Chapter 23

Approximate Max Cut

By Sariel Har-Peled, December 7, 2009[①]

23.1 Problem Statement

Given an undirected graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ and nonnegative weights ω_{ij}, for all $ij \in \mathsf{E}$, the ***maximum cut problem*** (MAX CUT) is that of finding the set of vertices S that maximizes the weight of the edges in the cut $\left(S, \overline{S}\right)$; that is, the weight of the edges with one endpoint in S and the other in $\overline{S}$. For simplicity, we usually set $\omega_{ij} = O$ for $ij \notin \mathsf{E}$ and denote the weight of a cut $\left(S, \overline{S}\right)$ by $w\left(S, \overline{S}\right) = \sum_{i \in S, j \in \overline{S}} \omega_{ij}$.

This problem is NP-Complete, and hard to approximate within a certain constant.

Given a graph with vertex set $\mathsf{V} = \{1, \ldots, n\}$ and nonnegative weights ω_{ij}, the weight of the maximum cut $w(S, \overline{S})$ is given by the following integer quadratic program:

$$\begin{aligned} \text{(Q)} \qquad & \max \quad \frac{1}{2} \sum_{i<j} \omega_{ij}(1 - y_i y_j) \\ & \text{subject to:} \quad y_i \in \{-1, 1\} \qquad \forall i \in \mathsf{V}. \end{aligned}$$

Indeed, set $S = \left\{ i \;\middle|\; y_i = 1 \right\}$. Clearly, $\omega\left(S, \overline{S}\right) = \frac{1}{2} \sum_{i<j} \omega_{ij}(1 - y_i y_j)$.

Solving quadratic integer programming is of course NP-Hard. Thus, we will relax it, by thinking about the numbers y_i as unit vectors in higher dimensional space. If so, the multiplication of the two vectors, is now replaced by dot product. We have:

$$\begin{aligned} \text{(P)} \qquad & \max \quad \gamma = \frac{1}{2} \sum_{i<j} \omega_{ij} \left(1 - \left\langle v_i, v_j \right\rangle\right) \\ & \text{subject to:} \quad v_i \in \mathbb{S}^{(n)} \qquad \forall i \in V, \end{aligned}$$

where $\mathbb{S}^{(n)}$ is the n dimensional unit sphere in $\mathbb{R}^{n+1}$. This is an instance of semi-definite programming, which is a special case of convex programming, which can be solved in polynomial time (solved here means approximated within a factor of $(1+\varepsilon)$ of optimal, for any arbitrarily small $\varepsilon > 0$, in polynomial time). Namely, the solver finds a feasible solution with a the target function being arbitrarily close to the optimal solution. Observe that (P) is a relaxation of (Q), and as such the optimal solution of (P) has value larger than the optimal value of (Q).

The intuition is that vectors that correspond to vertices that should be on one side of the cut, and vertices on the other sides, would have vectors which are faraway from each other in (P). Thus, we compute the optimal solution for (P), and we uniformly generate a random vector $\vec{r}$ on the unit sphere $\mathbb{S}^{(n)}$. This induces a hyperplane h which passes through the origin and is orthogonal to $\vec{r}$. We next assign all the vectors that are on one side of h to S, and the rest to $\overline{S}$.

Summarizing, the algorithm is as follows: First, we solve (P), next, we pick a random vector $\vec{r}$ uniformly on the unit sphere $\mathbb{S}^{(n)}$. Finally, we set

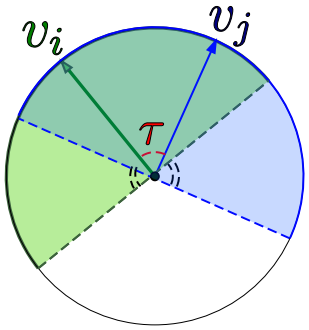
$$S = \left\{ v_i \;\middle|\; \langle v_i, \vec{r} \rangle \geq 0 \right\}.$$

23.1.1 Analysis

The intuition of the above rounding procedure, is that with good probability, vectors in the solution of (P) that have large angle between them would be separated by this cut.

Lemma 23.1.1 *We have* $\mathbf{Pr}\Big[\operatorname{sign}\Big(\langle v_i, \vec{r} \rangle\Big) \neq \operatorname{sign}\Big(\Big\langle v_j, \vec{r} \Big\rangle\Big)\Big] = \dfrac{1}{\pi} \arccos\Big(\Big\langle v_i, v_j \Big\rangle\Big)$.

