only hostile pair in H is $\{2,5\}$.



(b) Line 3 introduces edge $\{1,3\}$, and Line 4 disconnects the supernodes containing 2 and 5.



(c) Line 6 removes the pair of edges $\{1,7\}$ and $\{4,6\}$ because $1,4$ are in hostile supernodes while $6,7$ are in the same supernode.



(d) Line 9 introduces all edges connecting supernodes $\{4,5\}$ and $\{6,7\}$ because there were enough edges between them already.

Figure 2 Illustrates an application of TRANSFORM(G,F,H) (Algorithm 2). In the transformed graph, for any two supernodes U_1, U_2, either all pairs with an endpoint in U_1 and an endpoint in U_2 share an edge, or none of them do. Furthermore, all pairs within a supernode are connected and no hostile supernodes are connected.

Formally, let E' be the edge set of the transformed graph G', let E_3 be the edge set at Line 8 of Algorithm 2 (exactly before the rounding step), OPT be the edge set of an optimal clustering for E satisfying the hard constraints described by F and H, OPT$'$ be the edge set of an optimal clustering for the preferences defined by E', and E_C be the edge set of the clustering returned by our algorithm. Finally, let α be the approximation factor of the PIVOT algorithm used.

▶ **Lemma 9.** *Given an instance (V, E, F, H) of Constrained Correlation Clustering, if two nodes u_1, u_2 are in the same supernode, then they must be in the same cluster.*

Proof. The proof follows by "in the same cluster" being a transitive property.

More formally, u_1, u_2 are in the same connected component in (V, F), as $s(u_1) = s(u_2)$. Thus, there exists a path from u_1 to u_2. We claim that all nodes in a path must be in the same cluster. This is trivial if the path is of length 0 ($u_1 = u_2$) or of length 1 ($u_1u_2 \in F$). Else, the path is $u_1, w_1, \ldots, w_k, u_2$ for some $k \geq 1$. We inductively have that all of $w_1, \ldots, w_k, u_2$ must be in the same cluster, and u_1 must be in the same cluster with w_1 because $u_1w_1 \in F$. Therefore, all nodes in the path must be in the same cluster with w_1. ◀

We now show that it is enough to bound the symmetric difference between E and E'.

Algorithm 2 The procedure ConstrainedCluster is given a graph $G = (V, E)$ describing the preferences, a set of friendly pairs F and a set of hostile pairs H. It creates a new graph G' using the procedure Transform and uses any PIVOT algorithm on G' to return a clustering.

```
procedure Transform($G = (V, E), F, H$)
1   Compute the connected components of $(V, F)$
    // Impossible iff some pair must both be and not be in the same
       cluster.
2   if $\exists U \in SN$ hostile to itself then return $G' = (\emptyset, \emptyset)$
    // Connect nodes in the same supernode.
3   $E_1 \leftarrow E \cup \{ uv \in \binom{V}{2} \mid s(u) = s(v) \}$
    // Disconnect pairs in hostile supernodes.
4   $E_2 \leftarrow E_1 \setminus \{ uv \in \binom{V}{2} \mid s(u)$ and $s(v)$ are hostile$\}$
    // While hostile supernodes $U_1, U_2$ are both connected with super-
    // node $U_3$, drop an edge between $U_1, U_3$ and an edge between $U_2, U_3$
5   $E_3 \leftarrow E_2$
6   while $\exists U_1, U_2, U_3 \in \binom{SN}{3}$ such that $U_1, U_2$ are hostile and
     $\exists u_1 \in U_1, u_2 \in U_2, u_3 \in U_3, u_3' \in U_3$ such that $u_1u_3 \in E_3, u_2u_3' \in E_3$ do
7       $E_3 \leftarrow E_3 \setminus \{ u_1u_3, u_2u_3' \}$
    // Round connections between pairs of supernodes
8   $E_4 \leftarrow E_3$
9   foreach $\{U_1, U_2\} \in \binom{SN}{2}$ do
10      $E_{U_1,U_2} \leftarrow \{u_1u_2 \mid u_1 \in U_1, u_2 \in U_2\}$
11      if $|E_{U_1,U_2} \cap E_4| > \frac{3-\sqrt{5}}{2}|U_1| \cdot |U_2|$ then $E_4 \leftarrow E_4 \cup E_{U_1,U_2}$
12      else $E_4 \leftarrow E_4 \setminus E_{U_1,U_2}$
13  return $G' = (V, E_4)$

procedure ConstrainedCluster($G = (V, E), F, H$)
14  $G' \leftarrow$ Transform($G=(V,E)$, $F$, $H$)
15  if $G' = (\emptyset, \emptyset)$ then return "Impossible"
16  return PIVOT($G'$)
```

