CS 7880 Special Topics in TCS: Sublinear Algorithms (Fall'22) **Northeastern University**

Lecture 18

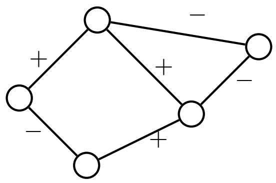
November 15, 2022

Instructor: Soheil Behnezhad *Scribe: William Schultz*

Disclaimer: *These notes have not been edited by the instructor.*

1 Correlation Clustering

The *correlation clustering* problem, introduced in [BBC02], considers a set of n objects (e.g. vertices) along with a $+/-$ labeling of each vertex pair. The goal is to partition the vertices into a set of clusters such that $+$ pairs "tend" to be in the same cluster and $-$ pairs tend to be in different clusters.



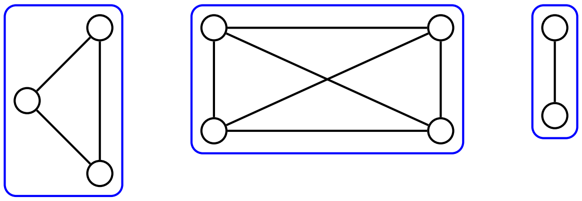
Intuitively, the $+/-$ labelings may encode some notion of desired similarity/closeness/affinity between pairs of vertices, and we want the chosen clusters to reflect this. More concretely, we can look at either maximizing *agreements* (number of $+$ edges inside clusters plus the number of $-$ edges between clusters), or minimizing *disagreements* (number of $-$ inside clusters plus the number of $+$ edges between clusters). These two can be seen as equivalent at optimality (but may differ from an approximation perspective). We focus on *minimum disagreement clustering on complete instances.* That is,

- For every pair we have a label $\{+, -\}$
- Objective: minimize (# of $+$ pairs cut) + (# of $-$ pairs not cut)

Remark. For complete instances of the problem (i.e. where every vertex pair has a labeling), it is useful to view the input as a graph. That is, just take the $+$ labelings between vertices as edges of the graph and $-$ labelings as non-edges of the graph.

So, we can formally view the clustering problem in this case as taking a graph $G = (V, E)$ as input, and our goal is to cluster the vertices V such that disagreements are minimized.

Consider a basic example, shown with its optimal clustering:



The above scenario consists of a set of disjoint cliques, in which case the optimal clustering consists of placing the vertices of each clique into their own, separate cluster. This clustering has zero cost, since it doesn't cut any $+$ edges (since each cluster is a clique), and we do not join any $-$ edges (since each cluster is disconnected from all others).

> **Remark.** In general, we have a cost zero solution iff the input is a union of disjoint cliques.

The correlation clustering problem was shown to be APX-hard [CGW03]. That is, even obtaining even a $1+\epsilon$ approximation to the optimal solution is NP-hard.

2 3-approximation Clustering Algorithm

The best currently known polynomial time algorithm for clustering achieves a 1.994 approximation [CALN22]. These approaches, however, typically require some use of LP (linear programming) as a component. The best known "combinatorial" algorithm obtains a 3-approximation [ACN05]

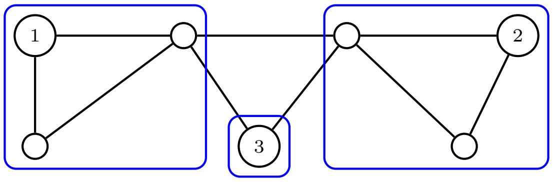
> **Remark.** The first constant approximation algorithm for the min-disagreement correlation clustering problem was introduced in [BBC02]. A 4-approximation algorithm was given in [CGW03].

The 3-approximation algorithm of [ACN05] follows a straightforward, randomized procedure:

Pivot Algorithm [ACN05] (PIVOT)

1. Pick a random "pivot" vertex $v \in V$ uniformly.
2. Cluster v with all of its remaining neighbors.
3. Remove the clustered vertices and recurse on remaining vertices.

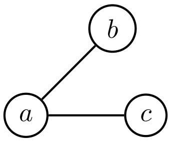
For example, the PIVOT algorithm may produce the following clustering on the given input, with pivot selection order annotated on the relevant nodes:



Theorem 1. *The expected cost of the PIVOT algorithm is at most 3 times the cost of an optimal clustering.*

Proving Theorem 1 is our main goal. The proof argument relies on a notion of *bad triangles*, which are defined as follows.

Definition 2 (Bad Triangle)**.** Three disjoint vertices a, b, c form a *bad triangle* if exactly 2 of the pairs are adjacent.



Claim 3. *Any clustering makes at least 1 mistake on one of the pairs of each bad triangle.*

Proof. Consider any bad triangle a, b, c. Either:

(i) The clustering includes $\{a, b, c\}$ in the same cluster, in which case there is there is a mistake made for the non-edge.

(ii) The clustering excludes at least one vertex in $\{a, b, c\}$, in which case one of the positive edges must be cut, since every vertex a, b, c is connected to some other vertex in the triangle.

□

Remark. Note that even though there must be at least one mistake per bad triangle, this does not mean that existence of k bad triangle implies any clustering has cost at least k. Bad triangles may share edges, so the cost may actually be lower than this.

2.1 3-approximation Analysis

We use reasoning about the existence and cost charging of bad triangles to drive the main arguments of the proof of Theorem 1, along with use of a linear program and its dual to establish the appropriate lower bound on the optimal solution cost.

