CMPUT 675: Approximation Algorithms **Fall 2013**

Lecture 12 (Oct 15, 2013): Approximation via rounding SDP: Max-Cut

Lecturer: Mohammad R. Salavatipour *Scribe: based on older notes*

12.1 Approximation Algorithm for Max-Cut

The next technique we learn is designing approximation algorithms using rounding semidefinite programs. This was first introduced to obtain improved approximation algorithms for the problem of Max-Cut.

input: $G(V, E)$ with weight function $w : E \to \mathbb{Q}^+$
goal: find $S \subset V$ s.t. it maximizes $\sum_{e \in \delta(S)} w(e)$

A trivial $\frac{1}{2}$-approximation is to obtain a random partition of the vertices; i.e. place every vertex $v \in V$ into set S with probability 1/2. We get $\mathbb{E}(\text{weight of cut}) = \frac{1}{2}\sum_{e \in E} w(e) \geq \frac{1}{2}\text{opt}$ and thus we obtain a $\frac{1}{2}$-approximation.

Every known LP relaxation of Max-Cut has an integrality gap of 2. Thus, to improve beyond the trivial algorithm we need more powerful techniques.

12.1.1 Semidefinite programming

A quadratic program is the problem of optimizing a quadratic function of variables subject to a set of quadratic constraints. Quadratic programs are difficult in general and we don't know how to solve them. A strict quadratic program is a special case in which each of the constraints and objective functions consist of only degree two or zero monomials. Let X be a symmetric $n \times n$ real matrix.

Definition 1 *We say X is positive semidefinite (p.s.d.) if and only if $\forall a \in \mathbb{R}^n : a^T X a \geq 0$.*

It is straightforward to prove the following theorem (e.g. see Vazirani):

Theorem 1 *If X is a symmetric matrix from $\mathbb{R}^{n \times n}$ then the following are equivalent:*

1. *X is p.s.d.*
2. *X has non-negative eigenvalues*
3. *$X = V^T V$ for some $V \in \mathbb{R}^{m \times n}$ $(m \leq n)$.*

Let $\mathbb{M}_n$ be the set of symmetric $n \times n$ real matrices. Let $C, D_1, \ldots, D_k \in \mathbb{M}_n$, $b_1, \ldots, b_k \in \mathbb{R}$. Then we can solve the following semidefinite programm SDP with an additive error of $\epsilon > 0$ in polynomial time in n and $\log(\frac{1}{\epsilon})$:

$$\begin{aligned} \min / \max \sum_{i,j} C_{ij} X_{ij} \qquad & \text{s.t.} \\ \forall 1 \leq l \leq k \quad \sum_{i,j} D_{ijl} X_{ij} \quad &= \quad b_l, \\ X \succeq 0 \text{ is p.s.d.} & \\ X \in \mathbb{M}_n & \end{aligned}$$

It can be verified that the set of feasible solution to a SDP forms a convex body, i.e. any convex combination of a set of feasible solutions is a feasible solution. SDP's are equivalent to vector programs (defined below). Let $\vec{v_1}, \ldots, \vec{v_n} \in \mathbb{R}^n$. A vector program VP is defined as

$$\begin{aligned} & \min / \max \sum c_{ij}(\vec{v_i} \cdot \vec{v_j}) && \text{s.t.} \\ \forall 1 \leq l \leq k \quad & \sum d_{ijl}(\vec{v_i} \cdot \vec{v_j}) && = \quad b_l, \\ & \vec{v_i} \in \mathbb{R}^n. \end{aligned}$$

To convert VP to SDP we replace $(\vec{v_i} \cdot \vec{v_j})$ by X_{ij} and require that $X > 0$ and symmetric and p.s.d.

Lemma 1 *VP and the corresponding SDP are equivalent, i.e. each feasible solution of VP corresponds to a feasible solution of SDP with the same objective value and vice versa.*

Proof. Let $\vec{a_1}, \ldots, \vec{a_n}$ be a solution to VP. Let

$$W = \begin{pmatrix} | & & | \\ a_1 & \cdots & a_n \\ | & & | \end{pmatrix}.$$

Then $X = W^T W$ is a feasible solution to the SDP. The proof of other direction is similar. ■

This theorem implies, that vector problems can be solved in polynomial time, too.

