ORIE 6334 Bridging Continuous and Discrete Optimization Oct 9, 2019

Lecture 11

Lecturer: David P. Williamson *Scribe: Devin Smedira*

Recall from the previous lecture that we defined

$$\beta(S) = \min_{(L,R)\ a\ partition\ of\ S} \frac{2|E(L)| + 2|E(R)| + |\delta(S)|}{\mathrm{vol}(S)} = \frac{\mathrm{vol}(S) - 2|\delta(L,R)|}{\mathrm{vol}(S)}$$

and

$$\beta(G) = \min_{\substack{S \subset V \\ S \neq \emptyset}} \beta(S).$$

We also defined β_n to be the smallest eigenvalue of $I + \mathcal{A}$, where $\mathcal{A}$ is the normalized adjacency matrix of G. Last time we showed the following:

Theorem 1 (Trevisan 2009 Lower Bound)

$$\frac{1}{2}\beta_n \leq \beta(G)$$

1 Finishing the Trevisan Inequality

To finish the proof of the Trevisan Inequality, we only need to show the upper bound holds.

Theorem 2 (Trevisan 2009 Upper Bound)

$$\beta(G) \leq \sqrt{2\beta_n}.$$

Proof:

First, assume we can pick a $y \in \mathbb{R}^n$ satisfying

$$\beta_n = \min_y \frac{y^\top (D+A) y}{y^\top y} = \min_y \frac{\Sigma_{(i,j)\in E}(y(i) + y(j))^2}{\Sigma_{i \in V} d(i) y(i)^2}$$

and assume that $\max_i y^2(i) = 1$ (if this is not true, scale y accordingly). Choose $t \in (0,1]$ uniformly at random, and set $x(i) = 1$ if $x(i) \geq \sqrt{t}$, $x(i) = -1$ if $x(i) \leq -\sqrt{t}$ and $x(i) = 0$ otherwise.

Claim 3 $\mathbb{E}[|x(i) + x(j)|] \leq |y(i) + y(j)| \cdot (|y(i)| + |y(j)|)$ *for all* $(i,j) \in E$.

[0]This lecture is derived from Lau's 2012 notes, Week 2, `http://appsrv.cse.cuhk.edu.hk/~chi/csc5160/notes/L02.pdf` and Lau's 2015 notes, Lecture 4, `https://cs.uwaterloo.ca/~lapchi/cs798/notes/L04.pdf`.

Now we know that x generates a

$$L = \{i : x(i) = -1\}, R = \{i : x(i) = 1\}, S = L \cup R$$

where we can show that $\beta(S) \le \sqrt{2\beta_n}$ which implies $\beta(G) \le \sqrt{2\beta_n}$ as follows: Summing over all $(i,j) \in E$ and using the claim and Cauchy-Schwarz gives

$$\begin{aligned}
\mathbb{E}\left[\sum_{(i,j)\in E} |x(i) + x(j)|\right] &\le \sum_{(i,j)\in E} |y(i) + y(j)| \cdot (|y(i)| + |y(j)|) \\
&\le \sqrt{\sum_{(i,j)\in E} (y(i) + y(j))^2}\sqrt{\sum_{(i,j)\in E} (|y(i)| + |y(j)|)^2} \\
&\le \sqrt{\beta_n \sum_{i\in V} d(i) y(i)^2}\sqrt{\sum_{(i,j)\in E} 2(y(i)^2 + y(j)^2)} \\
&= \sqrt{2\beta_n} \sum_{i\in V} d(i) y(i)^2 \\
&= \sqrt{2\beta_n}\mathbb{E}[\sum_{i\in V} d(i)|x(i)|],
\end{aligned}$$

so that there exists a t which generates an $x \in \{-1, 0, 1\}^n$ where

$$\beta(G) \le \frac{\sum_{(i,j)\in E} |x(i) + x(j)|}{\sum_{i\in V} d(i)|x(i)|} \le \sqrt{2\beta_n},$$

as desired. As with the proof of the Cheeger inequality, we can find such an x easily because there are only n possible different vectors x produced by the algorithm, and these correspond to $t = y(i)^2$ for all $i \in V$. $\square$

We return to the proof of the claim.

