MATH336 PAPER
THE MAXIMUM CUT PROBLEM AND THE GOEMANS-WILLIAMSON ALGORITHM

AUSTIN ULRIGG

Date: August 1, 2024.

Contents

1. Introduction

Graph theory is believed to have begun in the 18th century with the work of Swiss mathematician Leonhard Euler. In 1735, Euler presented a solution the the famous Könisberg Bridge problem which involved determining if a walk through the city of Königsberg that would cross each of its seven bridges exactly once was possible, which is widely believed to the first theorems of Graph Theory. Euler also discovered the crucial formula

$$V - E + F = 2$$

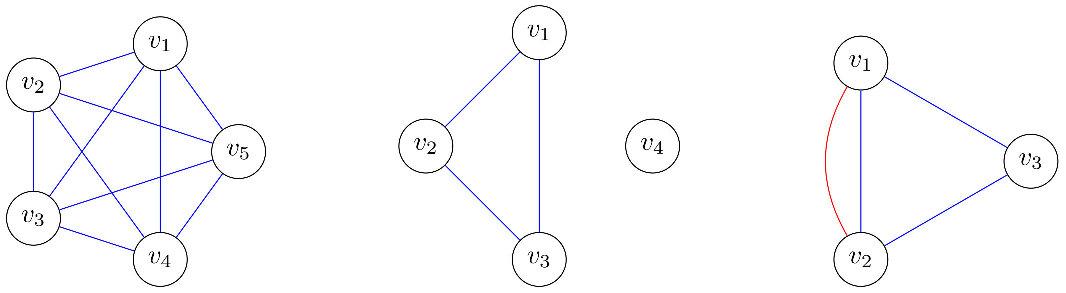
relating the number of vertices, edges, and faces in a convex polyhedron and related it to planar graphs. The constant in this formula is now known as the Euler characteristic of a graph and can be related to the genus of the graph. Euler also introduced many other fundamental ideas in graph theory, like that of a Eulerian path/circuit. In the 20th century, graph theory exploded through the contributions of Hungarian mathematician Paul Erdös. Known for his extensive collaborations, Erdös was one of the most productive mathematicians of all time, publishing over 1,500 papers, so much so that his legacy is remembered in the concept of the "Erdös number," which describes the collaborative distance between an author and Erdös. Erdös contributed much too many fundamental results in graph theory to describe them all here, but importantly for this paper he conjectured the Edwards-Erdös theorem which gives a lower bound on the maximum cut of a graph, and revolutionized much of graph theory through his popularization of the probabilistic method which allows mathematicians to prove the existence of graphs with specific properties without having to explicitly construct any graph. This paper focuses on the Maximum Cut Problem and the Goemans-Williamson Algorithm in graph theory, but we will begin with a list of relevant definitions.

Definition of a Graph. A graph $G(V, E)$ is defined by its vertex set V and edge set E.

- V is a non-empty set of vertices, $V = \{v_1, v_2, ..., v_n\}$.
- E is the edge set, $E = \{e_1, e_2, ..., e_m\}$, where each edge e_k is an unordered pair of vertices (v_i, v_j), indicating a connection between v_i and v_j.

We ensure that our graphs do not contain multiple edges with the same pair of endpoints, so no $e_i = e_j$ where $i \neq j$ (these types of graphs are called multigraphs) and we restrict our graph from having self-loops, where a vertex connects to itself via an edge.

Example Graphs. Here are a few examples of graphs and their respective vertex and edge sets, along with an example of the graphs we are not considering:



Vertex and Edge Sets:

(1) **K5 Graph:**
- Vertex set: $V = \{v_1, v_2, v_3, v_4, v_5\}$
- Edge set: $E = \{(v_1, v_2), (v_1, v_3), (v_1, v_4), (v_1, v_5),$
$(v_2, v_3), (v_2, v_4), (v_2, v_5), (v_3, v_4), (v_3, v_5),$
$(v_4, v_5)\}$

(2) **Triangle Graph with Extra Vertex:**
- Vertex set: $V = \{v_1, v_2, v_3, v_4\}$
- Edge set: $E = \{(v_1, v_2), (v_1, v_3), (v_2, v_3)\}$

(3) **Multigraph:**
- Vertex set: $V = \{v_1, v_2, v_3\}$
- Edge set: $E = \{(v_1, v_2), (v_1, v_2), (v_1, v_3), (v_2, v_3)\}$

A complete graph of n vertices is called K_n, where complete means every vertex is connected to every other vertex via an edge, as in our K_5 example, the complete graph of 5 vertices.