We can formulate Max-Cut using semidefinite programming as follows. Each vertex i is assigned a variable $y_i \in \{1, -1\}$ that indicates whether $i \in S$ or $i \in \bar{S}$, respectively. Then

$$1 - y_i y_j = \begin{cases} 2, & i \in S, j \in \bar{S} \\ 2, & i \in \bar{S}, j \in S \\ 0, & \text{o.w.} \end{cases}.$$

Therefore we can write Max-Cut as follows:

$$\begin{aligned} \max \frac{1}{2} \sum w_{ij}(1 - y_i y_j) & \\ y_i^2 = 1 & \\ y_i \in \mathbb{Z}. & \end{aligned}$$

Note that $y_i = +1$ or y_i=-1, specifies whether i is in S or $\overline{S}$, respectively. Considering the contribution of every edge to $(1 - y_i y_j)$ in the objective function value, it is easy to see that MC models Max-Cut. Next, we relax MC to a vector program (VP):

$$\begin{aligned} \max \quad & \tfrac{1}{2} \textstyle\sum_{i<j} w_{ij}(1 - \vec{v_i}\vec{v_j}) && (VP) \\ & \vec{v_i} \cdot \vec{v_j} = 1, && \forall i \\ & \vec{v_i} \in \mathbb{R}, && \forall i \end{aligned}$$

Given a feasible solution to MC; for any variable y_i, define the corresponding vector $\vec{v_i} = (y_i, 0, \ldots, 0)$. This set of vectors gives a feasible solution to VP. Let Z_{VP} and Z_{MC} be the value of the objective function of VP and MC, respectively.

Corollary 1 $Z_{VP} \geq Z_{MC}$.

The semidefinite program equivalent to VP is:

$$\begin{array}{lll} \max & \frac{1}{2}\sum_{i<j} w_{ij}(1-y_iy_j) & \\ & y_{ii}^2 = 1, & \forall i \\ & Y \succeq 0 & (Y \text{ is positive semidefinite}) \\ & Y \in M_n & (Y \text{ is symmetric}) \end{array}$$

Since $\vec{v}_i \cdot \vec{v}_i = 1$, each vector lies in the unit sphere in $\mathbb{R}^n$.

Example. Take the input graph to be C_5, with the weight of every edge $w_{ij} = 1$. The optimal solution of Max-Cut (MC) is to take S to be any two nonadjacent vertices, hence $Z_{MC} = 4$. And for (VP), it turns out that all vectors of the optimal solution lie in $\mathbb{R}^2$, and $Z_{VP} = 5 \cdot \frac{1}{2}(1 - \cos(\frac{4\pi}{5}))$.

Define θ_{ij} to be the angle between the two vectors $\vec{v}_i$ and $\vec{v}_j$. Note that $\vec{v}_i \cdot \vec{v}_j = \cos\theta_{ij}$. This suggests that, the larger θ_{ij}, the larger the contribution of the expression $w_{ij}(1 - \vec{v}_i \cdot \vec{v}_j)$ to the objective function will be. The idea will be put to work by randomized rounding.

Vector Rounding for MAX-CUT

1. Solve (VP) to obtain the optimal solution with value Z_{VP}.
2. Choose a random vector $\vec{r}$ (from the normal distribution with mean 0 and standard deviation 1), from the unit sphere S_n.
3. Let $S = \{i | \vec{r} \cdot \vec{v}_i \geq 0\}$.
4. Return S.

Theorem 2 [GW95] *Vector Rounding for Max-Cut is an α_{GW}-approximation for the Max-Cut problem, where $\alpha_{GW} \geq 0.8785$.*

The main step in the proof of the above theorem is the following lemma.

Lemma 2 $\Pr[v_i \text{ and } v_j \text{ are separated by the selection of } r] = \frac{\theta_{ij}}{\pi}$.

