**CS 810: Introduction to Complexity Theory** 04/03/03

# Lecture 20: Goemans-Williamson MAXCUT Approximation Algorithm

Instructor: Jin-Yi Cai Scribe: Christopher Hudzik, Sarah Knoop

## 1 Overview

First, we outline the Goemans-Williamson Approximation Algorithm for MAXCUT and demonstrate that can be expected to randomly produce a cut that is approximately 88% of the size of the maximum cut. The algorithm depends on nonlinear programming relaxation and semidefinite programming to achieve this result. These techniques have proven quite promising for designing approximation algorithms. The full paper published by Goemans-Williamson in JAMC (1995) can be found at http://www-math.mit.edu/∼goemans, and was used to aid in this write up. The second portion is dedicated to defining the hierarchy of languages that can be decided,with certain probability, by a randomized one-sided error or a randomized two-sided error in p-time TM.

## 2 Goemans-Williamson Approximation Algorithm for MAXCUT

### 2.1 Preliminaries: Defintions & Propositions

Before the algorithm is presented, we'll provided some needed definitions and propositions.

**Definition 1 (positive semidefinite).** *An $n \times n$ matrix, $A$, is* positive semidefinite *if* $\forall x \in \mathbb{R}^n, x^t Ax \geq 0$.

**Definition 2 (semidefinite program).** *A* semidefinite program *is the problem of optimizing a linear function of a symmetric matrix subject to linear equality constriaints and the constraint that the matrix be positive semidefintite.*

It pays to note that a semidefinite program is just a special case of a linear program. Thus it can be solved (using a generalized Simplex method, or other) in polynomial time.

**Proposition 1.** *For a symmetric matrix A, TFAE:*

1. *A is positive semidefinite.*
2. *All eigenvalues of A are non-negative*
3. *There exists a $m \times n$ $(m \leq n)$ matrix $B$ such that $A = B^t B$.*

We also note that finding such a $B$ can be done in p-time using Cholesky decomposition.

### 2.2 The Algorithm

We are given a graph, $G = (V, E)$ of n vertices. For simplicity, we denote $V = \{1, 2, ..., n\}$ and $E = \{(i, j)|$there is an edge between i and j in the graph, $i \leq j$(no double counting)$\}$.
We can view a maximum cut, $M$, for $G$ as the solution to the following integer quadratic program:

$$\text{Maximize } \frac{1}{4}\sum_{(i,j)\in E}(x_i - x_j)^2$$
$$\text{subject to } x_i \in \{-1, 1\} \forall i \in V$$

Equivalently, and useful later:

$$\text{(QP)} \qquad \text{Maximize } \frac{1}{4}\sum_{(i,j)\in E}(x_i^2 - x_j^2 + 2x_i x_j)$$
$$\text{subject to } x_i \in \{-1, 1\} \forall i \in V$$

Here the optimal values for the $x_i$'s correspond to vertex $i$ being on the one side of the cut ($x_i = 1$) or the other ($x_i$ = -1).

Solving QP is NP-complete. We'll relax the constraints of QP to formulate a semidefinite program that will find an approximate maximum to QP. This solution can be decomposed and partitioned to reveal an approximate maximal cut. So, hold on to your horses!!
We'll relax QP as follows:

1. We can view $x_i$ as 1 dimensional (rank 1) unit vectors. Denote these as $y_i$. Now consider $span\{y_1, y_2, ...y_n\}$. These vectors span a space no larger in dimension than $\mathbb{R}^n$. So these vectors can be seen as elements of $\mathbb{R}^n$. So we can restate the quadratic function in QP, using the dot product, as follows:

$$\frac{1}{4}\sum_{(i,j)\in E}(y_i \cdot y_i - y_j \cdot y_j + 2y_i \cdot y_j)$$

2. Form the matrix $A = (a_{ij})$ where $a_{ij} = y_i \cdot y_j$. Now we obtain the following semidefinite program that approximately solves QP.

$$\begin{aligned} \text{(SD)} \quad \text{Maximize} \quad & \frac{1}{4}\sum_{(i,j)\in E}(a_{ii} - a_{jj} + 2a_{ij}) \\ \text{subject to} \quad & a_{ii} = 1, \forall i \in V \\ & A \text{ is symmetric positive semidefinite.} \end{aligned}$$

Now we can proceed to find an approximate maximum cut as follows:

1. Solve SD to obtain an optimal solution, $z$, for SD. Note that the optimal solution for SD could be irrational. In such a case we can obtain, for any $\epsilon > 0$, an approximation $z > z' > z - \epsilon$ in p-time.

