Partial Optimality and Fast Lower Bounds for Weighted Correlation Clustering

Jan-Hendrik Lange 1 2 Andreas Karrenbauer 1 Bjoern Andres 1 3 4

Abstract

Weighted correlation clustering is hard to solve and hard to approximate for general graphs. Its applications in network analysis and computer vision call for efficient algorithms. To this end, we make three contributions: We establish partial optimality conditions that can be checked efficiently, and doing so recursively solves the problem for series-parallel graphs to optimality, in linear time. We exploit the packing dual of the problem to compute a heuristic, but non-trivial lower bound faster than that of a canonical linear program relaxation. We introduce a re-weighting with the dual solution by which efficient local search algorithms converge to better feasible solutions. The effectiveness of our methods is demonstrated empirically on a number of benchmark instances.

1. Introduction

This paper is about weighted correlation clustering (Bansal et al., 2004), a combinatorial optimization problem whose feasible solutions are all clusterings of a graph, and whose objective function is a sum of weights $w^0, w^1: E \rightarrow \mathbb{R}^+$ defined on the edges $E$ of the graph. The weight $w^0$ is added to the sum if the nodes $\{u, v\} = e \in E$ are in the same cluster, and the weight $w^1$ is added to the sum if these nodes are in distinct clusters. The problem consists in finding a clustering of minimum weight.

Weighted correlation clustering has found applications in the fields of network analysis (Cesa-Bianchi et al., 2012) and, more recently, computer vision (Kappes et al., 2011; Keuper et al., 2015; Insafutdinov et al., 2016; Beier et al., 2017; Tang et al., 2017), partly due to its key property that the number of clusters is not fixed, constrained or penalized in the problem statement but is instead defined by the (any) solution. Weighted correlation clustering in general graphs is hard to solve exactly and hard to approximate (Demaine et al., 2006). Remarkable progress has been made toward algorithms that find feasible solutions by approximations or heuristics (cf. Section 2). Yet, the computation of lower bounds remains challenging for large instances (Swoboda & Andres, 2017).

We make the following contributions: Firstly, in order to reduce instances in size, we establish partial optimality conditions on the graph and weights that can be checked combinatorially in polynomial time and determine the values of some variables in an optimal solution. By applying these conditions recursively, we reduce an instance in size without restricting the quality of solutions. For series-parallel graphs, our algorithm solves weighted correlation clustering exactly and in linear time, as we show. For general graphs, we demonstrate its effectiveness empirically.

Secondly, in order to compute lower bounds to the optimal objective value efficiently, we define an algorithm that outputs a heuristic solution to a packing problem that is the dual of a reformulation of weighted correlation clustering. Empirically, this algorithm is shown to exhibit a run-time/tightness trade-off that is different from both the cutting plane algorithm of Kappes et al. (2015) and the message passing algorithm of Swoboda & Andres (2017), both of which solve a canonical linear program relaxation of weighted correlation clustering.

Thirdly, toward the goal of obtaining primal feasible solutions, we define a transformation of the weights w.r.t. our heuristic solution to the dual problem. This transformation is again a heuristic and is motivated by complementary slackness. Empirically, local search algorithms are shown to find feasible solutions of lower original weight when applied to instances with transformed weights.

In the supplementary material, we provide additional results that were omitted from the main paper for the sake of space. Implementations of our algorithms are provided on GitHub.

2. Related Work

Weighted correlation clustering has a long history in the field of combinatorial optimization. Grötschel & Wakabayashi (1989) state an equivalent problem for complete graphs and
Partial Optimality and Fast Lower Bounds for Weighted Correlation Clustering

devise a branch-and-cut algorithm for solving this problem exactly. The polyhedral-cut geometry of its feasible set is studied by Grötschel & Wakabayashi (1990); Deza et al. (1990; 1992), in the case of general graphs by Chopra & Rao (1993); Chopra (1994) and, for a more general problem, by Hornák et al. (2017). For uniform absolute edge costs, Bansal et al. (2004) coined the name correlation clustering, established NP-hardness and the first approximation results. The connection between correlation clustering in general and by E V a graph. Let G be a graph.

3. Problem Formulations

3.1. Weighted Correlation Clustering

Weighted correlation clustering is a combinatorial optimization problem whose feasible solutions are all clusterings of a graph.