▶ **Lemma 10.** *The cost of our clustering C is $|E \triangle E_C| \leq (\alpha + 1)|E \triangle E'| + \alpha|E \triangle \mathrm{OPT}|$.*

Proof. The symmetric difference of sets satisfies the triangle inequality; we therefore have

$$|E \triangle E_C| \leq |E \triangle E'| + |E' \triangle E_C|.$$

C is an α-approximation for $G' = (V, E')$ and thus $|E' \triangle E_C| \leq \alpha|E' \triangle \mathrm{OPT}'| \leq \alpha|E' \triangle \mathrm{OPT}|$. Therefore:

$$|E \triangle E_C| \leq |E \triangle E'| + \alpha|E' \triangle \mathrm{OPT}| \leq |E \triangle E'| + \alpha|E' \triangle E| + \alpha|E \triangle \mathrm{OPT}|.$$

with the second inequality following by applying the triangle inequality again. ◀

In order to upper bound $|E \triangle E'|$ by the cost of the optimal clustering $|E \triangle \mathrm{OPT}|$, we first need to lower bound the cost of the optimal clustering.

▶ **Lemma 11.** *Let S be the set of all pairs of distinct supernodes U, W that are in the same cluster in* OPT*. Then* $|E \triangle \mathrm{OPT}| \geq \sum_{\{U,W\} \in S} |E(U,W) \triangle E_3(U,W)|$.

Proof. The high-level idea is that when a node is connected to two hostile nodes, then any valid clustering needs to pay for at least one of these edges. Extending this fact to supernodes, we construct an edge set of size $\sum_{\{U,W\} \in S} |E(U,W) \triangle E_3(U,W)|$ such that the optimal clustering needs to pay for each edge in this set.

First, for any $\{U, W\} \in S$ it holds that $E(U,W) \triangle E_3(U,W) = E(U,W) \setminus E_3(U,W)$ because Line 3 (Algorithm 2) does not modify edges between pairs of distinct supernodes, and Lines 4 and 6 only remove edges.

Each edge of $E(U,W) \setminus E_3(U,W)$ is the result of applying Line 6, seeing as Line 4 only removes edges from hostile pairs of supernodes. Thus each edge $uw \in E(U,W) \setminus E_3(U,W)$ can be paired up with a unique edge $xy \in E$ which is removed together with uw. Without loss of generality it holds that $x \in U, y \in Z$ for some supernode Z different from U and W. Due to the way Line 6 chooses edges it must be the case that Z and W are hostile, hence $xy \in E \triangle \mathrm{OPT}$.

Summing over all pairs of clustered supernodes gives the result stated in the lemma. ◀

We are now ready to bound $|E \triangle E'|$.

▶ **Lemma 12.** $|E \triangle E'| \leq (1 + \sqrt{5})|E \triangle \mathrm{OPT}|$

Proof. To prove this, we first charge each pair of nodes in a way such that the total charge is at most $2|E \triangle \mathrm{OPT}|$. Then we partition the pairs of nodes into 5 different sets, and show that the size of the intersection between $E \triangle E'$ and each of the 5 sets is at most $\frac{1+\sqrt{5}}{2}$ times the total charge given to the pairs in the given set.