Proof: Let us think about the vectors v_i, v_j and $\vec{r}$ as being in the plane. To see why this is a reasonable assumption, consider the plane g spanned by v_i and v_j, and observe that for the random events we consider, only the direction of $\vec{r}$ matter, which can be decided by projecting $\vec{r}$ on g, and normalizing it to have length 1. Now, the sphere is symmetric, and as such, sampling $\vec{r}$ randomly from $\mathbb{S}^{(n)}$, projecting it down to g, and then normalizing it, is equivalent to just choosing uniformly a vector from the unit circle.



Now, $\operatorname{sign}(\langle v_i, \vec{r} \rangle) \neq \operatorname{sign}\Big(\Big\langle v_j, \vec{r} \Big\rangle\Big)$ happens only if $\vec{r}$ falls in the double wedge formed by the lines perpendicular to v_i and v_j. The angle of this double wedge is exactly the angle between v_i and v_j. Now, since v_i and v_j are unit vectors, we have $\Big\langle v_i, v_j \Big\rangle = \cos(\tau)$, where $\tau = \angle v_i v_j$. Thus,

$$\mathbf{Pr}\Big[\operatorname{sign}(\langle v_i, \vec{r} \rangle) \neq \operatorname{sign}\Big(\Big\langle v_j, \vec{r} \Big\rangle\Big)\Big] = \frac{2\tau}{2\pi} = \frac{1}{\pi} \cdot \arccos\Big(\Big\langle v_i, v_j \Big\rangle\Big),$$

as claimed. ∎

Theorem 23.1.2 *Let W be the random variable which is the weight of the cut generated by the algorithm. We have*

$$\mathbf{E}[W] = \frac{1}{\pi} \sum_{i<j} \omega_{ij} \arccos\Big(\Big\langle v_i, v_j \Big\rangle\Big).$$

Proof: Let X_{ij} be an indicator variable which is 1 if and only if the edge ij is in the cut. We have

$$\mathbf{E}\Big[X_{ij}\Big] = \mathbf{Pr}\Big[\operatorname{sign}(\langle v_i, \vec{r}\,\rangle) \neq \operatorname{sign}\Big(\Big\langle v_j, \vec{r}\,\Big\rangle\Big)\Big] = \frac{1}{\pi}\arccos\Big(\Big\langle v_i, v_j\Big\rangle\Big),$$

by Lemma 23.1.1. Clearly, $W = \sum_{i<j} \omega_{ij} X_{ij}$, and by linearity of expectation, we have

$$\mathbf{E}[W] = \sum_{i<j} \omega_{ij}\,\mathbf{E}\Big[X_{ij}\Big] = \frac{1}{\pi}\sum_{i<j} \omega_{ij}\arccos\Big(\Big\langle v_i, v_j\Big\rangle\Big).$$

■

Lemma 23.1.3 *For $-1 \le y \le 1$, we have $\dfrac{\arccos(y)}{\pi} \ge \alpha \cdot \dfrac{1}{2}(1-y)$, where $\alpha = \min\limits_{0\le\psi\le\pi} \dfrac{2}{\pi}\dfrac{\psi}{1-\cos(\psi)}$.*

Proof: Set $y = \cos(\psi)$. The inequality now becomes $\frac{\psi}{\pi} \ge \alpha\frac{1}{2}(1-\cos\psi)$. Reorganizing, the inequality becomes $\frac{2}{\pi}\frac{\psi}{1-\cos\psi} \ge \alpha$, which trivially holds by the definition of α. ■

Lemma 23.1.4 $\alpha > 0.87856$.

Proof: Using simple calculus, one can see that α achieves its value for $\psi = 2.331122...$, the nonzero root of $\cos\psi + \psi\sin\psi = 1$. ■

Theorem 23.1.5 *The above algorithm computes in expectation a cut with total weight $\alpha \cdot \mathrm{Opt} \ge 0.87856\mathrm{Opt}$, where* Opt *is the weight of the maximal cut.*

Proof: Consider the optimal solution to (P), and lets its value be $\gamma \ge \mathrm{Opt}$. We have

$$\mathbf{E}[W] = \frac{1}{\pi}\sum_{i<j} \omega_{ij}\arccos\Big(\Big\langle v_i, v_j\Big\rangle\Big) \ge \sum_{i<j} \omega_{ij}\alpha\frac{1}{2}\Big(1 - \Big\langle v_i, v_j\Big\rangle\Big) = \alpha\gamma \ge \alpha\cdot\mathrm{Opt},$$

by Lemma 23.1.3. ■

23.2 Semi-definite programming

Let us define a variable $x_{ij} = \Big\langle v_i, v_j\Big\rangle$, and consider the n by n matrix M formed by those variables, where $x_{ii} = 1$ for $i = 1,\ldots,n$. Let V be the matrix having $v_1,\ldots,v_n$ as its columns. Clearly, $M = V^TV$. In particular, this implies that for any non-zero vector $v \in \mathbb{R}^n$, we have $v^TMv = v^TA^TAv = (Av)^T(Av) \ge 0$. A matrix that has this property, is called ***positive semidefinite***. Interestingly, any positive semidefinite matrix P can be represented as a product of a matrix with its transpose; namely, $P = B^TB$. Furthermore, given such a matrix P of size $n\times n$, we can compute B such that $P = B^TB$ in $O(n^3)$ time. This is know as ***Cholesky decomposition***.