Let G = (V, E) be a simple graph. We call a partition Π of V a clustering if every S ∈ Π induces a connected subgraph (cluster) of G. For any clustering Π of G, we denote by E1Π the set of those edges whose nodes are in the same cluster, and by E0Π the (complementary) set of those edges whose nodes are in distinct clusters:

\[ E^0_{\Pi} = \{ uv \in E \mid \exists S \in \Pi : u \in S \text{ and } v \in S \}, \]
\[ E^1_{\Pi} = E \setminus E^0_{\Pi}. \]

The set of edges E1Π is known as the multicut of G that corresponds to the clustering Π.

Definition 1. For any graph G = (V, E) and any w0, w1 : E → R+0, the instance of weighted correlation clustering w.r.t. G, w0 and w1 is the optimization problem

\[ \min_\Pi \sum_{e \in E^0_{\Pi}} w^0_e + \sum_{e \in E^1_{\Pi}} w^1_e. \]

3.2. Minimum Cost Multicut

Weighted correlation clustering is commonly stated in the form of a binary program whose feasible solutions are the incidence vectors of the multicut of the graph. The incidence vector xΠ ∈ {0, 1}E corresponding to the multicut induced by Π is defined as

\[ x^\Pi_e = \begin{cases} 1 & \text{if } e \in E^1_{\Pi}, \\ 0 & \text{else}. \end{cases} \]

Definition 2. For any graph G = (V, E) and any c : E → R, the instance of the minimum cost multicut problem w.r.t. G and c is the binary program

\[ \min_\Pi \sum_{e \in E} c_e x^\Pi_e. \]

The minimizers of an instance of weighted correlation clustering (Def. 1) coincide with the minimizers of the instance of minimum cost multicut (Def. 2) with c = w1 − w0, since

\[ \min_\Pi \sum_{e \in E^0_{\Pi}} w^0_e + \sum_{e \in E^1_{\Pi}} w^1_e \]

\[ = \min_\Pi \sum_{e \in E} (w^0_e (1 - x^\Pi_e^1) + w^1_e x^\Pi_e^1) \]

\[ = \sum_{e \in E} w^0_e + \min_{\Pi : E^0_{\Pi} \text{ const.}} \sum_{e \in E} (w^1_e - w^0_e) x^\Pi_e. \]

3.3. Linear Program Relaxation

By taking the convex hull of multicut incidence vectors

\[ \text{MC}(G) := \text{conv}\{ x^\Pi \mid \Pi \text{ clustering of } G \}, \]

the minimum cost multicut problem (Def. 2) can be written as the integer linear programming problem

\[ \min_{x \in \text{MC}(G)} \sum_{e \in E} c_e x_e. \quad (P_{\text{MC}}) \]
The set $MC(G)$ is called multicut polytope of $G$ (Chopra & Rao, 1993). As the minimum cost multicut problem is NP-hard, a full description of the multicut polytope in terms of its facets is impractical. For practical purposes a linear programming (LP) relaxation of $P_{MC}$ is derived as follows.

Denote by $C(G)$ the set of all simple cycles of $G$. For any cycle $C \in C(G)$, we write $E_C$ for the edge set of $C$. It is straightforward to check the fact that any multicut incidence vector $x^\Pi$ satisfies the system of linear inequalities

$$\forall C \in C(G) \forall f \in E_C : \quad x_f \leq \sum_{e \in E_C \setminus \{f\}} x_e,$$

the so-called cycle inequalities (Chopra & Rao, 1993).

Therefore, the standard linear programming relaxation is given by the program

$$\min_{x \in CYC(G)} \sum_{e \in E} c_e x_e \quad (P_{CYC})$$

whose feasible set

$$CYC(G) := \{ x \in [0,1]^E \mid x \text{ satisfies (10)} \} \quad (11)$$

is also known as the cycle relaxation of $MC(G)$. The problem $P_{CYC}$ is practical, because the cycle inequalities in (10) can be separated in polynomial time. The lower bounds thus obtained can serve to solve (small) instances of the minimum cost multicut problem by branch-and-cut because the cycle relaxation has no integer vertices except the incidence vectors of multicuts, according to Lemma 1.

**Lemma 1** (Chopra & Rao (1993)). For any graph $G = (V,E)$, it holds that $MC(G) = CYC(G) \cap \mathbb{Z}^E$.

A reference algorithm that we use for the experiments in Section 7 further exploits the fact that a cycle inequality in (10) defines a facet of $MC(G)$ iff the associated cycle is chordless.