The first three sets contain the pairs across non-hostile supernodes; out of them the first one is the most technically challenging, requiring a combination of Lemma 11 (related to Line 6 of Algorithm 2) and a direct analysis on $E \triangle \mathrm{OPT}$, as neither of them would suffice on their own. The analysis of the second and third sets relate to the rounding in Line 9. The fourth set contains pairs across hostile supernodes, while the fifth set contains pairs within supernodes. Their analysis is directly based on the hard constraints.

Let us define our charging scheme: first, each pair of nodes is charged if the optimal clustering pays for it, i.e. if this pair is in $E \triangle \mathrm{OPT}$. We further put a charge on the pairs $uw \in E \triangle E_3$ which connect supernodes that are clustered together in OPT. Notice that the number of such edges is a lower bound on $|E \triangle \mathrm{OPT}\,|$ by Lemma 11. Therefore the total charge over all pairs of nodes is at most $2|E \triangle \mathrm{OPT}\,|$ and no pair is charged twice.

Case 1. Consider two distinct supernodes U, W that are not hostile, which have more than $\frac{3-\sqrt{5}}{2}|U|\cdot|W|$ edges between them in E, and have at most $\frac{3-\sqrt{5}}{2}|U|\cdot|W|$ edges in E_3. Then the rounding of Line 9 removes all edges between them. Therefore $|E(U,W) \triangle E'(U,W)| = |E(U,W)| \leq |U| \cdot |W|$. If OPT separates U and W, then the pairs are charged $|E(U,W)|$; else they are charged $|U| \cdot |W| - |E(U,W)|$ due to the part of the charging scheme related to $E \triangle \mathrm{OPT}$. In the latter case, they are also charged $|E(U,W)| - |E_3(U,W)|$ due to the part of the charging scheme related to Lemma 11. Therefore they are charged at least

$$\begin{aligned} |U| \cdot |W| - |E(U,W)| + |E(U,W)| - |E_3(U,W)| &= |U| \cdot |W| - |E_3(U,W)| \\ &\geq |U| \cdot |W| - \tfrac{3-\sqrt{5}}{2}|U| \cdot |W|. \end{aligned}$$

Thus, in the worst case, these pairs contribute

$$\max\left\{ \frac{|E(U,W)|}{|E(U,W)|}, \frac{|E(U,W)|}{|U| \cdot |W| - \frac{3-\sqrt{5}}{2}|U| \cdot |W|} \right\} \leq \frac{1}{1 - \frac{3-\sqrt{5}}{2}} = \frac{1+\sqrt{5}}{2}$$

times more in $|E \triangle E'|$ compared to their charge.

Case 2. Consider two distinct supernodes U, W that are not hostile, which have more than $\frac{3-\sqrt{5}}{2}|U| \cdot |W|$ edges between them in E, and more than $\frac{3-\sqrt{5}}{2}|U| \cdot |W|$ edges in E_3. Then the rounding of Line 9 will include all $|U| \cdot |W|$ edges between them. Thus we have $|E(U,W) \triangle E'(U,W)| = |U| \cdot |W| - |E(U,W)| < (1 - \frac{3-\sqrt{5}}{2})|U| \cdot |W|$. If OPT separates U and W it pays for $|E(U,W)| > \frac{3-\sqrt{5}}{2}|U| \cdot |W|$ pairs. Otherwise it pays $|U| \cdot |W| - |E(U,W)|$. Thus, in the worst case, these pairs contribute $\frac{1-\frac{3-\sqrt{5}}{2}}{\frac{3-\sqrt{5}}{2}} = \frac{1+\sqrt{5}}{2}$ times more in $|E \triangle E'|$ compared to their charge.

