CS 6550: Randomized Algorithms **Spring 2019**

Lecture 11: Max-Cut Approximation Algorithm via Semidefinite Programming

February 7, 2019

Lecturer: Eric Vigoda *Scribes: Kenny Scharm and Ben Polk*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications.*

11.1 Max-Cut Problem

Given weighted graph $G = (V, E)$ with weights $w(e) > 0$ for $e \in E$, find a cut $(S, \bar{S})$ which maximizes $\sum_{(v,y)\in E, v\in S, y\in \bar{S}}$

11.1.1 Simple $\frac{1}{2}$-approximation Algorithm

For $v \in V$, assign $v = S$ with probability $\frac{1}{2}$ and $y = \bar{S}$ w.p $\frac{1}{2}$
Denote this cut by $Y : V \to \{0, 1\}$ Thus,

$$Y(v) = \begin{cases} +1 & \text{with prob. } \frac{1}{2} \\ 0 & \text{with prob. } \frac{1}{2} \end{cases}$$

$$\begin{aligned}
\mathbb{E}[\text{cut weight}] &= \mathbb{E}[\sum_{(v,z)\in E} w(v,z) Pr(Y(v) \neq Y(z))] \\
&= \sum_{(v,z)\in E} w(v,z) * \frac{1}{2} \\
&= \frac{W}{2} \text{ where } W = \sum_{(v,z)\in E} w(v,z) = \text{total weight of all edges}
\end{aligned}$$

Hence, the expected weight of the cut is $\geq \frac{1}{2}$ of the optimal. As we saw in the last lecture, we can derandomize by the method of conditional expectations [MR]. We will show that we can do better than this in the next section.

11.1.2 Max-Cut as an Integer Linear Programming (ILP) Problem

For each vertex v, create variable $y_v \in \{0, 1\}$.
For each edge (u, v), create variable $z_{uv} \in \{0, 1\}$ ($z_{uv} = 1$ iff $y_u \neq y_v$)
Constraints:

$$y_u + y_V \geq z_{uv} \tag{11.1}$$
$$2 - (y_u + y_v) \geq z_{uv} \tag{11.2}$$
$$\text{if } y_u = y_v = 0, \text{ then } (11.1) \implies z_{uv} = 0 \tag{11.3}$$
$$\text{if } y_u = y_v = 1, \text{ then } (11.2) \implies z_{uv} = 0 \tag{11.4}$$
$$\text{if } y_u \neq y_v, \text{ then } z_{uv} \in \{0, 1\}, \text{but we will maximize it} \tag{11.5}$$

Objective function: $\max \sum_{(u,v)\in E} w(u,v) z_{uv}$ s.t. $\forall (u,v) \in E$

$$\begin{aligned} y_u + y_v &\geq z_{uv} \\ 2 - (y_u + y_v) &\geq z_{uv} \\ z_{uv} &\in \{0,1\} \end{aligned}$$

Consider the LP relaxation by replacing $y \in \{0,1\}$ by $0 \leq y_v \leq 1$ & $z_{uv} \in \{0,1\}$ by $0 \leq z_{uv} \leq 1$. However, this LP is a poor estimate of the ILP.

Set $y_v = \frac{1}{2}$, $\forall v \in V$. Let us define a new objective function W. This is equivalent to doing randomized rounding in the simple random algorithm. Since this does not work, let us try a different method. Instead of $Y : V \to \{0,1\}$, do $Y : V \to \{-1,1\}$. Then,

$$Y(u) \neq Y(v) \iff Y(u)Y(v) = -1$$

$$\frac{1 - Y(u)Y(v)}{2} = \begin{cases} 1 & \text{if } Y(u) \neq Y(v) \\ 0 & \text{if } Y(u) = Y(v) \end{cases}$$

Now we can write the Max-Cut Problem as an Integer Quadratic Program (IQP) with the following objective function:

$$\max \sum_{(i,j)\in E} w(i,j)(\frac{1 - v_i v_j}{2}) \text{ s.t. } \forall i \in V, v_i^2 = 1 \text{ \& } v_i \in \mathbb{R}$$

Even though IQP is NP-hard, we can relax it s.t. each v_i is a unit-vector in $\mathbb{R}^n$ instead of 1 dimension. Thus, $v_i v_j$ becomes the dot-product $v_i \cdot v_j$.

