**601.435/635 Approximation Algorithms** **Lecturer:** Michael Dinitz
**Topic:** Semidefinite Programming: Correlation Clustering and Max 2-SAT **Date:** 4/18/24
**Scribe:** Michael Dinitz

## 24.1 Introduction

Today we're going to see some further uses of semidefinite programming in approximation algorithms. Our rounding algorithms will still be relatively simple, but we'll design more complex SDP relaxations, as well as do more complex analyses of the roundings.

## 24.2 Correlation Clustering

### 24.2.1 Definitions

We've talked about a few different clustering problems in this class, but let's introduce another one: Correlation Clustering. Formally, it's the following.

- Input:
  - Graph $G = (V, E)$
  - Weight functions $w^- : E \to \mathbb{R}^+$ and $w^+ : E \to \mathbb{R}^+$
- Feasible: Partition $\mathcal{S} = S_1, S_2, \ldots, S_n$ of $V$. Given $\mathcal{S}$, let $\delta(\mathcal{S})$ be the edges between different parts of the partition, and let $E(\mathcal{S})$ be the edges with both endpoints in same part of partition.
- Objective: $\max \sum_{\{i,j\} \in E(\mathcal{S})} w^+(i,j) + \sum_{\{i,j\} \in \delta(\mathcal{S})} w^-(i,j)$

Intuitively, in this problem every edge has a positive and a negative weight (usually representing similarity/dissimilarity), and our goal is to cluster the vertices to maximize the positive weight *within* the clusters plus the negative weight *between* the clusters. In other words we try to maximize similarity in same cluster + dissimilarity in different clusters.

This is a nice formulation of clustering for a few reasons. First, we don't have to prespecify a number of clusters – we're allowed to use as many clusters as we want. Second, by using different functions for $w^+$ and $w^-$, we can model a lot of different settings and problems.

Note that there's a very simple 1/2-approximation: if we set $\mathcal{S} = \{V\}$ then the objective value is $\sum_{e \in E} w^+(e)$, while if we set $\mathcal{S} = \{\{i\} : i \in V\}$ the objective value is $\sum_{e \in E} w^-(e)$. Since for any $\mathcal{S}$ the objective is at most $\sum_{e \in E} w^-(e) + \sum_{e \in E} w^+(e)$, taking the best of the two is a 1/2-approximation.

### 24.2.2 SDP relaxation

To do better we will write an SDP. Unlike Max-Cut, we're not going to start with a strict quadratic program and then relax it, but rather we'll start with a vector program which we cannot solve, and

then relax it to an SDP. Recall that $e_k$ is the $k$'th standard basis vector: the vector with a 1 in the $k$th coordinate and a 0 everywhere else. Then the following vector program is an exact formulation of Correlation Clustering:

$$\begin{aligned} \max \quad & \sum_{\{i,j\}\in E} \left(w^+(i,j)(v_i \cdot v_j) + w^-(i,j)(1 - v_i \cdot v_j)\right) \\ \text{s.t.} \quad & v_i \in \{e_1, e_2, \ldots, e_n\} && \forall i \in V \end{aligned}$$

**Theorem 24.2.1** *This is an exact formulation of* Correlation Clustering

**Proof:** Consider some clustering $\mathcal{S} = S_1, S_2, \ldots, S_k$. For all $i \in [k]$, and for all $j \in S_i$, let $v_j = e_i$. In other words, each vertex gets assigned the standard basis vector corresponding to its cluster. Then this is clearly a feasible set of vectors, and the objective function is

$$\sum_{\{i,j\}\in E} \left(w^+(i,j)(v_i \cdot v_j) + w^-(i,j)(1 - v_i \cdot v_j)\right) = \sum_{\{i,j\}\in E(\mathcal{S})} w^+(i,j) + \sum_{\{i,j\}\in\delta(\mathcal{S})} w^-(i,j).$$

Thus there is a solution to the vector program of the same value as $\mathcal{S}$.