Proof. Take two vectors v_i and v_j. Let r' be the vector defined by the projection of r onto the hyperplane defined by v_i and v_j. Now, we have $r = r' + r''$, where r'' is orthogonal to the plane defined by v_i and v_j.

$$\begin{aligned} v_i \cdot r = v_i \cdot (r' + r'') = v_i \cdot r + v_i \cdot r'' = v_i \cdot r' \\ v_j \cdot r = v_j \cdot (r' + r'') = v_j \cdot r + v_j \cdot r'' = v_j \cdot r' \end{aligned}$$

In order to see whether the two vectors are separated by $\vec{r}$, we have to compare the signs of $\vec{v}_i \cdot \vec{r}$ and $\vec{v}_j \cdot \vec{r}$. Let AB be the diameter orthogonal to $\vec{v}_i$, and CD be the diameter orthogonal to $\vec{v}_j$ (see figure 12.1).

Consider the position of r':

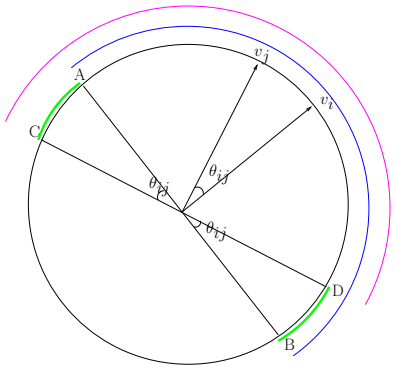

Figure 12.1: The vector $\vec{r}$ will separate v_i and v_j, if it falls in one of the green arcs AC or BD

- If $r' \in Av_iB$, then $r' \cdot v_i \geq 0$
- If $r' \in Cv_jD$, then $r' \cdot v_j \geq 0$

Hence, when r' lies in one of the two arcs AB or CD, the signs of $\vec{v}_i \cdot \vec{r}$ and $\vec{v}_j \cdot \vec{r}$ differ.

$$Pr[v_i \text{ and } v_j \text{ are separated}] = \Pr[r' \text{ lies in one of the arcs } AC \text{ or } BD]$$

Let W be the weight of the cut.

$$E[W] = \sum_{i<j} w_{ij} \cdot \Pr[\ v_i \text{ and } v_j \text{ are separated}] = \sum_{i<j} w_{ij} \cdot \frac{\theta_{ij}}{\pi}$$

Let $\alpha_{GW} = \frac{2}{\pi} \cdot \min_{0 \leq \theta \leq \pi} \frac{\theta}{1-\cos\theta}$; $\alpha_{GW} \geq 0.8785$. For any θ, we have $\frac{\theta}{\pi} \geq \alpha_{GW} \cdot (\frac{1-\cos\theta}{2})$.

$$\begin{aligned} E[W] &\geq \alpha_{GW} \cdot \frac{1}{2} \sum_{i<j} w_{ij}(1 - \cos\theta_{ij}) \\ &= \alpha_{GW} \cdot Z_{VP} \\ &\geq \alpha_{GW} \cdot OPT_{MC}. \end{aligned}$$

■

It is known that the integrality gap of the semidefinite program is almost surely the same as α_{GW}

Theorem 3 [Kar95] *There is an infinite family of graphs, such that* $\frac{E[W]}{Z_{SDP}}$ *tends to* α_{GW}*, as the size of the graph grows.*

The best hardness results known are as follows.

Theorem 4 [Has97] *There is no* α*-approximation for MC with* $\alpha > \frac{16}{17} \approx 0.941$*, unless* P = NP*.*

Theorem 5 [KKMO04] *Assuming the Unique-Games Conjecture (UGC) holds, there is no* $(\alpha_{GW}+\epsilon)$*-approximation for any constant* $\epsilon > 0$*.*

The same technique has been applied to devise an approximation algorithm for the MAX-2SAT problem.

