Goemans and Williamson (1995) — "Algorithm for MAXCUT"

February 14, 2018; rev. July 13, 2022

Kevan Hollbach, Lily Li

The following is adapted from a CSC2429 presentation as well as the lecture notes of Ryan O'Donnell scribed by Franklin Ta.

1 Max-Cut Problem

Let $G = (V, E)$ be a weighted undirected graph with $|V| = n$ and where each edge (i, j) has weight $w_{ij} \geq 0$. The goal of MAXCUT is to find a subset $S \subset V$ which maximizes the sum of the weights of edges crossing the partition $(S, V \backslash S)$. Equivalently, we want to find a function $f : V \to \{0, 1\}$ so as to minimize $\sum_{uv \in E} w_{uv}[f(u) \neq f(v)]$.[1]

1.1 ILP Formuation and 1/2-Approximation

Let $\{x_v\}_{v \in V}$, $\{z_e\}_{e \in E} \in \{0, 1\}$ where x_v encodes the partition that v is in and z_{uv} encodes whether or not the edge uv is cut. Then our ILP is

$$\begin{aligned} \max \sum_{uv \in E} & w_{uv} z_{uv} \\ \text{subject to: } & z_{uv} \leq x_u + x_v, \\ & z_{uv} \leq 2 - (x_u + x_v) \end{aligned}$$

Observe how the inequalities behaves when u and v are in the same partition and when they are in different partitions. If u and v are in the same partition then either $z_{uv} \leq x_u + x_v = 0$ or $z_{uv} \leq 2 - (x_u + x_v) = 0$ (the former when $x_u = x_v = 0$ and the latter when $x_u = x_v = 1$. Thus $z_{uv} = 0$ and the edge uv canot be cut. Only when $x_u \neq x_v$ do we have $z_{uv} \leq x_u + x_v = 1$ and $z_{uv} \leq 2 - (x_u + x_v) = 1$ so z_{uv} can be set to one.

The obvious relaxation sets $x_v, z_e \in [0, 1]$, but this is not a good LP relaxation. If $x_v = 1/2$ for all v then $z_e = 1$ for all e and $OPT_{LP} = 1$ for all graphs.

There is a however a simple 1/2-Approximation: randomly assign $f(v)$ to 0 or 1. Then

$$\begin{aligned} \mathbb{E}\left[\text{size of the cut}\right] &= \mathbb{E}\left[\sum_{uv \in E} w_{uv}[f(u) \neq f(v)]\right] \\ &= \sum_{uv \in E} w_{uv} \Pr\left[f(u) \neq f(v)\right] && \text{(linearity of expectations)} \\ &= \sum_{uv \in E} \frac{w_{uv}}{2} && \text{(u and v randomly assigned)} \\ &= \frac{OPT}{2}. \end{aligned}$$

Why is the last inequality true?[2]

Since Max-Cut is an NP-Complete problem[3], we do not expect a polynomial time algorithm which solves it exactly.

[1]These are Iverson brackets. x is a statement and $[x] = 1$ if and only if x is true otherwise $[x] = 0$.

[2]Remember that we want to find a cut which maximizes the sum of edge-weights crossing this cut. Since the weights are non-negative, the best you could do is to have *every edge* cross the cut.

[3]NP-complete problems are the "hardest" problems in the set of all *non-deterministically polynomial-time decision problems*. This is a notion you will learn about later in the course. As it turns out, many important problems — e.g. SAT, traveling salesman, graph 3-coloring, etc. — are in this class.

1.2 Goemans Williamson Algorithm (Standard Vector Program)

We can formulate the problem as the following quadratic program:

$$\text{Maximize:} \quad \sum_{(i,j)\in E} \frac{w_{ij}(1 - x_i x_j)}{2} \tag{1}$$

$$\text{Subject to:} \quad x_i \in \{-1, +1\}, \text{ for } i \in [n] \tag{2}$$

where x_i is associated with vertex v_i and $x_i x_j = 1$ if and only if v_i and v_j are placed in the same set. Let OPT denote the optimum solution to this quadratic program.

Next we introduce the vector programming relaxation of the above quadratic program:

$$\text{Maximize:} \quad \sum_{(i,j)\in E} \frac{w_{ij}(1 - \mathbf{u_i} \cdot \mathbf{u_j})}{2} \tag{3}$$

$$\text{Subject to:} \quad \|\mathbf{u}_i\|^2 = 1 \text{ and } \mathbf{u}_i \in \mathbb{R}^n, \text{ for } i \in [n]. \tag{4}$$

To see that this is indeed a relaxation, take $\mathbf{u}_i = (x_i, 0, ..., 0)$ for each $i \in [n]$. These $\mathbf{u}_i$'s satisfy the constraints ($\|\mathbf{u}_i\|^2 = 1$ and $\mathbf{u}_i \in \mathbb{R}^n$) and $\mathbf{u_i} \cdot \mathbf{u_j} = x_i x_j$. Thus, if OPT_{VP} denotes the optimum solution to the vector program, then $OPT_{VP} \geq OPT$.