Similarly, let $\{v_i\}_{i\in V}$ be a solution to the vector program. Let $S_i = \{i \in V : v_i = e_i\}$, and let $\mathcal{S} = S_1, S_2, \ldots, S_n$. Then $\mathcal{S}$ is a partition of $V$, and the value is equal to the objective function of the vector program. ■

We cannot solve this vector program (the requirement that the vectors are standard basis vector cannot be expressed as an SDP), but we can relax it to an SDP:

$$\begin{aligned} \max \quad & \sum_{\{i,j\}\in E} \left(w^+(i,j)(v_i \cdot v_j) + w^-(i,j)(1 - v_i \cdot v_j)\right) \\ \text{s.t.} \quad & v_i \cdot v_i = 1 && \forall i \in V \\ & v_i \cdot v_j \geq 0 && \forall i,j \in V \\ & v_i \in \mathbb{R}^n && \forall i \in V \end{aligned}$$

Note that this is not an entirely obvious SDP relaxation: it might have been natural for us not to include the $v_i \cdot v_j \geq 0$ constraints. These are clearly valid constraints (since the standard basis vectors satisfy them), but we might not have included them on our first attempt. Then we wouldn't have been able to round the SDP well (there would have been a large integrality gap), so we would have been stuck.

We will round this SDP using random hyperplanes, as we did with Max-Cut. But instead of using a single random hyperplane, we will use two independent random hyperplanes. Slightly more formally, given a solution to the SDP, we will construct a partition into four parts by choosing

random unit vectors $r_1$ and $r_2$ and defining the following sets:

$$\begin{aligned}
R_1 &= \{i \in V : r_1 \cdot v_i \geq 0,\ r_2 \cdot v_i \geq 0\} \\
R_2 &= \{i \in V : r_1 \cdot v_i \geq 0,\ r_2 \cdot v_i < 0\} \\
R_3 &= \{i \in V : r_1 \cdot v_i < 0,\ r_2 \cdot v_i \geq 0\} \\
R_4 &= \{i \in V : r_1 \cdot v_i < 0,\ r_2 \cdot v_i < 0\}.
\end{aligned}$$

We let $\mathcal{S} = \{R_1, R_2, R_3, R_4\}$.

Let $X_{ij}$ be a random variable which is 1 if vertices $i$ and $j$ end up in the same cluster. We saw in the last lecture that the probability that a single random hyperplane separates $i$ and $j$ is $\frac{\theta_{ij}}{\pi} = \frac{\arccos(v_i \cdot v_j)}{\pi}$. Hence $\mathbf{E}[X_{ij}] = (1 - \frac{1}{\pi}\arccos(v_i \cdot v_j))^2 = (1 - \frac{\theta_{ij}}{\pi})^2$. Let $W$ be the value of the objective function for our partition $\mathcal{S}$. Then

$$W = \sum_{\{i,j\} \in E} \left( w^+(i,j) X_{ij} + w^-(i,j)(1 - X_{ij}) \right),$$

and so by linearity of expectations

$$\begin{aligned}
\mathbf{E}[W] &= \sum_{\{i,j\} \in E} \left( w^+(i,j)\mathbf{E}[X_{ij}] + w^-(i,j)(1 - \mathbf{E}[X_{ij}]) \right) \\
&= \sum_{\{i,j\} \in E} \left( w^+(i,j) \left(1 - \frac{1}{\pi}\theta_{ij}\right)^2 + w^-(i,j) \left(1 - \left(1 - \frac{1}{\pi}\theta_{ij}\right)^2\right)\right)
\end{aligned}$$

