“ADFOCS’23: Exercises
Vincent Cohen-Addad
August 21 2023

In this series of exercises, we will be deriving a simple, versatile, practical $O(1)$-approximation algorithm for the Correlation Clustering problem. We encourage you to watch the great videos that Evangelos Kipouridis has prepared on the pivot algorithm for Correlation Clustering. This is not necessary, rather complementary to the result we will be deriving here.

1 Setup

We first recall the definition and basic properties about the Correlation Clustering problem.

Definition 1.1. *Given an unweighted undirected graph $G = (V, E)$, the goal of the Correlation Clustering problem is to find a partition of arbitrary number of sets $\{C_1, \dots, C_t\}$ of V that minimizes*

$$f(C_1, \dots, C_t) = \sum_{\substack{\{u,v\} \in E: \\ u \in C_i, v \in C_j, i \neq j}} 1 + \sum_{\substack{\{u,v\} \notin E: \\ u,v \in C_i}} 1.$$

The problem is NP-Hard and so we aim at providing an approximation algorithm. The set of edges is often refer to as the set of *positive* edges, while the set of non-edges is the set of *negative* edges.
In the following, we will say that two vertices u, v are neighbors is $(u, v) \in E$. For a vertex $v \in V$, we refer to its neighborhood by $N(v)$ and to its degree by $d(v)$; we further let $N(v, H)$ denote the neighborhood of v in a subgraph H of G. We also consider the degree of a vertex v induced on an arbitrary subset $S \subseteq V$ of nodes and we denote it by $d(v, S)$. Finally for any two sets R, S, we denote their symmetric difference by $R \triangle S$.
Our goal is to show that one can obtain a simple *local* algorithm for Correlation Clustering, where all the clustering decisions are made locally.
Our algorithm is parameterized by two constants β, λ. We should simply think of β, λ as small constants, i.e.: $\beta < 1/20$. Our final approximation guarantees will be of the form $(\beta\lambda)^{-c}$ for some constant c.

Definition 1.2 (Weak Agreement). *Two vertices u and v are in i-*weak agreement *if $|N(u) \triangle N(v)| < i\beta \cdot \max\{|N(u)|, |N(v)|\}$. If u and v are in 1-weak agreement, we also say that u and v are in* agreement.

Having the agreement notion in hand, we provide our approach in Algorithm 1.
We will refer to $\widetilde{G}$ as the *sparsified graph.*
Our goal in the rest of this sequence of exercises is to:

Algorithm 1 Correlation-Clustering(G)

```
1: Discard all edges whose endpoints are not in agreement. (First compute the set of these
   edges. Then remove this set.)
2: Call a vertex *light* if it has lost more than a $\lambda$-fraction of its neighbors in the previous
   step. Otherwise call it *heavy*.
3: Discard all edges between two light vertices.
4: Let $\widetilde{G}$ be the resulting graph. Compute its connected components, and output them as
   the solution.
```

1. Capture the structure of $\widetilde{G}$;

2. Based on the above, show that the connected components of $\widetilde{G}$ are good clusters.

2 Basic Properties of the Algorithm

In this section, we will analyze the properties of $\widetilde{G}$.

Exercise 2.1. *Suppose that $\beta < \frac{1}{20}$.*

Question 1: $\forall u, v$, if u, v are in i-weak agreement, for some $1 \le i < \frac{1}{\beta}$, then show that

$$(1-\beta i)d(u) \le d(v) \le \frac{d(u)}{1-i\beta}.$$

Question 2: Let $k \in \{2,3,4,5\}$ and $v_1, \ldots, v_k \in V$ be a sequence of vertices such that v_i is in agreement with v_{i+1} for $i = 1, \ldots, k-1$. Then show that v_1 and v_k are in k-weak agreement.

Question 3: If u and v are in i-weak agreement, for some $1 \le i < \frac{1}{\beta}$, then show that $|N(v) \cap N(u)| \ge (1-i\beta)d(v)$.

We can now use the above results to start understanding the structure of the connected components of $\widetilde{G}$.

Exercise 2.2. *Suppose that $5\beta + 2\lambda < 1$. Let CC be a connected component of $\widetilde{G}$. Then,*

Question 1: Using the Questions 2 and 3 of the previous exercise, show that for every $u, v \in CC$, if u and v are heavy, then the hop-distance between u, v in $\widetilde{G}$ is at most 2.

Question 2: Show that for every $u, v \in CC$, the hop-distance between u, v in $\widetilde{G}$ is at most 4.