Theorem 11.1 *For $a, b \in \mathbb{R}^n$, $a \cdot b = \sum_{i=1}^{n} a_i b_i = ||a||||b|| \cos\theta$*

In our definition, $||a|| = ||b|| = 1$, so $a \cdot b = \cos\theta$. This definition yields a Semi-definite Program (SDP) which can be solves in polynomial time.

11.2 Semi-definite Programming

Objective function: $\max \sum_{(u,v)\in E} w(u,v) \frac{1 - y_u \cdot y_v}{2}$ s.t. $\forall v \in V, y_v \cdot y_v = 1, y_v \in \mathbb{R}^n$

Because any solution in the IQP corresponds to a feasible point in the SDP, the solution to the SDP must be at least as good as the solution to the IQP. Algorithms exist to find a solution to an SDP in polynomial time, so we can take a solution to this program and "round" it to a feasible point in our IQP. We perform this rounding by selecting a random hyperplane H in $\mathbb{R}^n$ that passes through the origin. We will assign variables in the IQP corresponding to vectors on one side of the hyperplane in the SDP to 1, and variables in the IQP corresponding to vectors on the other side of the hyperplane in the SDP to -1.

We know we want the hyperplane to pass through the origin, so we select a random hyperplane by simply selecting a random unit vector $r \in \mathbb{R}^n$ to be its normal vector. Now for each of our vectors y_i we can set the corresponding variable in the IQP to be $\text{sgn}(r \cdot y_i)$. This is because if $\text{sgn}(r \cdot y_i) > 0$ then y_i and r fall on the same side of the hyperplane and if $\text{sgn}(r \cdot y_i) < 0$ they fall on opposite sides of the hyperplane. Now, imagine two of these vectors y_u and y_v projected into 2-D space. This would look like two vectors coming out of the origin with angle θ between them. If we project H into 2-D, it will look like a line through the origin. This line splits y_u and y_v with probability $\frac{\theta}{\pi}$ which means that the edge (u,v) crosses the cut with this probability. Note that because y_u and y_v are unit vectors $\theta = \cos^{-1}(y_u \cdot y_v)$, so the expected value of the weight of the cut is given as:

$$\mathbb{E}[cutweight] = \sum_{(u,v)\in E} \frac{\cos^{-1}(y_u \cdot y_v) * 2}{\pi}$$

We will now present a lemma that will help us show that this is a 0.87856...- approximation:

Theorem 11.2 *For* $\alpha \approx 0.87856$ *and* $\forall \sigma_{uv} \in [-1, 1]$,

$$\frac{\cos^{-1}(\sigma_{uv})}{\pi} \geq \alpha(\frac{1-\sigma_{uv}}{\pi})$$

The proof will not be included here, but further explanation can be found in this paper [GT]. Recall that the solution for the SDP which we know is at least as good as the true solution is given as:

$$\sum_{(u,v)\in E} w(u,v)\frac{1 - y_u \cdot y_v}{2}$$

As such we can write the following inequality:

$$\sum_{(u,v)\in E} w(u,v)\frac{1 - y_u \cdot y_v}{2} \geq \sum_{(u,v)\in E} \frac{\cos^{-1}(y_u \cdot y_v)2}{\pi} \geq \alpha \sum_{(u,v)\in E} w(u,v)\frac{1 - y_u \cdot y_v}{2}$$

Thus proving that the solution we found is a 0.87856...-approximation of the true max cut.

11.3 Summary

We saw a $\frac{3}{4}$-approximation algorithm for the Max-SAT Problem. For the Max-3SAT Problem, we can use SDP to get a $\frac{7}{8}$-approximation algorithm [KZ]. This is the best possible approximation under the Unique Games Conjecture [GT]. In this lecture, we saw that we can use SDP to get a 0.87856...-approximation algorithm for the Max-Cut Problem.

References

[1] A. Gupta and K. Talwar. Approximating Unique Games. *SODA '06 Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 99–106, 2006.

[2] H. Karloff and U. Zwick. A 7=8-Approximation Algorithm for MAX 3SAT?. *Proc. of 38th FOCS*, pages 406–415, 1997.

[3] S Mahajan and H. Ramesh. Derandomizing approximation algorithms based on semidefinite srogramming. *SIAM Journal on Computing 28*, pages 1641-1663, 1999.