Lecture 17: Approximation algorithm for max-cut

Rajat Mittal

IIT Kanpur

Today we will see the first application of semidefinite programming in terms of giving the best known approximation algorithm for max-cut problem. This algorithm was discovered by Goemans and Williamson in 1990's and it gave rise to various other approximation algorithms using similar techniques. The algorithm and the analysis is not difficult but uses lot of great ideas.

1 Max cut

Given an undirected graph $G = (V, E)$, the max-cut problem asks for the partition $S_1, S_2 \subseteq V$, s.t., the number of edges going from S_1 to S_2 are maximized. Remember that since S_1, S_2 is a partition, so $S_1 \cup S_2 = V$ and $S_1 \cap S_2 = \varnothing$.

Clearly this is an optimization problem. It is known to be NP-hard and so not expected to have a polynomial time algorithm. We are interested in finding an approximation algorithm for this problem.

Many problems in the real world can be formulated as a max-cut instance. For example, suppose the graph gives information about the friendships in a set of students. More concretely, the vertices correspond to students in the class and there is an edge present between two students if they are friends.

The problem is to put them in two different classroom so that the number of friendships (edges) across the two rooms is maximized (one measure to stop copying). This is essentially solving a max-cut problem on the given graph.

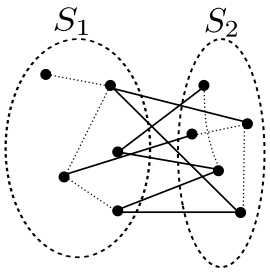

Fig. 1. Example of a cut with edges in the cut shown with solid line

2 Integer programming formulation

An integer program is an optimization problem where variables are constrained to be a set of integers. The max-cut problem can be framed as an integer programming problem.

Given a graph $G = (V, E)$, which has $|V| = n$ vertices and $|E| = m$ edges, introduce variable y_i for every vertex $i \in [n]$. Say $y_i = 1$ if the vertex is assigned to S_1 and $y_i = -1$ if it is assigned to S_2. Then the task is to maximize the number of edges between S_1 and S_2.

If the edge i, j is part of the cut (between S_1 and S_2), $y_i y_j = -1$ otherwise it is $+1$. Then $\sum_{(i,j)\in E} \frac{1-y_i y_j}{2}$ counts the number of edges in the cut. Hence, the following integer program gives us the maximum cut.

$$\begin{aligned} &\max \textstyle\sum_{(i,j)\in E} \frac{1-y_i y_j}{2} \\ \text{s.t.} \quad &y_i \in \{-1, 1\} \quad \forall i \end{aligned} \tag{1}$$

In general, NP hard problems can be formulated as integer programs. Hence, we can't hope to solve integer programs efficiently (in general). The new idea from Goemans and Williamson was to convert this into a semidefinite program and then convert the solution of SDP into an integer solution ($\{-1, +1\}$) again. Similar techniques have been applied to linear programming also.

Exercise 1. Write the maximum independent set problem as an integer program.

3 SDP relaxation and rounding

One of the main ideas of this work was to relax the integer program into an SDP. The relaxation of an optimization program is another optimization program which ideally is easier to solve and and every solution of original program is also a solution of the new program with the same (or related) objective value.

In the case of Eqn. 1, we change the domain of y_i's to be unit vectors instead of integers $\{-1, +1\}$.

$$\begin{aligned} &\max \textstyle\sum_{(i,j)\in E} \frac{1-y_i^T y_j}{2} \\ \text{s.t.} \quad &\|y_i\| = 1 \quad \forall i \end{aligned} \tag{2}$$

The only change is that the y_i's are vectors instead of integers now. It is clear that any solution of Eqn. 1 will still be a solution of Eqn. 2 (one dimensional vectors) with same objective value. Hence the objective value of the SDP is at least the value of the integer program. The SDP is called the relaxation of the integer program.

Since the SDP can be solved in polynomial time (with some precision), we can obtain the solutions of Eqn. 2. The goal now is to convert these vector solutions into integer ($\{-1, +1\}$) solutions. The process of converting vector solution into integers is called *rounding*.

Exercise 2. What kind of rounding techniques can be possible?

This is done by choosing a random hyperplane and if vectors lie on one side (say positive) then they are assigned $+1$ otherwise -1. Hence, the algorithm can be written as,

1. Given a graph G, solve the SDP relaxation, Eqn. 2, for that graph and obtain the solution $\{y_1, \cdots, y_n\}$.
2. Say the vectors y_i belong to $\mathbb{R}^n$. Choose a random vector $v \in \mathbb{R}^n$.
3. If $y_i^T v \geq 0$ then $y'_i = 1$ otherwise $y'_i = -1$.