Case 3. If two distinct supernodes U, W are not hostile and have at most $\frac{3-\sqrt{5}}{2}|U| \cdot |W|$ edges between them in E, then they also have at most that many edges in E_3 as we only remove edges between such supernodes. There are thus no edges between them in E', meaning that $|E(U,W) \triangle E'(U,W)| = |E(U,W)| \le \frac{3-\sqrt{5}}{2}|U| \cdot |W|$. If OPT separates U, W it pays for $|E(U,W)|$ pairs related to the connection between U, W; else it pays for $|U| \cdot |W| - |E(U,W)| \ge (1 - \frac{3-\sqrt{5}}{2})|U| \cdot |W| > \frac{3-\sqrt{5}}{2}|U| \cdot |W|$. Thus these pairs' contribution in $|E \triangle E'|$ is at most as much as their charge.

Case 4. Pairs uv with $s(u) \ne s(v)$ and $s(u)$ hostile with $s(v)$ are not present in E'. That is because by Line 4 no pair of hostile supernodes is connected; then Line 6 only removes edges, and Line 9 does not add any edge between $s(u)$ and $s(v)$ as they had $0 \le \frac{3-\sqrt{5}}{2}|s(u)| \cdot |s(v)|$ edges between them. The edge uv is also not present in OPT as $s(u)$ and $s(v)$ are not in the same cluster because they are hostile. These pairs' contribution in $|E \triangle E'|$ is exactly equal to their charge.

Case 5. Pairs uv with $s(u) = s(v)$ are present in E' by Line 3 and the fact that all subsequent steps only modify edges whose endpoints are in different supernodes. The pair uv is also present in OPT, by Lemma 9. Therefore these pairs' contribution in $|E \triangle E'|$ is exactly equal to their charge.

In the worst case, the pairs of each of the five sets contribute at most $\frac{1+\sqrt{5}}{2}$ times more in $|E \triangle E'|$ compared to their charge, which proves our lemma. ◀

We are now ready to prove the main theorem.

Proof of Theorem 1. In Theorem 2 we established that there is a deterministic combinatorial PIVOT algorithm computing a Correlation Clustering with approximation factor $\alpha = 3 + \epsilon$ in time $\widetilde{O}(n^3)$, for any constant $\epsilon > 0$. Using this algorithm in Algorithm 2 gives a valid clustering. By Lemmas 10 and 12, its approximation factor is bounded by $(\alpha+1)\cdot(1+\sqrt{5})+\alpha$. This is less than 16 for $\epsilon = 0.01$. ◀

4 PIVOT Algorithms for Correlation Clustering

4.1 Lower Bound

First we prove Theorem 3 which states that there is no PIVOT algorithm for Correlation Clustering with approximation factor better than 3.

Proof of Theorem 3. Let $G = ([2n], E)$ for some integer n, where the edge set E contains all pairs of nodes except for pairs of the form $(2k+1, 2k+2)$. In other words, the edge set of G contains all edges except for a perfect matching.

Note that if we create a single cluster containing all nodes, then the cost is exactly n. On the other hand, let u be the first choice that a PIVOT algorithm makes. If u is even, let $v = u - 1$, otherwise let $v = u + 1$. By definition of G, v is the only node not adjacent to u. Therefore, the algorithm creates two clusters – one containing all nodes except for v, and one containing only v. There are $2n-2$ edges across the two clusters, and $n-1$ missing edges in the big cluster, meaning that the cost is $3n-3$.