### 3.4. Cycle Covering Formulation

For the presentation of this paper, we employ an alternative (integer) linear programming formulation in terms of covering cycles, which was similarly considered, e.g., by Demaine et al. (2006) for the combinatorial problem and by Charikar et al. (2005) in connection with the LP relaxation for complete graphs. We rewrite the feasible set of the general LP relaxation relative to the cost vector $c$. Therefore, let $G$ and $c$ be fixed.

We call an edge $e \in E$ repulsive if $c_e < 0$ and we call it attractive if $c_e > 0$. Note that we may w.l.o.g. remove all edges $e \in E$ with $c_e = 0$, since the choice of $x_e$ is irrelevant to the objective. We write $E = E^+ \cup E^-$ where $E^+$, $E^-$ collect all attractive and repulsive edges, respectively.

We call a cycle of $G$ conflicted w.r.t. $(G, c)$ if it contains precisely one repulsive edge. We denote by $C^-(G,c) \subseteq C(G)$ the set of all such cycles.

We consider the relaxation of $CYC(G)$ that is constrained only by conflicted cycles. More specifically, we consider the system

$$\forall C \in C^-(G,c), f \in E_C \cap E^- : \quad x_f \leq \sum_{e \in E_C \setminus \{f\}} x_e$$

of only those linear inequalities of (10) for which the edge on the left-hand side is repulsive and all other edges are attractive. Defining

$$CYC^-(G,c) := \{ x \in [0,1]^E \mid x \text{ satisfies (12)} \} \quad (13)$$

and replacing $CYC(G)$ by $CYC^-(G,c)$ in $P_{CYC}$ has no effect on the solutions, due to the following lemma, a weaker form of which was also given by Yarkony et al. (2015).

**Lemma 2.** For any $c: E \rightarrow \mathbb{R}$ it holds that

$$\min_{x \in CYC(G)} c^\top x = \min_{x \in CYC^-(G,c)} c^\top x \quad (14)$$

and

$$\min_{x \in MC(G)} c^\top x = \min_{x \in CYC^-(G,c) \cap \mathbb{Z}^E} c^\top x. \quad (15)$$

**Proof.** Let $x^*$ be an optimal solution to the right-hand side of (14). We show that $x^*$ satisfies all cycle inequalities (10) by contradiction. To this end, suppose there exists a cycle $C \in C(G)$ and $f \in E_C$ such that

$$x^*_f > \sum_{e \in E_C \setminus \{f\}} x^*_e.$$

If any edge $g \in E_C \setminus \{f\}$ is repulsive, then increasing $x^*_g$ would lower the objective. Since $x^*$ is optimal, there must be a conflicted cycle $C'$ with $g \in E_{C'}$ such that

$$x^*_g = \sum_{e \in E_{C'} \setminus \{g\}} x^*_e. \quad \text{Note that this means } f \notin E_{C'}.$$

We write $C \Delta C'$ for the cycle obtained from the symmetric difference of $E_C$ and $E_{C'}$. Apparently, the cycle $C \Delta C'$ has one repulsive edge less and $f \in E_{C \Delta C'}$. Therefore, by repeating the argument, we may w.l.o.g. assume that all edges in $E_C \setminus \{f\}$ are attractive.

Now assume that $f$ is attractive as well, then decreasing $x^*_f$ would lower the objective. Therefore, since $x^*$ is optimal, there is a conflicted cycle $C'$ with $f \in E_{C'}$ and $g \in E_{C'} \cap E^-$ such that

$$x^*_g = x^*_f + \sum_{e \in E_{C'} \setminus \{f,g\}} x^*_e$$

$$\geq \sum_{e \in E_C \setminus \{f\}} x^*_e + \sum_{e \in E_{C'} \setminus \{f,g\}} x^*_e$$

$$\geq \sum_{e \in E_{C \Delta C'} \setminus \{g\}} x^*_e.$$
Note that $C \triangle C'$ is a conflicted cycle. Thus, we conclude that $x^*$ violates an inequality of (12) and hence cannot be feasible. This concludes the proof of (14), the argument for (15) is analogous.

With the help of Lemma 2, we formulate $P_{SC}$ with

$\text{PSC} \bigcap \mathbb{Z}^E$ is analogous.