12.2 Approximation Algorithm for for MAX-2SAT

We saw an LP-based approximation algorithm with ratio $\frac{3}{4}$. This approximation ratio can not be beaten using LP-based methods, because of the integrality gap of the corresponding LP. We see an improved algorithm using Semidefinite Programming. The algorithm is similar to that of the previous section for Max-Cut; the main idea is the formulation of the problem as an SDP.

Introduce variables $y_i \in \{+1, -1\}$ for each boolean variable x_i, also introduce an extra variable y_0; $y_i = y_0$, if and only if x_i is true.

For a clause C, its value $V(C) = 1$ if and only if C is satisfied. For clauses of size one, we have:

$$V(x_i) = \frac{1 + y_i y_0}{2}, \quad V(\overline{x_i}) = \frac{1 - y_i y_0}{2}$$

And for clauses of two variables,

$$\begin{aligned} V(x_i x_j) &= 1 - V(1 - \overline{x}_i) \cdot V(\overline{x}_j) \\ &= 1 - \frac{1 - y_i y_0}{2} \cdot \frac{1 - y_j y_0}{2} \\ &= \frac{1}{4}(3 + y_i y_0 + y_j y_0 - y_i y_j y_0^2) \\ &= \frac{1 + y_i y_0}{4} + \frac{1 + y_j y_0}{4} - \frac{1 - y_i y_j}{4} \end{aligned}$$

For any clause C, $V(C)$ is a linear combination of the expressions of the form $1 + y_i y_j$ and $1 - y_i y_j$ (for $1 \le i, j \le n$).

Therefore, the MAX-2SAT problem can be formulated as a quadratic programming as follows, where a_{ij} and b_{ij} are coefficients.

$$\begin{array}{ll} \max & \sum_{i<j}[a_{ij}(1 + y_i y_j) + b_{ij}(1 - y_i y_j)] \quad (QP) \\ & y_i \in \{+1, -1\} \end{array}$$

Similarly, the problem can be formulated as a vector program.

$$\begin{array}{ll} \max & \sum_{i<j}[a_{ij}(1 + \vec{v}_i \cdot \vec{v}_j) + b_{ij}(1 - \vec{v}_i \cdot \vec{v}_j)] \quad (VP) \\ & \vec{v}_i \cdot \vec{v}_i = 1 \\ & \vec{v}_i \in \mathbb{R}^{n+1} \end{array}$$

The algorithm will go like this.

Approximation Algorithm for MAX-2SAT

---

1. Solve VP to obtain a solution Z_{VP}.

2. Take a randm vector $\vec{r} \in S_{n+1}$.
3. Round each y_i to 1, if and only if $\vec{r} \cdot \vec{v}_i \geq 0$.

The analysis of the algorithm is essentially the same as that of MAX-CUT.

Lemma 3 $E[W] \geq \alpha_{GW} \cdot Z_{VP}$.

The best SDP-based approximation algorithm has ratio 0.940.

Theorem 6 **[LLZ02]** *SDP rounding gives* 0.940 *for MAX-2SAT.*

The currently best known hardness of approximation results for the MAX-2SAT problem are as follows:

Theorem 7 **[Has97]** *There is no* 0.9545*-approximation for MAX-2SAT, unless* P = NP.

Theorem 8 **[KKMO'04]** *There is no* 0.943*-approximation for MAX-2SAT, unless the UGC is false.*

References

GW95 M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *Journal of the ACM*,1995, pp. 1115–1145.

Has97 J. Hastad, Some optimal inapproximability results, *Proceedings of the 29th ACM Symposium on Theory of Computing*, 1997, pp. 1–15.

Kar99 H. Karloff, How good is the Goemans-Williamson MAX CUT algorithm, *SIAM Journal on Computing*, 1999, pp. 336–350.

KKMO04 S. Khot, G. Kindler, E. Mossel and R. O'Donnell, Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?, 2004, pp. 146–1546.

LLZ02 M. Lewin, D. Livnat, and U. Zwick, Improved rounding techniques for the MAX-2SAT and MAX DI-CUT problems 2002, pp. 67–82.