Therefore, the approximation factor of any PIVOT algorithm is at least $(3n-3)/n = 3-\frac{3}{n}$. This proves the theorem, as for any constant less than 3, there exists a sufficiently large n such that $3-\frac{3}{n}$ is larger than that constant. ◀

4.2 Optimal Deterministic PIVOT: 3-Approximation

A *covering* LP is a linear program of the form $\min_x\{\, cx \mid Ax \geq b \,\}$ where A, b, c, and x are restricted to vectors and matrices of non-negative entries. Covering LPs can be solved more efficiently than LPs in general and we rely on the following known machinery to prove Theorem 2:

▶ **Theorem 13** (Covering LPs, Combinatorial [29, 27]). *Any covering LP with at most N nonzero entries in the constraint matrix can be $(1+\epsilon)$-approximated by a combinatorial algorithm in time $\widetilde{O}(N\epsilon^{-3})$.*[5]

▶ **Theorem 14** (Covering LPs, Non-Combinatorial [4, 39]). *Any covering LP with at most N nonzero entries in the constraint matrix can be $(1+\epsilon)$-approximated in time $\widetilde{O}(N\epsilon^{-1})$.*

Of the two theorems, the time complexity of the algorithm promised by Theorem 14 is obviously better. However, the algorithm of Theorem 13 is remarkably simple in our setting and could thus prove to be faster in practice. Note that either theorem suffices to obtain a $(3+\epsilon)$-approximation for Correlation Clustering in $\widetilde{O}(n^3)$ time, for constant $\epsilon > 0$.

For completeness, and in order to demonstrate how simple the algorithm from Theorem 13 is in our setting, we include the pseudocode as Algorithm 3. In [26] we formally prove that Algorithm 3 indeed has the properties promised by Theorem 13.

Algorithm 3 The combinatorial algorithm to $(1+O(\epsilon))$-approximate the LP in Figure 1 (Page 7) using the multiplicative weights update method. The general method was given by Garg and Könemann [29] and later refined by Fleischer [27]. We here use the notation $m(x) = \min_{uvw \in T(G)} x_{uv} + x_{vw} + x_{wu}$.

```
procedure CHARGE($G = (V, E)$)
1   Initialize $x_{uv}$, $x^*_{uv} \leftarrow 1$ for all $uv \in \binom{V}{2}$
2   while $\sum_{uv} x_{uv} < B := (\binom{n}{2}(1+\epsilon))^{1/\epsilon} / (1+\epsilon)$ do
3       Find a bad triplet $uvw$ minimizing $x_{uv} + x_{vw} + x_{wu}$
4       $x_{uv} \leftarrow (1+\epsilon) \cdot x_{uv}$
5       $x_{vw} \leftarrow (1+\epsilon) \cdot x_{vw}$
6       $x_{wu} \leftarrow (1+\epsilon) \cdot x_{wu}$
7       if $(\sum_{uv} x_{uv}) / m(x) < (\sum_{uv} x^*_{uv}) / m(x^*)$ then
8           foreach $uv \in \binom{V}{2}$ do $x^*_{uv} \leftarrow x_{uv}$
9   return $\{\, x^*_{uv} / m(x^*) \,\}_{uv}$
```

[5] The running time we state seems worse by a factor of ϵ^{-1} as compared to the theorems in [29, 27]. This is because the authors assume access to a machine model with exact arithmetic of numbers of size exponential in ϵ^{-1}. We can simulate this model using fixed-point arithmetic with a running time overhead of $\widetilde{O}(\epsilon^{-1})$.

The solution found by Algorithm 3 is used together with the framework by van Zuylen and Williamson [34], see CLUSTER in Algorithm 4. CLUSTER is discussed further in the full version [26], where the following lemmas are proven.

Algorithm 4 The PIVOT algorithm by van Zuylen and Williamson [34]. Given a graph G and a good charging $\{ x_{uv} \}_{uv}$ (in the sense of Lemma 15), it computes a correlation clustering.

```
  procedure CLUSTER($G = (V, E), x = \{ x_{uv} \}_{uv \in \binom{V}{2}}$)
1     $C \leftarrow \emptyset$
2     while $V \neq \emptyset$ do
3         Pick a pivot node $u \in V$ minimizing
              $$\frac{\sum_{vw:uvw \in T(G)} 1}{\sum_{vw:uvw \in T(G)} x_{vw}}$$
4         Add a cluster containing $u$ and all its neighbors to $C$
5         Remove $u$, its neighbors and all their incident edges from $G$
6     return $C$
```