4. Partial Optimality

In this section, we study partial optimality for $P_{MC}$. More precisely, we establish conditions on an edge $e \in E$ which guarantee that $x_e$ assumes one particular value, either 0 or 1, in at least one optimal solution (weak persistency). Fixations to 0 are of particular interest as they can be implemented as edge contractions (with subsequent merging of parallel edges), which effectively reduce the size of a given instance of the problem. As a corollary, we obtain an algorithm that solves weighted correlation clustering problems on series-parallel graphs in linear time.

4.1. Basic Conditions

A direct consequence from Lemma 3 is that we may disregard all edges that are not contained in any conflicted cycle. There are (at least) two ways this can happen: 1. An edge $e \in E$ is not contained in any cycle at all, that is, $e$ is a bridge. 2. The endpoints of a repulsive edge $e = \{u, v\} \in E^-$ belong to different components of $G^+ = (V, E^+)$. In both cases, for any optimal solution $x^*$ of $P_{MC}$, it holds that $x_e^* = 0$ if $e$ is attractive, and $x_e^* = 1$ if $e$ is repulsive. Thus, we can restrict the instance of the problem to the maximal components of $G$ that are connected in $G^+$ and biconnected in $G$. This was also observed by Alush & Goldberger (2012).

Below, we establish more general partial optimality conditions. To this end, we need the following notation. A cut of $G$ is a bipartition $B = (S_1, S_2)$ of the nodes $V$, i.e. $V = S_1 \cup S_2$. The edge set of the cut $B$ is denoted by $E_B = \{uv \in E \mid u \in S_1, v \in S_2\}$.

4.2. Dominant Edges

Definition 4. Let $G = (V, E)$ be any graph and let $c \in \mathbb{R}^E$. An edge $f \in E$ is called dominant attractive iff $c_f > 0$ and there exists a cut $B$ with $f \in E_B$ such that

$$c_f \geq \sum_{e \in E_B \setminus \{f\}} |c_e|.$$ (20)

An edge $f \in E^-$ is called dominant repulsive iff $c_f < 0$ and there exists a cut $B$ with $f \in E_B$ such that

$$c_f \leq \sum_{e \in E_B \cap E^+} c_e.$$ (21)

An edge is called dominant iff it is dominant attractive or dominant repulsive.

Lemma 4. Let $G = (V, E)$ be any graph and let $c \in \mathbb{R}^E$.

(i) If $f \in E$ is dominant attractive, then $x_f^* = 0$ in at least one optimal solution $x^*$ of $P_{MC}$.

(ii) If $f \in E$ is dominant repulsive, then $x_f^* = 1$ in at least one optimal solution $x^*$ of $P_{MC}$.

Proof. (i) We use the set covering formulation of $P_{MC}$. Suppose $f \in E^+$ is dominant and $\hat{x}_e^* = 1$ in an optimal solution $\hat{x}^*$ of $P_{SC}$. Every conflicted cycle that contains $f$ also contains some edge $e \in E_B$, since $B$ is a cut. Therefore, the vector $\hat{x} \in \{0, 1\}^E$ defined by

$$\hat{x}_e = \begin{cases} 0 & \text{if } e = f \\ 1 & \text{if } e \in E_B, e \neq f \\ \hat{x}_e^* & \text{else} \end{cases}$$
is a feasible solution to $P_{SC}$. It has the same objective value as $\hat{x}^*$, since $f$ is dominant and $\hat{x}^*$ is optimal.

(ii) Suppose $f \in E^{-}$ is dominant and $\hat{x}^*_f = 1$ in an optimal solution $\hat{x}^*$ of $P_{SC}$. Every conflicted cycle that contains $f$ also contains some edge $e \in E_B \cap E^+$, since $B$ is a cut and every conflicted cycle contains only one repulsive edge. Then the vector $\hat{x} = \{0, 1\}^E$ defined by $\hat{x}_f = 0, \hat{x}_e = 1$ for all $e \in E_B \cap E^+$ and $\hat{x}_e = \hat{x}^*_e$ elsewhere is a feasible solution to $P_{SC}$. It has the same objective value as $\hat{x}^*$, since $f$ is dominant and $\hat{x}^*$ is optimal.

Lemma 4 generalizes the basic conditions discussed in Section 4.1, since each edge $f \in E$ that is not contained in any conflicted cycle is also dominant. Dominance of edges can be decided in polynomial time, by computing minimum st-cuts in $G$ for a suitable choice of capacities. In practice, the required computational effort may be mitigated by constructing a cut tree of $G$ (Gomory & Hu, 1961). The practically most relevant cuts can even be checked in linear time, which we discuss in the following section.