2. Say the maximum value of SD, $z$, is produced when the matrix $A^*$ is used. By the proposition above and using Cholesky Decomposition, in p-time we can decompose $A^* = U^t U$. Consider column vectors of the matrix $U = \{u_1, ...u_n\}$. Since $A^*$ is positive semidefinite with $a^*_{ii} = 1$ these vectors are unit vectors: $1 = a^*_{ii} = u_i^t \cdot u_i = ||u_i||^2$.

3. We can imbed these vectors in $\mathbb{R}^n$ (since $m \leq n$). Now we randomly pick a hyperplane through the origin. We can, in p-time, determine which of the vectors $u_i$ have endpoints above or below the plane. The $u_i$'s above will highlight vertices i on one side of the cut and the $u_j$'s below the plane will give verticies on the other side of the cut. Different hyperplanes separating the $u_i$ will produce different cuts, however, any cut made by this proceedure will be guaranteed to have an expected size significantly larger than $\frac{1}{2}$ the size of the maximum cut.

### 2.3 Analysis

Here we attend to the details of why this random algorithm produces a cut of size approximately 88% of the size of the maximum cut. Let $C$ denote the cut produced by this proceedure. We easily establish: $E[|C|] = \sum_{(i,j)\in E} \Pr[(i,j) \text{ is on the cut }]$ and $\Pr[(i,j)$ is on the cut] is the same as the probability that the hyperplane randomly selected separtes the vectors $u_i$ and $u_j$.

**Lemma 1.** *The probability of a random hyperplane separating two vectors is proportional to the angle, $\theta_{ij}$, between the two vectors.*

*Proof.* Let $N$ denote a normal of unit length to the hyperplane selected. Now, saying that $u_i$ and $u_j$ are on opposite sides of this plane is equivalent to $\text{sign}(u_i \cdot N) \neq \text{sign}(u_j \cdot N)$ Thus, $\Pr[\text{hyperplane separates } u_i \text{ and } u_j] = \Pr[\text{sign}(u_i \cdot N) \neq \text{sign}(u_j \cdot N)]$. Using an equivalent statement and by symmetry we have $\Pr[\text{sign}(u_i \cdot N) \neq \text{sign}(u_j \cdot N)] = 2\Pr[v_i \cdot N \geq 0 \text{ and } v_j \cdot N < 0]$ Now the set of all unit length normals, $N$, for which $v_i \cdot N \geq 0$ and $v_j \cdot N < 0$, geometrically is the intersection of two half spaces and the unit sphere in which the dihedral angle between the half spaces is precicely $\theta_{ij}$. This resultant solid has volume: $\frac{\theta_{ij}}{2\pi} *$(volume of the full sphere).

(See http://mathforum.org/dr.math/faq/formulas/faq.sphere.html#lune for 3D picture. Here you can think of $\theta = \theta_{ij}$ and the volume as depicted in light blue)

In other words, the chance that our plane falls between $u_i$ and $u_j$ is the equivalently the chance that our plane lands in this solid. Thus, $Pr[v_i \cdot N \geq 0 \text{ and } v_j \cdot N < 0] = \frac{\theta_{ij}}{2\pi}$.
Thus,

$$\begin{aligned} \Pr[\text{plane is between } u_i \text{ and } u_j] &= \Pr[\text{sign}(u_i \cdot N) \neq sign(u_j \cdot N)] \\ &= 2\Pr[v_i \cdot N \geq 0 \text{ and } v_j \cdot N < 0] \\ &= \frac{\theta_{ij}}{2\pi} \end{aligned}$$

□

So we have

$$\begin{aligned} \mathrm{E}[|C|] &= \sum_{(i,j)\in E} \Pr[\text{plane is between } u_i \text{ and } u_j] \\ &= \sum_{(i,j)\in E} \frac{\theta_{ij}}{(2\pi)} \\ &= \sum_{(i,j)\in E} \frac{4}{\pi} \frac{\theta_{ij}}{(2\sin\frac{\theta_{ij}}{2})^2} \frac{||u_i - u_j||^2}{4} \end{aligned}$$