▶ **Lemma 15** (Correctness of CLUSTER). *Assume that $x = \{ x_{uv} \}_{uv}$ is a feasible solution to the LP in Figure 1. Then* CLUSTER(G, x) *computes a correlation clustering of cost $3 \sum_{uv} x_{uv}$. In particular, if x is an α-approximate solution to the LP (for some $\alpha \geq 1$), then* CLUSTER(G, x) *returns a 3α-approximate correlation clustering.*

▶ **Lemma 16** (Running Time of CLUSTER, [34]). CLUSTER(G, x) *runs in time $O(n^3)$.*

Given Theorems 13 and 14 we quickly prove Theorem 2.

Proof of Theorem 2. We compute a $(1 + \epsilon/3)$-approximate solution x of the charging LP using Theorem 13 (that is, using the procedure CHARGE(G)). Plugging this solution x into CLUSTER(G, x) returns a $(3 + \epsilon)$-approximate correlation clustering by Lemma 15. The total running time is bounded by $O(n^3)$ by Lemma 16 plus $\widetilde{O}(n^3\epsilon^{-3})$ by Theorem 13 (note that there are n^3 constraints, each affecting only a constant number of variables, hence the number of nonzeros in the constraint matrix is $N \leq O(n^3)$). For constant $\epsilon > 0$, this becomes $\widetilde{O}(n^3)$.

To obtain a 3-approximation, we observe that any correlation clustering has cost less than $\binom{n}{2}$. Hence, we can run the previous algorithm with $\epsilon = 1/\binom{n}{2}$ and the $(3 + \epsilon)$-approximate solution is guaranteed to also be 3-approximate. The running time would be bounded by $\widetilde{O}(n^9)$. To improve upon this, we use the covering LP solver in Theorem 14 which runs in time $\widetilde{O}(n^3\epsilon^{-1})$. By again setting $\epsilon = 1/\binom{n}{2}$, the running time becomes $\widetilde{O}(n^5)$. ◀

References

1 Rakesh Agrawal, Alan Halverson, Krishnaram Kenthapadi, Nina Mishra, and Panayiotis Tsaparas. Generating labels from clicks. In Ricardo Baeza-Yates, Paolo Boldi, Berthier A. Ribeiro-Neto, and Berkant Barla Cambazoglu, editors, *Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009*, pages 172–181. ACM, 2009. doi:10.1145/1498759.1498824.

2 Nir Ailon and Moses Charikar. Fitting tree metrics: Hierarchical clustering and phylogeny. *SIAM J. Comput.*, 40(5):1275–1291, 2011. Announced at FOCS'05. doi:10.1137/100806886.

3 Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):23:1–23:27, 2008. Announced in STOC 2005. doi:10.1145/1411509.1411513.

4 Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly linear-time packing and covering LP solvers – achieving width-independence and -convergence. *Math. Program.*, 175(1-2):307–353, 2019. doi:10.1007/s10107-018-1244-x.

5 Arvind Arasu, Christopher Ré, and Dan Suciu. Large-scale deduplication with constraints using dedupalog. In Yannis E. Ioannidis, Dik Lun Lee, and Raymond T. Ng, editors, *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pages 952–963. IEEE Computer Society, 2009. doi:10.1109/ICDE.2009.43.

6 Sepehr Assadi and Chen Wang. Sublinear time and space algorithms for correlation clustering via sparse-dense decompositions. *CoRR*, abs/2109.14528, 2021. arXiv:2109.14528.

7 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Mach. Learn.*, 56(1-3):89–113, 2004. doi:10.1023/B:MACH.0000033116.57574.95.

8 Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Almost 3-approximate correlation clustering in constant rounds. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 720–731. IEEE, 2022. doi:10.1109/FOCS54457.2022.00074.