4.3. Two-Edge Cuts & Single-Node Cuts

In practice, it is expected that dominant edges are more likely to be found in cuts that are relatively sparse. We discuss two special cases of sparse cuts that are of particular interest, due to the following reasons. First, they can be checked in linear time, which gives rise to a fast preprocessing algorithm. Second, we show that our techniques solve $P_{MC}$ to optimality if $G$ is series-parallel.

Two-edge cuts. Suppose $B$ is a two-edge cut of $G$, i.e. $E_B = \{e, f\}$ for two edges $e, f \in E$. Apparently, according to (20) and (21), at least one of them must be dominant. Furthermore, it is guaranteed that we can simplify the instance by edge deletions or contractions. To see this, distinguish the following cases. If both $e$ and $f$ are repulsive, then both of them are dominant and we can delete them, as they are not contained in any conflicted cycle. If $f$ is dominant attractive, we can contract $f$. Finally, if $f$ is dominant repulsive and $e$ is attractive, then we can switch the signs of their coefficients and redefine $x_f := 1 - x_f$ as well as $x_e := 1 - x_e$. Since $|E_B| = 2$, this operation does not change the set of conflicted cycles of $G$ and thus is valid (while only adding a constant to the objective). Afterwards, the edge $f$ is dominant attractive and we can contract $f$. The two-edge cuts of $G$ can be found in linear time, by computing the 3-edge-connected components of $G$, cf. (Mehlhorn et al., 2017).

Single-node cuts. For any $v \in V$, let $B_v = (\{v\}, V \setminus \{v\})$ denote the cut that is induced by $v$. Whether $E_{B_v}$ contains a dominant edge is easily decided by considering all edges incident to $v$. Moreover, if $\deg v = 2$, then $B_v$ is also a two-edge cut and we can apply the operation described in the last paragraph. Updating the graph and applying these techniques recursively as specified in Algorithm 1 takes linear time. This has the following theoretical consequence.

Corollary 1. If $G$ has treewidth at most 2, then Algorithm 1 can be implemented to solve $P_{MC}$ exactly in $O(|V|)$ time.

Proof. Place the vertices of $G$ into buckets of ascending degree and always pick a vertex of minimal degree. Every graph of treewidth 2 has a vertex $v$ with $\deg v \leq 2$. Since Algorithm 1 only contracts or deletes edges, fixing the variables according to Lemma 4, the updated graph still has treewidth at most 2. The number of nodes decreases by 1 in every iteration, hence the algorithm terminates in time $O(|V|) = O(|E|)$ and outputs an optimal solution. □

5. Dual Lower Bounds

In this section, we define an algorithm for computing lower bounds for $P_{MC}$. This algorithm exploits the structure of
Algorithm 2 Iterative Cycle Packing (ICP)

**Input:** $G = (V, E), c: E \to \mathbb{R}$

1. Initialize $w_e = |c_e|$ for all $e \in E$ and $y = 0$, $L = L^{\text{triv}}$.
2. for $\ell = 3 \ldots |E|$ do
3. \hspace{1em} while $\exists C \in C^-(G, c) : |E_C| \leq \ell$ do
4. \hspace{2em} Pick $C \in C^-(G, c)$ such that $|E_C| \leq \ell$.
5. \hspace{2em} Compute $y_C = \min_{e \in E_C} w_e$.
6. \hspace{2em} Redefine $w_e = \begin{cases} w_e - y_C & \text{if } e \in E_C \\ w_e & \text{else.} \end{cases}$
7. \hspace{2em} Increase lower bound $L = L + y_C$.
8. \hspace{2em} Remove all edges $e \in E$ with $w_e = 0$ from $G$.
9. \hspace{1em} end while
10. if $C^-(G, c) = \emptyset$ then return $y, L$
11. end if
12. end for

the reformulation $P_{SC}$. It computes a heuristic solution to the dual of its LP relaxation.