Proof of claim:

Without loss of generality suppose $y(i)^2 \ge y(j)^2$. If $y(i), y(j)$ have the same sign then

$$\begin{aligned}
&t \le y(i)^2 \le y(j)^2 \to x(i) = -x(j) && \to |x(i) + x(j)| = 0 \\
&y(i)^2 < t \le y(j)^2 \to x(i) = 0, x(j) = 1 && \to |x(i) + x(j)| = 1 \\
&t \ge y(j)^2 \ge y(i)^2 \to x(i) = x(j) = 0 && \to |x(i) + x(j)| = 0
\end{aligned}$$

So we can conclude that

$$\begin{aligned}
\mathbb{E}[|x(i) + x(j)|] &= 1 \cdot \mathbb{P}[y(j)^2 \le t \le y(i)^2] \\
&= y(i)^2 - y(j)^2 \\
&= (y(i) - y(j))(y(i) + y(j)) \\
&\le |y(i) + y(j)| \cdot (|y(i)| + |y(j)|).
\end{aligned}$$

Otherwise, $y(i), y(j)$ have different signs, so

$$\begin{aligned}
&t \le y(j)^2 \le y(i)^2 \to x(i) = x(j) = \pm 1 && \to |x(i) + x(j)| = 2 \\
&y(j)^2 < t \le y(i)^2 \to x(i) = 0, x(j) = \pm 1 && \to |x(i) + x(j)| = 1 \\
&t \ge y(i)^2 \ge y(j)^2 \to x(i) = x(j) = 0 && \to |x(i) + x(j)| = 0
\end{aligned}$$

So we can conclude that

$$\begin{aligned}
\mathbb{E}[|x(i) + x(j)|] &= 1 \cdot \mathbb{P}[y(j)^2 \le t \le y(i)^2] + 2 \cdot \mathbb{P}[t \le y(j)^2)] \\
&= y(i)^2 - y(j)^2 + 2y(j)^2 \\
&= y(j)^2 + y(i)^2 \\
&\le |y(i) + y(j)| \cdot (|y(i)| + |y(j)|),
\end{aligned}$$

as claimed. □

Notice this proof means we can find a L, R, $S = L \cup R$ where

$$\frac{\mathrm{vol}(S) - 2|\delta(L,R)|}{\mathrm{vol}(S)} \le \sqrt{2\beta_n}$$

2 MAX CUT

Next, we develop and and analyze randomized approximation algorithms for MAX CUT. Recall the MAX CUT problem: Given $G = (V, E)$, find $S \subset V$ that maximizes $\delta(S)$.

Definition 1 (MAX CUT) *Given $G = (V, E)$, find $S \subset V$ that maximizes $|\delta(S)|$*

Definition 2 (Approximation algorithm) *A (randomized) α-approximation algorithm runs in (randomized) polynomial time and computes a solution with (expected) value within α of the value of an optimal solution.*

Note that there exists an easy randomized algorithm for MAX CUT: Flip a coin for each $i \in V$ to decide whether or not $i \in S$. Then

$$\mathbb{E}|\delta(S)| = \sum_{(i,j)\in E} \Pr[(i,j) \in S] = \frac{1}{2}|E| \ge \frac{1}{2}\,\mathrm{OPT},$$

where OPT is the value of an optimal solution to MAX CUT on G.

Today, we will show a .529-approximation algorithm due to Trevisan using a combination of this naive randomized algorithm and Trevisan's Cheeger-like inequalities.

3 Trevisan's Algorithm for MAX CUT

The main idea of this algorithm is to trade off between two cases:

- If OPT $< (1-\epsilon)|E|$, then we get an approximation ratio from the naive random algorithm that is better than 1/2.
- If OPT $\geq (1-\epsilon)|E|$, then we can use Trevisan's inequality to get a better bound.

For Max Cut S^*, let $S = V$, $L = S^*$, $R = V - S^*$. Suppose that OPT $\geq (1-\epsilon)|E|$. Then

$$\begin{aligned}\beta(G) \leq \beta(S) &= \frac{2|E(S^*)| + 2|E(V-S^*)| + |\delta(V)|}{\mathrm{vol}(V)} = \frac{2(|E| - |\delta(S^*)|)}{2|E|} \\ &\leq \frac{2(|E| - (1-\epsilon)|E|)}{2|E|} \\ &= \epsilon.\end{aligned}$$

Notice that we know $\frac{1}{2}\beta_n \leq \beta(G)$, so in this case we can infer that $\beta_n \leq 2\epsilon$.

So if $\beta_n > 2\epsilon$, then OPT $< (1-\epsilon)|E|$. So the naive randomized algorithm finds S such that

$$\mathbb{E}\delta(S) = \frac{1}{2}|E| \geq \frac{\mathrm{OPT}}{2(1-\epsilon)}.$$

Thus in this case it is a $\frac{1}{2(1-\epsilon)}$- approximation algorithm.

Now suppose that $\beta_n \leq 2\epsilon$. We can run the algorithm to find a set S and a partition of S into L and R such that $\beta(S)$ is small, namely, at most $\sqrt{2\beta_n} \leq 2\sqrt{\epsilon}$.