It turns out due to trig/calculus that $(1 - \frac{\theta_{ij}}{\pi})^2 \geq \frac{3}{4}\cos(\theta_{ij})$ and that $1 - (1 - \frac{\theta_{ij}}{\pi})^2 \geq \frac{3}{4}(1 - \cos(\theta_{ij}))$, as long as $\theta_{ij} \leq \pi/2$. But $\theta_{ij} \leq \pi/2$ because $v_i \cdot v_j \geq 0$. Hence we have that

$$\begin{aligned}
\mathbf{E}[W] &\geq \sum_{\{i,j\} \in E} \left( w^+(i,j) \left(\frac{3}{4}\cos\theta_{ij}\right) + w^-(i,j) \left(\frac{3}{4}(1 - \cos\theta_{ij})\right)\right) \\
&= \frac{3}{4} \sum_{\{i,j\} \in E} \left( w^+(i,j)(v_i \cdot v_j) + w^-(i,j)(1 - v_i \cdot v_j) \right) \\
&= \frac{3}{4} \cdot OPT_{SDP}
\end{aligned}$$

Thus this is a 3/4-approximation. This algorithm and analysis is due to Swamy [Swa04].

## 24.3 Max-2SAT

Max-2SAT is the following problem:

- Input:
  - $n$ variable $x_1, \ldots, x_n$

  - $m$ CNF clauses $C_1, \ldots, C_m$, each of which has exactly two literals

- Feasible: assignment of T/F to variables
- Objective: maximize number of satisfied constraints

Unlike 3SAT, 2SAT is not NP-hard: there is an easy polynomial-time algorithm to determine whether all clauses can be satisfied. Max-2SAT, on the other hand, *is* NP-hard (or, more formally, the decision problem of whether more than an $\alpha$-fraction of clauses can be satisfied is NP-hard for certain values of $\alpha$, but not for $\alpha = 1$).

Writing a strict quadratic program for Max-2SAT is actually a bit tricky. To see this, suppose that we have a variable $y_i \in \{-1, 1\}$ for each input variable, where assigning $y_i = -1$ corresponds to setting $x_i$ to T and assigning $y_i = 1$ corresponds to setting $x_i$ to F. Consider a clause of the form $x_i \vee \bar{x_j}$. Then assigning $y_i = -1$ and $y_j = 1$ corresponds to a satisfying assignment, but assigning $y_i = 1$ and $y_j = -1$ is not a satisfying assignment. But in a strict quadratic program, where we can only use terms like $y_i y_j$, there is no way of distinguishing between these assignments!

So we can't just think of $y_i = 1$ as a necessarily true or false. Instead, we'll add a new "dummy" variable $y_T \in \{-1, 1\}$, and whatever value $y_T$ gets is what we define as T. This means that for $x_i \vee x_j$ and an assignment $y_T, y_i, y_j$, if the assignment satisfies the clause then

$$\frac{3 + y_i y_T + y_j y_T - y_i y_j}{4} = 1$$

and otherwise it equals 0. If the clause has negated variables (e.g. $x_i \vee \bar{x}_j$), then we just use the same formula but with negated variables corresponding to negative (integer) variables. So for this example, we would negate $y_j$ to get

$$\frac{3 + y_i y_T - y_j y_T + y_i y_j}{4}.$$

This lets us write the following strict quadratic program:

$$\begin{aligned}
\max \quad & \sum_{\text{clauses } x_i \vee x_j} \frac{3 + y_i y_T + y_j y_T - y_i y_j}{4} + \sum_{\text{clauses } x_i \vee \bar{x}_j} \frac{3 + y_i y_T - y_j y_T + y_i y_j}{4} \\
& + \sum_{\text{clauses } \bar{x}_i \vee x_j} \frac{3 - y_i y_T + y_j y_T + y_i y_j}{4} + \sum_{\text{clauses } \bar{x}_i \vee \bar{x}_j} \frac{3 - y_i y_T - y_j y_T + y_i y_j}{4} \\
\text{s.t.} \quad & y_i \in \{-1, 1\} && \forall i \in V \\
& y_T \in \{-1, 1\}
\end{aligned}$$