The LP relaxation (up to the constant $L^{\text{triv}}$) of problem $P_{SC}$ is given by

$$\min \sum_{e \in E} c_e \hat{x}_e$$

subject to

$$\begin{align*}
\sum_{e \in E_C} \hat{x}_e & \geq 1 & \forall C \in C^-(G, c) \\
\hat{x}_e & \geq 0 & \forall e \in E .
\end{align*}$$

The corresponding dual program reads

$$\max \sum_{C \in C^-(G, c)} y_C$$

subject to

$$\begin{align*}
\sum_{C : e \in E_C} y_C & \leq |c_e| & \forall e \in E \\
y_C & \geq 0 & \forall C \in C^-(G, c) .
\end{align*}$$

A heuristic solution of (24), and thus a lower bound for (22), is found by Algorithm 2 that we call Iterative Cycle Packing (ICP). It works as follows: Firstly, it chooses a conflicted cycle $C$ and increases $y_C$ as much as possible. Secondly, it decreases the weights $w_e$ (initially $|c_e|$) of all edges $e \in E_C$ by $y_C$ and removes all edges of zero weight. These steps are repeated until there are no conflicted cycles left.

**Implementation details.** The absolute running time of ICP as well as the quality of the output lower bounds depends on the choice of cycles $C$. We pursue the following strategy that we found to perform well empirically in both aspects: In each iteration of the main loop, we choose a repulsive edge $e = uv \in E^r$ such that $u$ and $v$ are in the same connected component of $G^+ = (V, E^+)$, and then we find a conflicted cycle containing $e$ by searching for a shortest path from $u$ to $v$ in $G^+$.

6. Re-weighting for Primal Algorithms

In this section, we exploit the dual solution in primal algorithms. The motivation is due to complementary slackness, which is made explicit in the following lemma.

**Lemma 5.** Assume the primal LP (22) is tight, i.e., its optimal solution $\hat{x}^*$ also solves $P_{SC}$, and the solution output by ICP solves the dual (24) optimally. Then, for every $e \in E$ with positive residual weight $w_e > 0$, it holds that $\hat{x}_e^* = 0$.

**Proof.** If $w_e > 0$, the constraint (25) at $e \in E$ is inactive at the optimal dual solution. Thus, $\hat{x}_e^* = 0$ in the optimal primal solution, by complementary slackness.

Of course, the assumption of Lemma 5 is too strong for practical purposes. However, the intuition is that if the LP relaxation is fairly tight and the obtained dual solution is close to optimal, it can still provide useful information about the primal problem. More specifically, the weights $w_e$ output by ICP can be interpreted as an indication of how likely the primal variable $\hat{x}_e$ is zero in an optimal solution. In order to make use of this information, we propose to shift the weights of the primal problem to a convex combination $\lambda |c_e| + (1 - \lambda)w_e$ of the original and residual weights, for a suitable choice of $\lambda \in (0, 1)$. Experiments in Section 7 show that this shift can guide primal heuristics toward better feasible solutions to the original problem.

7. Experiments

In this section, we study partial optimality, dual lower bounds and re-weightings empirically, for all instances of
the weighted correlation clustering problem from Kappes et al. (2015) and Leskovec et al. (2010).

**Instances.** From Kappes et al. (2015), we consider all three collections of instances: *Image Segmentation* contains instances w.r.t. planar superpixel adjacency graphs of photographs. *Knott-3D* contains instances w.r.t. non-planar supervoxel adjacency graphs of volume images taken by a serial sectioning electron microscope. *Modularity Clustering* contains instances w.r.t. complete graphs. In all three collections, the edge costs $c_e$ are fractional and non-uniform. For all these instances, except one in the collection *Modularity Clustering*, optimal solutions are accessible and are computed here as a reference. From Leskovec et al. (2010), we consider directed graphs of the social networks *Epinions* and *Slashdot*, each with more than half a million edges labeled either +1 or −1. Instances of the minimum cost multicut problem are defined here by removing the orientation of edges, by deleting all self-loops, and by replacing parallel edges by a single edge with the sum of their costs.

### 7.1. Partial Optimality

In order to study the partial optimality conditions of Section 4 empirically, we process the above instances as follows: First, we remove all edges of cost 0; all bridges, as well as all repulsive edges whose nodes belong to distinct connected components of $G^+$. Second, we check for every $v \in V$ whether the cut $B_v = (\{v\}, V \setminus \{v\})$ induces dominant edges. If we find dominant attractive edges or vertices of degree $\leq 2$, we perform contractions and deletions according to Alg. 1. Both steps are repeated until no further edges can be removed or contracted.