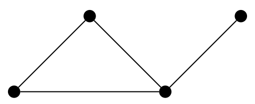
Definition of the Adjacency Matrix of a Graph. The adjacency matrix A_G for a graph G is defined as

$$A_G = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

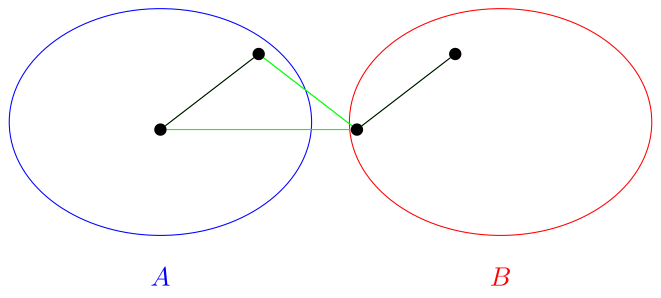
Where a_{ij} being the entry in the i'th row and j'th column of A_G is 1 if there is an edge between v_i and v_j or is 0 if there is not an edge.

2. Maximum Cut Problem

Given a graph $G(V, E)$ (which we will from here on refer to as just G) consider partitioning the vertex set V into two disjoint sets A and B such that $V = A \cup B$, and counting the number of edges $e_k = (v_i, v_j)$ where $v_i \in A$ and $v_j \in B$. For example:



G



For this partition of G we have 2 edges crossing between A and B. Is it possible to have more? For an arbitrary graph G, can we maximize the number of edges that cross from A to B, and if so, can we identify the partition(s) of V that attain the maximum number of crossings? This is known as the maximum cut problem. We say the size of a cut of a graph is the number of edges crossing between A and B and the maximum cut of a graph is a cut with at least the amount of edges as all other cuts.

3. P=NP Conjecture

The maximum cut problem is known to be NP-complete, in fact, it was one of Karp's original 21 NP-complete problems [4]. NP-complete problems are problems such that:

- The problem is a decision problem, i.e, the answer is "yes" or "no".
- If the solution to the problem is "yes" it can be demonstrated by a short polynomial length solution.
- The correctness of each solution can be verified in polynomial time, meaning the running time can be upper bounded by a polynomial expansion in the size of the input for the algorithm.
- The problem can be used to simulate any other problem for which we can verify the solution quickly.

Because of condition 4, solving any one NP-complete problem with a polynomial time algorithm, solves the famous $P = NP$ problem, one of the millennium prize problems selected by the Clay Institute of Mathematics, which conjectures "Can all problems whose solutions can be verified quickly also be quickly solved?" [8]

4. What's to Come

In this paper we will formulate lower bounds for the max cut of a general graph G, and we will find solutions for the max cut on specific classes of graphs such as complete graphs and planar graphs. Additionally, we will present a proof for the current optimal algorithm for finding the max cut of an arbitrary graph, the Goemans-Williamson algorithm, and the implications it and the unique games conjecture have on the $P = NP$ conjecture.

Quadratic Program Reformulation

We first prove a lemma reformulation of the max cut as a quadratic program which will be useful later.

Lemma 4.1

[9] For a graph G with n vertices, and a_{ij} being the entry in the i'th row and j'th column of the adjacency matrix of G,

$$\text{maxcut } G = \frac{1}{4} \max_{x_i \in \{-1,1\}} \sum_{i,j=1}^{n} a_{ij}(1 - x_i x_j)$$

Proof

Proof. Consider any partition of the vertices of G into disjoint vertex sets A and B. Assign all the vertices in A a value of 1 and all of the values in B a value of -1. So we can think of our partition as assigning values of x_i correspondent to each vertex in our graph, and then the sum will return a value of 2 for every edge that crosses the cut, as if v_i and v_j are connected by an edge then $a_{ij} = 1$ and if one lies in A and the other in B then $a_{ij}(1 - x_i x_j) = 2$, however, the (i, j) component of the sum will return the same as the (j, i) component of the sum, so we are quadruple counting all of the edges in the cut correspondent to our partition of vertices into A and B. Maximizing this sum over choices of x_i will then be the same as maximizing choices of placement for vertices into the two vertex sets A and B and it will return $\frac{1}{4} \cdot 4 \cdot \text{maxcut } G$.