Let T be the set of all bad triangles for a given input. By Claim 3, we know that any clustering must be charged at least unit cost for each bad triangle. Clearly, for a set of *edge disjoint* triangles, the number of bad triangles would be a lower bound for the optimal cost. This is also true *fractionally.* That is, if for every triangle we assign a set of non-negative weights $\{\beta_t\}$ such that for all edges $e \in E$, the sum of edge weights for that triangle is ≤ 1, then it must be that $\sum_{t \in T} \beta_t \leq OPT$. So, consider the following linear program:

$$\begin{aligned} &\text{minimize:} \quad \sum_{\{a,b\}, a \neq b} x_{\{a,b\}} \\ &\text{subject to:} \quad \underset{\{a,b\}}{x} + \underset{\{b,c\}}{x} + \underset{\{a,c\}}{x} \geq 1 \\ &\text{and } x \geq 0 \\ \\ &\text{for any bad triangle } \{a, b, c\} \in T \end{aligned} \tag{LP}$$

Claim 4. $val(LP) \leq OPT$

Proof. Consider an optimal clustering C, and let $x_{\{a,b\}} = 1$ iff the edge $\{a, b\}$ is a mistake (i.e. in disagreement) in the clustering C. Such an assignment of weights to edges is a valid solution to the LP conditions above, so the solution to the LP must be $\leq OPT$. □

Above we used the stated LP to establish a lower bound on OPT, the cost of the optimal clustering. Next, we consider the dual LP, where y_t is a fractional assignment $\in [0, 1]$ to each bad triangle $t \in T$.

$$\begin{aligned} &\text{maximize} \quad \sum_{t=\{a,b,c\} \in T} y_t \\ &\text{subject to} \quad \sum_{\{a,b\} \subseteq t} y_t \leq 1 \quad \text{for any pair } \{a, b\} \\ &\text{and } y \geq 0 \end{aligned} \tag{DUAL}$$

By *strong duality*, we know that $val(LP) = val(DUAL)$. So, if we have a valid solution $\hat{y}$ to the DUAL, it should serve as a lower bound on the cost of the optimal solution i.e.

$$val(\hat{y}) \leq val(DUAL) = val(LP) \leq OPT$$

Now, for any bad triangle $t = \{a, b, c\}$, define

$$\hat{y_t} = Pr[a, b, c \text{ still in graph } \wedge \text{ one is chosen as pivot}]/3$$

We can also construct $\hat{y}$ as a potential solution to the DUAL by assigning $\hat{y_t}$ to every edge in a bad triangle $t \in T$. Next we state and prove two main claims to establish our main result.

Claim 5. $\sum_{t \in T} \hat{y_t} = \mathbb{E}\left[\textit{cost of PIVOT}\right]/3$

Proof. Every time we pick a pivot, the cost we pay is the number of bad triangles involving the pivot whose all 3 vertices still belong to the graph. Every time we pick a pivot we charge each of the bad triangles involving v whose all 3 vertices still in the graph by unit cost. So,

$$\begin{aligned}
\mathbb{E}\left[\text{cost of } PIVOT\right] &= \mathbb{E}\left[\sum_{t \in T} \text{charge to } t\right] \\
&= \sum_{t \in T} \mathbb{E}\left[\text{charge to } t\right] \\
&= \sum_{\{a,b,c\} \in T} Pr[a, b, c \text{ still in graph } \wedge \text{ one is chosen as pivot}] \\
&= \sum_{t \in T} \hat{y_t}
\end{aligned}$$

□

Claim 6. *$\hat{y}$ is a valid solution to DUAL.*

Proof. It suffices to prove that for any pair (a, b)

$$\sum_{t:(a,b) \subseteq t} y_t \leq 1$$

Suppose we pick a as the first pivot. Picking a causes the charging of many bad triangles, so we have the sum

$$\sum_{t:(a,b) \subseteq t} y_t = deg(a)$$

Now, let S be the set of vertices that together with $\{a, b\}$ form a bad triangle. Observe that

$$\sum_{c \in S} Pr[a, b, c \text{ are still in graph} \wedge c \text{ is chosen as pivot}] \leq 1$$

since the first such c that is chosen removes $\{a, b\}$. We now have

$$\begin{aligned}
\sum_{t:(a,b) \subseteq t} y_t = \frac{1}{3}(\sum & Pr[a, b, c \text{ are still in graph } \wedge c \text{ is chosen as pivot}] + \\
& Pr[a, b, c \text{ are still in graph } \wedge a \text{ or } b \text{ is chosen as pivot}] \cdot |S|) \\
&= \frac{1}{3}\left(\frac{|S| - 2}{|S|} + \frac{2}{|S| + 2} \cdot |S|\right) \\
&= \frac{1}{3}\left(1 + \frac{2}{|S| + 2} \cdot |S|\right) \\
&\leq 1
\end{aligned}$$

□

Combining Claims 5 and 6, we have that

$$\mathbb{E}\left[\text{cost of PIVOT}\right]/3 = \sum_{t \in T} \hat{y}_t \leq val(DUAL) = val(LP) \leq OPT$$

which implies our desired result:

$$\mathbb{E}\left[\text{cost of PIVOT}\right] \leq 3 * OPT$$

Remark. Note a similarity of the PIVOT algorithm to previous sublinear algorithms we have seen. Namely, the randomized greedy algorithm for maximal independent set (MIS). Indeed, the set of pivot vertices chosen by PIVOT is exactly the output of the randomized greedy MIS algorithm (for the same permutation). This observation implies the corollary below.

Corollary 7. *There is an $O(n)$ space MPC algorithm finding a 3-approximation of correlation clustering in $O(\log \log n)$ rounds.*

References

[ACN05] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '05, page 684–693, New York, NY, USA, 2005. Association for Computing Machinery. 2

[BBC02] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, page 238, Los Alamitos, CA, USA, nov 2002. IEEE Computer Society. 1, 2

[CALN22] Vincent Cohen-Addad, Euiwoong Lee, and Alantha Newman. Correlation clustering with sherali-adams, 2022. 2

[CGW03] M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 524–533, 2003. 2