Once we have this S, what should we do to find a large cut? In this case, we will attempt to improve our bounds by making some recursive calls. We recurse our Max-Cut algorithm on $V - S$, to find (L', R') that partition $S - V$.

Consider the following two possible cuts of G (presented as partitions on V):

- $(L \cup L', R \cup R')$
- $(L \cup R', R \cup L')$

Notice that every edge in $\delta(S)$ either "stays on the same side", going from L to L' or R to R', or else "crosses sides", going from L to R' or R to L'. That means that one of the above cuts must contain at least 1/2 the edges in $\delta(S)$. We choose that cut.

Call the size of the cut our algorithm finds on G, ALG(G), and the size of the maximum cut in G, OPT(G). Then:

$$\mathrm{ALG}(G) \geq |\delta(L,R)| + \frac{1}{2}|\delta(S)| + \mathrm{ALG}(G-S),$$

and

$$\begin{aligned}\mathrm{OPT}(G) &\leq |E(L)| + |E(R)| + |\delta(L,R)| + |\delta(S)| + \mathrm{OPT}(G-S). \\ &= \frac{1}{2}\,\mathrm{vol}(S) + \frac{1}{2}|\delta(S)| + \mathrm{OPT}(G-S)\end{aligned}$$

Then

$$\begin{aligned}\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} &\geq \min\left\{\frac{|\delta(L,R)|}{\frac{1}{2}\,\mathrm{vol}(S)}, \frac{\frac{1}{2}|\delta(S)|}{\frac{1}{2}|\delta(S)|}, \frac{\mathrm{ALG}(G-S)}{\mathrm{OPT}(G-S)}\right\}. \\ &= \min\left\{\frac{|\delta(L,R)|}{\frac{1}{2}\,\mathrm{vol}(S)}, \frac{\mathrm{ALG}(G-S)}{\mathrm{OPT}(G-S)}\right\}.\end{aligned}$$

Since $\beta_n \leq 2\epsilon$, using Trevisan's inequalities we bound:

$$\begin{aligned}2\sqrt{\epsilon} &\geq \frac{\mathrm{vol}(S) - 2|\delta(L,R)|}{\mathrm{vol}(S)} \\ &= 1 - \frac{2|\delta(L,R)|}{\mathrm{vol}(S)}.\end{aligned}$$

Thus

$$\frac{|\delta(L,R)|}{\frac{1}{2}\,\mathrm{vol}(S)} \geq 1 - 2\sqrt{\epsilon}.$$

So, we can conclude that

$$\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \geq \min\left\{1 - 2\sqrt{\epsilon}, \frac{\mathrm{ALG}(G-S)}{\mathrm{OPT}(G-S)}\right\}.$$

The same must hold true for $G - S$ recursively. But note that for some subgraph of G we consider in some recursive step, it may be possible that $\beta_n \geq 2\epsilon$. Thus we conclude that:

$$\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \geq \min\left\{1 - 2\sqrt{\epsilon}, \frac{1}{2(1-\epsilon)}\right\}.$$

These two expressions are equal for $\epsilon \approx .0554$, at which point the ratio is about .529. So this is a .529-approximation algorithm.[1]

Better analyses were given in Trevisan 2009, which improved the bound to .531, and in Soto 2015, which improved it to .614.

[1] Lau, in his lecture notes, attributes this analysis to Nick Harvey.

4 Discussion

Goemans, W (1995) give a .878-approximation algorithm for MAX CUT by using semidefinite programming (SDP). So why do we care about Trevisan's spectral algorithm?

- Computing eigenvectors is a lot easier (in terms of speed and memory) than solving SDP. (Although, Trevisan's algorithm makes recursive calls that require recomputing new vectors).
- Some experimental results indicate that this algorithm performs better than the SDP-based algorithm (both in time and result quality).
- This method may be more powerful than LP. Kothari, Meka, Raghavendra (STOC 2017) showed you need at least subexponentially-sized LPs to get an integrality gap for MAX CUT greater than $\frac{1}{2}$.

These observations raise some research questions:

- The current bound on the algorithm's performance doesn't seem tight – is it?
- Is there a "one-shot" spectral algorithm, one that doesn't require recursive calls? The recursion makes it hard to analyze the algorithm, and forces recomputation of eigenvectors.
- Can we apply this algorithm to other problems with a similar structure (called 2-CSP)? For instance, the MAX DICUT problem (MAX CUT in directed graphs) and the MAX 2SAT problem have this structure. In the MAX 2SAT problem, we are given n boolean variables $x_1, \ldots, x_n$, and some number of clauses with at most two variables (e.g. $\bar{x}_1$, $x_2 \vee \bar{x}_3$, etc.) The goal is to find a setting of the variables to true or false so as to maximize the total number of satisfied clauses.