The new integer assignment y'_i specifies a cut. We know that $\sum_{(i,j)\in E} \frac{1-y_i^T y_j}{2}$ for optimal y is greater than the value of max-cut. If it can be shown that $\sum_{(i,j)\in E} \frac{1-y'_i y'_j}{2}$ is a significant proportion of $\sum_{(i,j)\in E} \frac{1-y_i^T y_j}{2}$, we will get a good approximation algorithm.

Exercise 3. What could be the best possible value of

$$\frac{\sum_{(i,j)\in E} \frac{1-y'_i y'_j}{2}}{\sum_{(i,j)\in E} \frac{1-y_i^T y_j}{2}}$$

4 Analysis of the algorithm

Suppose the optimum value of the SDP is $s = \sum_{(i,j)\in E} \frac{1-y_i^T y_j}{2}$. We will show that the rounded solution $\{y'_1, \cdots, y'_n\}$ has expected value at least cs with $c \approx .8785$. The expectation is taken over random choice of the hyperplane. Since there are n vectors, we can assume they live in $\mathbb{R}^n$.

So the expected value of the integer solution, using linearity of expectation, can be written as,

$$\mathbb{E}_{v\in\mathbb{R}^n} \sum_{(i,j)\in E} \frac{1-y'_i y'_j}{2} = \sum_{(i,j)\in E} \mathbb{E}_{v\in\mathbb{R}^n} \frac{1-y'_i y'_j}{2}.$$

To calculate this, we need the probability that a random v separates vectors y_i and y_j.

$$\mathbb{E}_{v\in\mathbb{R}^n} \frac{1-y'_i y'_j}{2} = Pr_{v\in\mathbb{R}^n}(y'_i \neq y'_j)$$

The angle between vectors y_i and y_j is $\cos^{-1} y_i^T y_j$ (they are unit vectors). Hence the probability that a random v separates them is

$$\frac{\cos^{-1} y_i^T y_j}{\pi}.$$

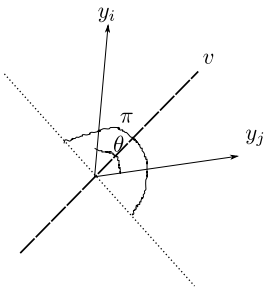

Fig. 2. Random hyperplane separating two vectors

Hence the expected value is

$$\sum_{(i,j)\in E} \frac{\cos^{-1} y_i^T y_j}{\pi}$$

Now use the change of variable $y_i^T y_j = \cos\theta_{ij}$.

$$\begin{aligned}
& \textstyle\sum_{(i,j)\in E} \frac{\cos^{-1} y_i^T y_j}{\pi} \\
= \quad & \textstyle\sum_{(i,j)\in E} \frac{\theta_{ij}}{\pi} \\
= \quad & \textstyle\frac{1}{\pi} \sum_{(i,j)\in E} \frac{\theta_{ij}}{1-\cos\theta ij}(1 - y_i^T y_j) \\
\geq \quad & \left(\min_{0\leq\theta\leq\pi} \frac{2\theta}{\pi(1-\cos\theta)}\right) s \\
\geq \quad & cs
\end{aligned}$$

Exercise 4. Give a lower bound on $\min_{0 \leq \theta \leq \pi} \frac{2\theta}{\pi(1-\cos\theta)}$.

Where using calculus we can show that $c \geq .8785$. Hence,

$$Obj(y_i') \geq c \;\; Obj(y_i) \geq c \;\; max - cut(G).$$

Here $Obj(x)$ denotes the objective value for the solution x and $max - cut(G)$ is the maximum cut in G. The second inequality follows from the fact that SDP is relaxation of original maximum cut integer program. Hence the algorithm given above is a c approximation algorithm for max-cut.

5 Consequences of the algorithm

One consequence from the previous analysis is that the SDP gives a tight bound on the value of the integer program. In general, once an optimization program is relaxed, there is no guarantee about the value of the relaxed program. The feasible set is increased and hence the objective value can be arbitrarily higher (in case of maximization problem) or lower (minimization problems) as compared to the optimum value of the original program.

The *rounding* provides a proof that the relaxed value is comparable to the original value. In the case of max-cut, the SDP value is definitely at least the value of integer program. Rounding shows that the SDP value is less than

$$\frac{1}{c} \;\; Obj(y_i') \leq \frac{1}{c} \;\; max - cut(G).$$

This shows that the SDP is a tight bound on the value of the integer program.

$$max - cut(G) \leq Opt(SDP) \leq \frac{1}{c} \;\; max - cut(G).$$