Question 3: *Using the Questions 2 and 3 of the previous exercise, show that for every $u, v \in CC$, then the hop-distance between u and v in G is at most 2.*

Question 4: *Using Question 2 of the previous exercise, show that for every $u, v \in CC$, if u or v is heavy, then u and v are in 4-weak agreement. [Hint: The reasoning is similar as the second question of this exercise.]*

Now, we would like to show that the connected components are "good" clusters. What would be a "good" cluster? A cluster is objectively good if each vertex in the cluster is connected to a large fraction of the other vertices in the cluster. This is our next exercise. Observe first that any connected component of $\widetilde{G}$ of size at least 2 must have a heavy vertex. We can then derive the following result.

Exercise 2.3. *Let CC be a connected component of $\widetilde{G}$ such that $|CC| \geq 2$. Show that for each vertex $u \in CC$ we have that*

$$d(u, CC) \geq (1 - 8\beta - \lambda)|CC|.$$

[Hint: You may want to consider a heavy vertex $v \in CC$ and reason from it using the properties derived at the previous exercises. In particular, Question 4 of the previous exercise and Questions 1 and 3 of the first exercise may become handy for your argument.]

For the cluster to be good, we would like to make sure that $8\beta + \lambda$ is a small number (e.g.: smaller than 1%), so that the clusters found are "near-cliques". This motivates our assumption about the values of β and λ.
We are going to conclude our analysis of $\widetilde{G}$ with two crucial observations.

Exercise 2.4. *Let CC be a connected component in $\widetilde{G}$. Assume that $8\beta + \lambda \leq 1/4$. Show that the cost of keeping CC as a single cluster in G is no larger than the cost of splitting CC into two or more clusters. [Hint: For this proof, you may want to consider the cost of splitting CC into 2 or more clusters $C_1, \dots, C_k$ and show that this cost is higher than keeping CC as a single cluster. You may want to make a case distinction: Either there exists a C_i that contains a large fraction of CC or not. Use the result of the previous exercise to bound the cost.]*

We conclude with a last but important property about $\widetilde{G}$ that will facilitate our analysis of the approximation guarantee.

Exercise 2.5. *Let $G' = (V, E')$ be the correlation clustering instance obtained from $G = (V, E)$ by deleting all the edges (u, v) that are deleted by our algorithm (or in other words, all the edges that our algorithm removes are now turn into negative edges). Then, show that our algorithm outputs a solution that is optimal for the instance G'.*

3 Approximation Guarantee

We now have all the tools we need to prove that our algorithm outputs an $O(1)$-approximate solution for our problem. In our analysis we will consider a fixed optimal solution (of instance G), denoted by $\mathcal{O}$, whose cost is denoted by OPT.
Recall that our algorithm returns a clustering that is optimal for the instance G' defined at the end of the previous section. Therefore, to bound the approximation ratio of our solution we need to bound the cost in G of an optimal clustering for G'. To do so, it is enough to bound the number of edge in G that are not in G' – since this is the set of edges that have been deleted by our algorithm.
The main intuition behind the proofs of the next exercises is that when two vertices are not in agreement, or when a vertex is light, then there are many edges (or non-edges) in the 1-hop or 2-hop vicinity that $\mathcal{O}$ pays for. We can thus build a charging scheme to account for them.

Exercise 3.1. *Show that the number of edges deleted in Line 2 of our algorithm that are not cut in $\mathcal{O}$ is at most $\frac{2}{\beta} \cdot OPT$. [Hint: One way to prove this is to define a charging argument with the following strategy. Each edge as in the statement will distribute fractional debt to edges (or non-edges) that $\mathcal{O}$ pays for, in such a way that (1) each edge as in the statement distributes debt worth at least* 1 *unit, and (2) each edge/non-edge that $\mathcal{O}$ pays for is assigned at most $\frac{2}{\beta}$ units of debt, and (3) edges/non-edges that $\mathcal{O}$ does* not *pay for are assigned no debt. Show that a scheme that satisfies the above yields the desired bound.]*

Exercise 3.2. *Show that the number of edges deleted in Line 3 of our algorithm that are not cut in $\mathcal{O}$ is at most $\left(\frac{1}{\beta} + \frac{1}{\lambda} + \frac{1}{\beta\lambda}\right) \cdot OPT$. [Hint: We propose the following charging argument inspired from the previous exercise. For each endpoint $y \in \{u, v\}$, we proceed as follows. As y is light, there are edges $(y, v_1), ..., (y, v_{\lambda \cdot d(y)})$ whose endpoints are not in agreement. For each $i = 1, ..., \lambda \cdot d(y)$, proceed as follows:*