When we relax this to an SDP, we get the following:

$$\begin{aligned}
\max \quad & \sum_{\text{clauses } x_i \vee x_j} \frac{3 + v_i \cdot v_T + v_j \cdot v_T - v_i \cdot v_j}{4} + \sum_{\text{clauses } x_i \vee \bar{x}_j} \frac{3 + v_i \cdot v_T - v_j \cdot v_T + v_i \cdot v_j}{4} \\
& + \sum_{\text{clauses } \bar{x}_i \vee x_j} \frac{3 - v_i \cdot v_T + v_j \cdot v_T + v_i \cdot v_j}{4} + \sum_{\text{clauses } \bar{x}_i \vee \bar{x}_j} \frac{3 - v_i \cdot v_T - v_j \cdot v_T + v_i \cdot v_j}{4} \\
\text{s.t.} \quad & v_i \cdot v_i = 1 \qquad \forall i \in V \\
& v_i \in \mathbb{R}^n \qquad \forall i \in V \\
& v_T \cdot v_T = 1 \\
& v_T \in \mathbb{R}^n
\end{aligned}$$

We can round this using random hyperplane rounding (again): we choose a random $r$, and we set to true every $x_i$ with $sign(v_i \cdot r) = sign(v_T \cdot r)$ and set to false all $x_i$ with $sign(v_i \cdot r) \neq sign(v_T \cdot r)$. And if $x_i$ is set to true then we set $y_i$ to $-1$ and if we set $x_i$ to false then we set $y_i$ to 1.

To analyze this, let's rewrite the term for each clause. For simplicity, we'll just consider the first type of clauses $x_i \vee x_j$ (the other cases are similar). Then we can rewrite $\frac{3+v_i \cdot v_T + v_j \cdot v_T - v_i \cdot v_j}{4}$ as $\frac{1}{4}((1 + v_i \cdot v_T) + (1 + v_j \cdot v_T) + (1 - v_i \cdot v_j))$. Then all terms look like $(1 \pm v \cdot u)$ for vectors $u$ and $v$. We'll analyze each of these types separately. Recall that $\alpha_{GW} = \inf_{0 \leq \theta \leq \pi} \frac{2\theta}{\pi(1-\cos\theta)}$.

Consider a term $1 - v_i \cdot v_j$ (where possibly either $i$ or $j$ is $T$). Then the contribution to the SDP of this term is $1 - v_i \cdot v_j = 1 - \cos\theta_{ij}$. On the other hand, the probability that $i$ and $j$ are on different sides of the hyperplane is $\theta_{ij}/\pi$. Hence the expected value of this term in the rounded solution is $\mathbf{E}[1 - y_i y_j] = 2\frac{\theta_{ij}}{\pi} \geq \alpha_{GW}(1 - v_i \cdot v_j)$.

Similarly, consider a term $1 + v_i \cdot v_j$. Then the contribution to the SDP is $1 + v_i \cdot v_j = 1 + \cos\theta_{ij}$. In the rounded solution, the expected value is $\mathbf{E}[1 + y_i y_j] = 2(1 - \frac{\theta_{ij}}{\pi})$. Hence the ratio between the integral contribution and the SDP contribution is $\frac{2(1-\theta_{ij}/\pi)}{1+\cos\theta_{ij}} = \frac{2(\pi-\theta_{ij})}{\pi(1+\cos\theta_{ij})}$. If we set $\theta' = \pi - \theta_{ij}$, then by basic trig we get that this is equal to $\frac{2\theta'}{\pi(1-\cos\theta')} \geq \alpha_{GW}$ (since $\cos(\pi - \theta) = -\cos(\theta)$).

Since for every term the expected contribution to our integral solution is at least $\alpha_{GW}$ times the expected contribution to the SDP solution, by linearity of expectations this gives us an $\alpha_{GW}$-approximation.

## References


[Swa04] Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In J. Ian Munro, editor, *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 526–527. SIAM, 2004.