Observe, that if a semidefinite matrix $P = B^TB$ has a diagonal where all the entries are one, then B has columns which are unit vectors. Thus, if we solve (P) and get back a semi-definite matrix, then we can recover the vectors realizing the solution, and use them for the rounding.

In particular, (P) can now be restated as

$$\begin{aligned}
(SD) \qquad \max \quad & \frac{1}{2}\sum_{i<j}\omega_{ij}(1-x_{ij}) \\
\text{subject to:} \quad & x_{ii} = 1 && \text{for } i = 1,\ldots,n \\
& \Big(x_{ij}\Big)_{i=1,\ldots,n,\, j=1,\ldots,n} && \text{is a positive semi-definite matrix.}
\end{aligned}$$

We are trying to find the optimal value of a linear function over a set which is the intersection of linear constraints and the set of positive semi-definite matrices.

Lemma 23.2.1 *Let $\mathcal{U}$ be the set of $n \times n$ positive semidefinite matrices. The set $\mathcal{U}$ is convex.*

Proof: Consider $A, B \in \mathcal{U}$, and observe that for any $t \in [0, 1]$, and vector $v \in \mathbb{R}^n$, we have:

$$v^T\Big(tA + (1-t)B\Big)v = v^T\Big(tAv + (1-t)Bv\Big) = tv^TAv + (1-t)v^TBv \geq 0 + 0 \geq 0,$$

since A and B are positive semidefinite. ∎

Positive semidefinite matrices corresponds to ellipsoids. Indeed, consider the set $x^TAx = 1$: the set of vectors that solve this equation is an ellipsoid. Also, the eigenvalues of a positive semidefinite matrix are all non-negative real numbers. Thus, given a matrix, we can in polynomial time decide if it is positive semidefinite or not (by computing the eigenvalues of the matrix).

Thus, we are trying to optimize a linear function over a convex domain. There is by now machinery to approximately solve those problems to within any additive error in polynomial time. This is done by using the interior point method, or the ellipsoid method. See [BV04, GLS88] for more details. The key ingredient that is required to make these methods work, is the ability to decide in polynomial time, given a solution, whether its feasible or not. As demonstrated above, this can be done in polynomial time.

23.3 Bibliographical Notes

The approximation algorithm presented is from the work of Goemans and Williamson [GW95]. Håstad [Hås01] showed that MAX CUT can not be approximated within a factor of $16/17 \approx 0.941176$. Recently, Khot *et al.* [KKMO04] showed a hardness result that matches the constant of Goemans and Williamson (i.e., one can not approximate it better than α, unless P = NP). However, this relies on two conjectures, the first one is the "Unique Games Conjecture", and the other one is "Majority is Stablest". The "Majority is Stablest" conjecture was recently proved by Mossel *et al.* [MOO05]. However, it is not clear if the "Unique Games Conjecture" is true, see the discussion in [KKMO04].

The work of Goemans and Williamson was very influential and spurred wide research on using SDP for approximation algorithms. For an extension of the MAX CUT problem where negative weights are allowed and relevant references, see the work by Alon and Naor [AN04].

Bibliography

[AN04] N. Alon and A. Naor. Approximating the cut-norm via grothendieck's inequality. In *Proc. 36th Annu. ACM Sympos. Theory Comput.*, pages 72–80, 2004.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge, 2004.

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin Heidelberg, 2nd edition, 1988. 2nd edition 1994.

[GW95] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.*, 42(6):1115–1145, November 1995.

[Hås01] J. Håstad. Some optimal inapproximability results. *J. Assoc. Comput. Mach.*, 48(4):798–859, 2001.

[KKMO04] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for max cut and other 2-variable csps. In *Proc. 45th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 146–154, 2004. To appear in SICOMP.

[MOO05] E. Mossel, R. O'Donnell, and K. Oleszkiewicz. Noise stability of functions with low influences invariance and optimality. In *Proc. 46th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 21–30, 2005.