- *If (y, v_i) is not cut by $\mathcal{O}$, then, as in the proof of* **??**, *(y, v_i) has at least $\beta \cdot \max(d(y), d(v_i))$ adjacent edges/non-edges for whom $\mathcal{O}$ pays. Each of these edges/non-edges is of the form (v_i, w) or (y, w). We will have the edge (u, v) charge $\frac{1}{2\beta\lambda d(v_i) d(y)}$ units of debt, which we will call* ***blue debt****, to the former ones (those of the form (v_i, w)), and $\frac{1}{2\beta\lambda d(y)^2}$ units of debt, which we will call* ***red debt****, to the latter ones (those of the form (y, w)).*[1]
- *If (y, v_i) is cut by $\mathcal{O}$, then $\mathcal{O}$ pays for (y, v_i). We will have the edge (u, v) charge $\frac{1}{2\lambda d(y)}$ units of debt, which we will call* ***green debt****, to (y, v_i).*

Show that the above charging scheme allows to derive the lemma.]

Exercise 3.3. *Deduce that our algorithm outputs an $O(1)$-approximate solution.*

[1]Notice that the latter edges/non-edges might be charged many times by the same y (for different i).

4 Maximization Version

The "maximization version" of the problem has also been studied. The maximization version goes as follows.

Definition 4.1. *Given an unweighted undirected graph $G = (V, E)$, the goal of the Maximization Version of the Correlation Clustering problem is to find a partition of arbitrary number of sets $\{C_1, \ldots, C_t\}$ of V that maximizes*

$$f^{max}(C_1, \ldots, C_t) = \sum_{\substack{\{u,v\} \notin E: \\ u \in C_i, v \in C_j, i \neq j}} 1 + \sum_{\substack{\{u,v\} \in E: \\ u,v \in C_i}} 1.$$

Of course the maximization version of the problem is equivalent to the minimization version when we talk about exact algorithms. However, when considering approximation algorithms, the maximization version is significantly less interesting, as you can show in the following exercise.

Exercise 4.1. *Provide a 2-approximation to the Maximization Version of the Correlation Clustering problem.*

5 Beyond Unweighted Graphs

As we have discussed in class, it is often very valuable to be able to handle edge weights that represent how similar or dissimilar two vertices are. For this, we may want to analyze the weighted correlation clustering problem that we define next.

Definition 5.1. *Given an undirected graph $G = (V, E)$, and a weight function $w : E \mapsto \mathbb{R}_+ \cup \{0\}$, the goal of the Weighted Correlation Clustering problem is to find a partition of arbitrary number of sets $\{C_1, \ldots, C_t\}$ of V that minimizes*

$$f^{weight}(C_1, \ldots, C_t) = \sum_{\substack{\{u,v\} \in E: \\ u \in C_i, v \in C_j, i \neq j}} w(u,v) + \sum_{\substack{\{u,v\} \notin E: \\ u,v \in C_i}} w(u,v).$$

This problem is unfortunately drastically much hard than the unweighted version we studied in the previous sections. This problem is in fact "equivalent" in terms of approximation guarantee to the Multicut problem, which we define below.

Definition 5.2. *Given an undirected graph $G = (V, E)$, a weight function $w : E \mapsto \mathbb{R}_+ \cup \{0\}$, and a collection of pairs of vertices of V $\{(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)\}$, the goal of the Weighted Multicut problem is to find a collection of edges $E_S \subseteq E$ of minimum total weight such that for all $i \in [k]$, s_i and t_i are not in the same connected component of $G' = (V, E - E_S)$.*

We now provide a gap-preserving equivalence between Weighted Multicut and Weighted Correlation Clustering. Hint for the next two exercises: Observe that when creating a Weighted Correlation Clustering, it is possible to have "don't care" pairs of vertices (u, v), i.e.: edges (u, v) of weight 0 so that it does not matter for the objective whether u and v are clustered together or not.

Exercise 5.1. *Show that if we have a polynomial time α-approximation algorithm for Weighted Correlation Clustering, then we have a polynomial-time α-approximation algorithm for Weighted Multicut.*

Exercise 5.2. *Show that if we have a polynomial time α-approximation algorithm for Weighted Multicut, then we have a polynomial-time α-approximation algorithm for Weighted Correlation Clustering. [Hint: While the cost contribution of the edges in E is similar in both problems, you need to find a gadget that turns non-edges into s_i, t_i pairs.]*