The last equality holds because $2\sin\frac{\theta_{ij}}{2} = ||u_i - u_j||$ by simple geometry. Using calculus, we can show that the minimum of $\frac{4}{\pi}\frac{\theta_{ij}}{(2\sin\frac{\theta_{ij}}{2})^2}$ for $0 \leq \theta_{ij} \leq \pi$ is greater than .87856. Call the minimum $\alpha$. Thus, we get

$$\mathrm{E}[|C|] \geq \frac{\alpha}{4} \sum_{(i,j)\in E} ||u_i - u_j||^2$$

But $\frac{1}{4}\sum_{(i,j)\in E} ||u_i - u_j||^2$ is the exact size of maximum cut, $|M|$. Therefore, we have an $\alpha$-approximation for MAXCUT.

### 2.4 Closing Notes

It is known that there exists a constant $c < 1$ such that the existence of a $c$-approximation algorithm for MAXCUT (and other NP problems) would imply that P = NP (Arora, Lund, Motwani, Sudan, Szegedy). It has been shown (Bellare, Goldreich, Sudan: 1995) that $c$ is as small as $\frac{83}{84}$ for MAXCUT. Thus, it is not likely that vast improvements can be made on this approximation scheme.

## 3 Probabilistic Complexity Classes

Define BPP, RP, co-RP, ZPP and their hierarchy.

**Definition 3 (Probabilistic Turing Machine, PTM).** *A* PTM $M$ *is a DTM,* $M$*, that takes in two inputs;* $x$ *the regular input, and a random bit string* $\gamma \in \{0,1\}^{p(|x|)}$ *that indicates probabilistic moves of* $M$ *where* $p$ *is some polynomial. The probability that* $M$ *accepts* $x$ *is defined as follows:*

$$P_M(x) = \Pr_{\gamma\in\{0,1\}^{p(|x|)}}[M(x,\gamma) = 1]$$

**Definition 4 (BPP).** *The class* BPP *consists of all languages* $L$ *in which membership in* $L$ *can be checked with two-sided error by a PTM. Formally,* $L \in$ *BPP iff* $\exists$ *a p-time PTM,* $M$*, such that*

$$\begin{aligned} x \in L &\Rightarrow P_M(x) \geq \frac{3}{4} \\ x \notin L &\Rightarrow P_M(x) \leq \frac{1}{4} \end{aligned}$$

**Definition 5 (RP).** *The class* RP *consists of all languages* $L$ *in which membership in* $L$ *can be checked with one-sided error by a PTM. Formally,* $L \in$ *RP iff* $\exists$ *a p-time PTM,* $M$*, such that*

$$\begin{aligned} x \in L &\Rightarrow P_M(x) \geq \frac{1}{2} \\ x \notin L &\Rightarrow P_M(x) = 0 \end{aligned}$$

**Definition 6 (co-RP).** *The class* co-RP *consists of all languages* $L$ *in which membership in* $L$ *can be checked with one-sided error by a PTM. Formally,* $L \in$ *co-RP iff* $\exists$ *a p-time PTM,* $M$*, such that*

$$\begin{aligned} x \in L &\Rightarrow P_M(x) = 1 \\ x \notin L &\Rightarrow P_M(x) \leq \frac{1}{2} \end{aligned}$$

**Definition 7 (ZPP).** ZPP = *RP* $\cap$ *co-RP. Intuitively, A language* $L \in$ *ZPP if there exists a PTM that with high probability will correctly determine membership in L and with small probability will say "I don't know". This small probability can be made exponentially small.*

With these definitions, we have established the relationships among these classes as depicted in Figure 1.

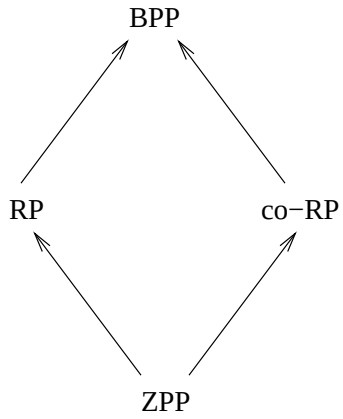


Figure 1: Probabilistic Complexity Class Hierarchy