9 Soheil Behnezhad, Moses Charikar, Weiyun Ma, and Li-Yang Tan. Single-pass streaming algorithms for correlation clustering. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 819–849. SIAM, 2023. doi:10.1137/1.9781611977554.CH33.

10 Francesco Bonchi, Aristides Gionis, and Antti Ukkonen. Overlapping correlation clustering. *Knowl. Inf. Syst.*, 35(1):1–32, 2013. doi:10.1007/s10115-012-0522-9.

11 Mark Bun, Marek Eliáš, and Janardhan Kulkarni. Differentially private correlation clustering. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 1136–1146. PMLR, 2021. URL: http://proceedings.mlr.press/v139/bun21a.html.

12 Melanie Cambus, Fabian Kuhn, Etna Lindy, Shreyas Pai, and Jara Uitto. *A (3+ε)-Approximate Correlation Clustering Algorithm in Dynamic Streams*, pages 2861–2880. SIAM, 2024. doi:10.1137/1.9781611977912.101.

13 Nairen Cao, Vincent Cohen-Addad, Euiwoong Lee, Shi Li, Alantha Newman, and Lukas Vogl. Understanding the cluster linear program for correlation clustering. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1605–1616. ACM, 2024. doi:10.1145/3618260.3649749.

14 Nairen Cao, Shang-En Huang, and Hsin-Hao SU. *Breaking 3-Factor Approximation for Correlation Clustering in Polylogarithmic Rounds*, pages 4124–4154. SIAM, 2024. doi:10.1137/1.9781611977912.143.

15 Deepayan Chakrabarti, Ravi Kumar, and Kunal Punera. A graph-theoretic approach to webpage segmentation. In Jinpeng Huai, Robin Chen, Hsiao-Wuen Hon, Yunhao Liu, Wei-Ying Ma, Andrew Tomkins, and Xiaodong Zhang, editors, *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*, pages 377–386. ACM, 2008. doi:10.1145/1367497.1367549.

16 Sayak Chakrabarty and Konstantin Makarychev. Single-pass pivot algorithm for correlation clustering. keep it simple! *CoRR*, abs/2305.13560, 2023. doi:10.48550/arXiv.2305.13560.

17 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005. Announced in FOCS 2003. doi:10.1016/j.jcss.2004.10.012.

18 Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 219–228. ACM, 2015. doi:10.1145/2746539.2746604.

19 Yudong Chen, Sujay Sanghavi, and Huan Xu. Clustering sparse graphs. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 2213–2221, 2012. URL: https://proceedings.neurips.cc/paper/2012/hash/1e6e0a04d20f50967c64dac2d639a577-Abstract.html.

20 Vincent Cohen-Addad, Chenglin Fan, Euiwoong Lee, and Arnaud de Mesmay. Fitting metrics and ultrametrics with minimum disagreements. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 301–311. IEEE, 2022. doi:10.1109/FOCS54457.2022.00035.

21 Vincent Cohen-Addad, Silvio Lattanzi, Slobodan Mitrovic, Ashkan Norouzi-Fard, Nikos Parotsidis, and Jakub Tarnawski. Correlation clustering in constant many parallel rounds. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2069–2078. PMLR, 2021. URL: http://proceedings.mlr.press/v139/cohen-addad21b.html.

22 Vincent Cohen-Addad, Euiwoong Lee, Shi Li, and Alantha Newman. Handling correlated rounding error via preclustering: A 1.73-approximation for correlation clustering. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1082–1104. IEEE, 2023. doi:10.1109/FOCS57990.2023.00065.

23 Vincent Cohen-Addad, Euiwoong Lee, and Alantha Newman. Correlation clustering with sherali-adams. *CoRR*, abs/2207.10889, 2022. doi:10.48550/arXiv.2207.10889.