The above vector program is equivalent to the following semidefinite program:

$$\text{Maximize:} \quad \sum_{(i,j)\in E} \frac{w_{ij}(1 - X_{ij})}{2} \tag{5}$$

$$\text{Subject to:} \quad X_{ii} = 1 \text{ for } i \in [n] \text{ and } X \succeq 0 \tag{6}$$

where X has entries X_{ij}; to see that these two forms are equivalent remark that $X \succeq 0$ if and only if $X = U^T U$. If we take the columns of U to be the set of vectors $\{\mathbf{u}_i\}$ of the vector program, then feasible solutions of SDP corresponds to feasible solutions of the vector program and vice versa.

We can solve this SDP in polynomial time and obtain an optimal solution X^*. Cholesky factorize X^* into $(U)^T U$ and let the columns of U, $\mathbf{u}_i \in \mathbb{R}^n$, be the solutions to the vector program. We want to round each $\mathbf{u}_i$ to $x_i \in \{-1, +1\}$. Then the set $\{x_i\}_{i=1}^n$ will be a solution to our original quadratic program. Apply randomized rounding as follows: pick $\mathbf{r} = (r_1, ..., r_n)$ by drawing each r_i independently from the distribution $\mathcal{N}(0, 1)$. Then let

$$x_i = \begin{cases} 1 & \mathbf{u}_i \cdot \mathbf{r} \geq 0 \\ -1 & \text{otherwise} \end{cases}.$$

It is helpful to have the geometric picture in mind: each $\mathbf{u}_i$ is a vector which lies on the $(n-1)$-dimensional unit sphere. The hyper-plane with normal $\mathbf{r}$ splits the sphere in-half. All vectors $\mathbf{u}_i$ in the same half of the sphere gets mapped to the same value $c \in \{-1, 1\}$ and all vectors $\mathbf{u}_j$ in the other half gets mapped to $-c$.

To show the constant of approximation, we consider the probability that an edge (i, j) gets cut. This is equivalent to the probability that $\mathbf{u}_i$ and $\mathbf{u}_j$ fall in different halves of the sphere cut by the hyper-plane. Consider the projecting of the normalized vector $\mathbf{r}$ onto the span of $\{\mathbf{u}_i, \mathbf{u}_j\}$. See Figure 1.

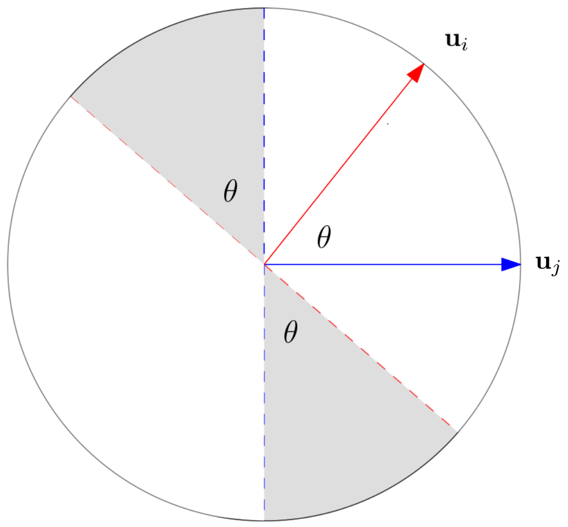

Figure 1: If the normalized $\mathbf{r}$ lies in the shaded region then $\mathbf{u}_i \cdot \mathbf{r}$ and $\mathbf{u}_i \cdot \mathbf{r}$ have different sign.

Thus the probability that $\mathbf{r} \cdot \mathbf{u}_i$ and $\mathbf{r} \cdot \mathbf{u}_j$ have different sign is $\frac{2\theta}{2\pi} = \frac{\theta}{\pi}$. Since $\theta = \arccos(\mathbf{u}_i \cdot \mathbf{u}_j)$,

$$\Pr[(i,j) \text{ is in the cut}] = \frac{\arccos(\mathbf{u}_i \cdot \mathbf{u}_j)}{\pi}. \tag{7}$$

We state without proof that

$$\frac{\arccos(x)}{\pi} \geq 0.878 \left(\frac{1-x}{2} \right) \tag{8}$$

for $x \in [-1,1]$ — it helps to observe that the constant 0.878 approximately minimizes $f(x) = \frac{2\arccos(x)}{\pi(1-x)}$. Thus the expected sum of weights obtained by the algorithm is

$$\begin{aligned}
\mathbb{E}[W] &= \sum_{(i,j)\in E} w_{ij} \Pr[(i,j) \text{ is in the cut}] \\
&= \sum_{(i,j)\in E} w_{ij} \frac{\arccos(\mathbf{u}_i \cdot \mathbf{u}_j)}{\pi} && \text{by 7} \\
&\geq 0.878 \cdot \left(\sum_{(i,j)\in E} w_{ij} \frac{1 - \mathbf{u}_i \cdot \mathbf{u}_j}{2} \right) && \text{by 8} \\
&= 0.878 \cdot OPT_{VP}
\end{aligned}$$

Since the vector program is a relaxation of the original quadratic program, it is the case that $\mathbb{E}[W] \geq 0.878 \cdot OPT_{VP} \geq 0.878 \cdot OPT$. Further, since this algorithm is constructive, the cut found can have value at most OPT so $OPT \geq \mathbb{E}[W] \geq 0.878 \cdot OPT_{VP}$.

Reference

Goemans, M. X. and Williamson, D. P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145.