After the main reduction step, which takes linear time and is thus very fast, we further check all remaining edges $uv \in E$ for dominance in any (general) $uv$-cut. To this end, we construct a cut tree of $G$ with the help of Gusfield’s algorithm (Gusfield, 1990), which takes $|V| \sim 1$ max-flow computations. Despite the increased computational effort, we only found a small number of additional dominant attractive edges and thus could only perform few further contractions. However, we found a significant number of additional dominant repulsive edges.

The effect of our method in the total number of nodes and edges is shown in Table 1. We also report the number of remaining edges that are not dominant repulsive. It can be seen from this table that the numbers are effectively reduced. This is explained, firstly, by the sparsity of the graphs and, secondly, by the non-uniformity of the costs. From the comparison to the number of remaining non-persistent variables when only the criteria of Alush & Goldberger (2012) are applied, it can be seen that our more general criteria reveal considerably more persistency.

It may be expected that optimization methods benefit in terms of runtime from the reduced size of the instances. On the instances of Kappes et al. (2015), we found the effect to be insignificant due to their small original size. On *Epinions* and *Slashdot*, however, the runtime of the local search algorithm GAEC+KLj (cf. Section 7.3) decreased by more than 70%. For completeness, we provide the numbers in the supplements.

### 7.2. Dual Lower Bounds

In order to put into perspective the dual lower bounds output by Iterative Cycle Packing (ICP) as described in Section 5, we compare this algorithm, firstly, to the cutting plane algorithm for $P_{CYC}$ of Kappes et al. (2015), with Gurobi for solving the LPs (denoted here by LP) and, secondly, to the message passing algorithm of Swoboda & Andres (2017), applied to $P_{CYC}$, with code and parameter settings kindly provided by the authors (denoted here by MPC).

Results are shown in Figure 1 and Table 2. It can be seen from the figure and the table that, for the large and hard instances *Epinions* and *Slashdot*, ICP converges at under $10^2$ seconds, outputting lower bounds that are matched and exceeded by MPC at around $10^3$ seconds. It can be seen from Table 2 that the situation is similar for the smaller instances: The lower bounds output by ICP are a bit worse than those output by LP or MPC (here compared to the best optimal solution known) but are obtained faster (by as much as three orders of magnitude for *Knott-3D-450*).

It is known from Kappes et al. (2015) that their instances can be solved faster than their LP relaxations by means of branch-and-cut, separating only integer infeasible points.
by cycle inequalities using BFS (instead of Dijkstra’s algorithm), and resorting to the strong (undisclosed) cuts of Gurobi for cutting off fractional solutions. We restrict our comparison here to algorithms that seek to solve the LP relaxation $P_{\text{Pyc}}$. This is justified by the fact that size ultimately renders integer linear programming intractable. We conclude that ICP is capable of computing non-trivial lower bounds fast.

### 7.3. Re-weighting

In order to study the re-weighting described in Section 6, we measure its effect on heuristic algorithms for finding feasible solutions. To this end, we employ the implementations of Levinkov et al. (2017) of Greedy Additive Edge Contraction (GAEC), an algorithm that starts from singleton clusters and greedily contracts attractive edges with maximum non-negative cost, and of KLj, the well-known Kernighan-Lin heuristic for graph partitioning that recursively improves an initial clustering by splitting, merging or exchanging nodes between neighboring clusters.

A comparison between the feasible solutions found by applying the heuristics GAEC and GAEC+KLj to original instances, on the one hand, and to instances re-weighted by ICP with $\lambda = \frac{1}{2}$, on the other hand, can be found in Table 3. Note that we only re-weight the input to GAEC and let KLj run with original weights, starting from the solution returned by GAEC, as we found this approach to be advantageous. It can be seen from Table 3 that our re-weighting consistently improves the gap. On average, it is slightly less effective than the reparameterization with the more accurate dual solutions obtained from MPC, as proposed by Swoboda & Andres (2017). A more detailed comparison is provided in the supplements.

### 8. Conclusion

We have established partial optimality conditions, a heuristic lower bound and a heuristic re-weighting for instances of the weighted correlation clustering problem. We have shown advantages of each of these constructions empirically. Checking a subset of our partial optimality conditions recursively gives a fast combinatorial algorithm that efficiently reduces the size of problem instances. Conceptually, it solves the problem for series-parallel graphs to optimality, in linear time. Our dual heuristic algorithm provides non-trivial lower bounds and valuable dual information fast. For future work, it is relevant to examine if more sophisticated dual solvers such as MPC benefit from a “warm-start” that transforms and exploits the heuristic dual solution.
References