24 Vincent Cohen-Addad, David Rasmussen Lolck, Marcin Pilipczuk, Mikkel Thorup, Shuyi Yan, and Hanwen Zhang. Combinatorial correlation clustering. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1617–1628. ACM, 2024. doi:10.1145/3618260.3649712.

25 Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci.*, 361(2-3):172–187, 2006. doi:10.1016/j.tcs.2006.05.008.

26 Nick Fischer, Evangelos Kipouridis, Jonas Klausen, and Mikkel Thorup. A faster algorithm for constrained correlation clustering, 2025. arXiv:2501.03154.

27 Lisa Fleischer. A fast approximation scheme for fractional covering problems with variable upper bounds. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 1001–1010. SIAM, 2004. URL: http://dl.acm.org/citation.cfm?id=982792.982942.

28 Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michal Pilipczuk, and Yngve Villanger. Tight bounds for parameterized complexity of cluster editing with a small number of clusters. *J. Comput. Syst. Sci.*, 80(7):1430–1447, 2014. doi:10.1016/j.jcss.2014.04.015.

29 Naveen Garg and Jochen Koenemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, FOCS '98, page 300, USA, 1998. IEEE Computer Society.

30 Dmitri V. Kalashnikov, Zhaoqi Chen, Sharad Mehrotra, and Rabia Nuray-Turan. Web people search via connection analysis. *IEEE Trans. Knowl. Data Eng.*, 20(11):1550–1565, 2008. doi:10.1109/TKDE.2008.78.

31 Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang Dong Yoo. Higher-order correlation clustering for image segmentation. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain*, pages 1530–1538, 2011. URL: `https://proceedings.neurips.cc/paper/2011/hash/98d6f58ab0dafbb86b083a001561bb34-Abstract.html`.

32 Domenico Mandaglio, Andrea Tagarelli, and Francesco Gullo. Correlation clustering with global weight bounds. In Nuria Oliver, Fernando Pérez-Cruz, Stefan Kramer, Jesse Read, and José Antonio Lozano, editors, *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2021, Bilbao, Spain, September 13-17, 2021, Proceedings, Part II*, volume 12976 of *Lecture Notes in Computer Science*, pages 499–515. Springer, 2021. `doi:10.1007/978-3-030-86520-7_31`.

33 Gregory J. Puleo and Olgica Milenkovic. Correlation clustering with constrained cluster sizes and extended weights bounds. *SIAM J. Optim.*, 25(3):1857–1872, 2015. `doi:10.1137/140994198`.

34 Anke van Zuylen and David P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. *Math. Oper. Res.*, 34(3):594–620, 2009. Announced in SODA 2007. `doi:10.1287/moor.1090.0385`.

35 Nate Veldt. Correlation clustering via strong triadic closure labeling: Fast approximation algorithms and practical lower bounds. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 22060–22083. PMLR, 2022. URL: `https://proceedings.mlr.press/v162/veldt22a.html`.

36 Nate Veldt, David F. Gleich, and Anthony Wirth. A correlation clustering framework for community detection. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 439–448. ACM, 2018. `doi:10.1145/3178876.3186110`.

37 Michael D. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Trans. Software Eng.*, 17(9):972–975, 1991. `doi:10.1109/32.92917`.

38 Alastair J. Walker. New fast method for generating discrete random numbers with arbitrary frequency distributions. *Electronics Letters*, 10:127–128(1), April 1974.

39 Di Wang, Satish Rao, and Michael W. Mahoney. Unified acceleration method for packing and covering problems via diameter reduction. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 50:1–50:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. `doi:10.4230/LIPIcs.ICALP.2016.50`.

40 Julian Yarkony, Alexander T. Ihler, and Charless C. Fowlkes. Fast planar correlation clustering for image segmentation. In Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012 – 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI*, volume 7577 of *Lecture Notes in Computer Science*, pages 568–581. Springer, 2012. `doi:10.1007/978-3-642-33783-3_41`.