5. Lower Bounds

It is easy to find a cut of any graph that attains at least half of the max cut.

Theorem 5.1

[9] For an arbitrary graph G, let $\#E(A, B)$ represent the number of edges in the cut for the partition of $V = A \cup B$. Then, one can find a partition $V = A \cup B$ such that

$$\#E(A, B) \geq \frac{\text{maxcut } G}{2}$$

Proof

Proof. First note the obvious inequalities that $|E| \geq \text{maxcut } G \implies \frac{|E|}{2} \geq \frac{\text{maxcut } G}{2}$.
Now partition the vertices of G with equal probability into disjoint vertex sets A and B. The expected number of edges in the cut after performing this sorting will then be $\frac{|E|}{2} \geq \frac{\text{maxcut } G}{2}$. Thus, simply randomly sorting vertices for an arbitrary graph G gives us a cut with an expected value of $\frac{|E|}{2}$ which implies that there exists a cut for any graph G of size $\frac{|E|}{2}$ or greater, which is at least half of the max cut of G.

Thus, simply randomly sorting vertices we can quickly find a partition with at least half of the maxcut of G.

Theorem 5.2 ▶ Edwards-Erdös

[3]For a graph G with n vertices and m edges

$$\text{maxcut } G \geq \frac{m}{2} + \frac{1}{4}\left(\sqrt{2m + \frac{1}{4}} - \frac{1}{2}\right)$$

However, before proving this result we must first cite a useful lemma.

Lemma 5.1 ▶ Bollobas & Scott

For a connected graph G, let $f(G)$ be the maximal number of edges in a bipartite subgraph of G, i.e, the max cut of G, and let $e(G)$ denote the number of edges of G and $|G|$ the number of vertices of G. Then,

$$f(G) \geq \frac{e(G)}{2} + \frac{|G| - 1}{4}$$

Proof

Proof. To see a proof of this lemma view [1].

Remark. We now use this lemma to prove Theorem 0.2. It was referenced that Lemma 0.2 could be used to prove Theorem 0.2 in Bollobas & Scott but was not proven in the paper or elsewhere, so we provide the proof here.

Proof

Proof. Theorem 0.3 (Edwards-Erdös)
First define $f(m)$ to be the minimum of $f(G)$ over all graphs with m edges, i.e, the minimum of the max cut of graphs G with m edges. Then, for a graph G with n vertices and m edges either G is connected, or it is disconnected. If G is connected then,

$$e(G) \leq \binom{n}{2} = \frac{n!}{2(n-2)!} = \frac{n(n-1)}{2} = \frac{n^2 - n}{2}$$

Since $e(G) = m$ we have from this that

$$2m \leq n^2 - n$$
$$2m \leq (n - \frac{1}{2})^2 - \frac{1}{4}$$
$$2m + \frac{1}{4} \leq (n - \frac{1}{2})^2$$
$$\sqrt{2m + \frac{1}{4}} \leq n - \frac{1}{2}$$
$$\frac{1}{2} + \sqrt{2m + \frac{1}{4}} \leq n = |G|$$

Then, by Lemma 0.4 we have

$$f(G) \geq \frac{e(G)}{2} + \frac{|G| - 1}{4}$$
$$f(G) \geq \frac{m}{2} + \frac{\frac{1}{2} + \sqrt{2m + \frac{1}{4}} - 1}{4}$$

$$f(G) \geq \frac{m}{2} + \frac{1}{4}\left(\sqrt{2m + \frac{1}{4}} - \frac{1}{2}\right)$$

Now because $f(m)$ is the minimum of $f(G)$ for graphs with m edges, since this bound is true for all graphs G with m edges it must also be true for the graph(s) which minimize $f(G)$.

$$f(m) \geq \frac{m}{2} + \frac{1}{4}\left(\sqrt{2m + \frac{1}{4}} - \frac{1}{2}\right)$$

Which completes our proof in the case where G is a connected graph. If G is disconnected then let it have connected components $g_1, \dots g_k$. Then $f(G) = f(g_1) + \cdots + f(g_k)$. Let $e(g_i) = \ell_i$ and $|g_i| = \alpha_i$ then by assumption $f(g_i)$ is connected and by Lemma 0.4 we have

$$f(g_i) \geq \frac{e(g_i)}{2} + \frac{|g_i| - 1}{4}$$

$$e(g_i) \leq \binom{\alpha_i}{2} \leq \frac{\alpha_i!}{2(\alpha_i - 2)!} \leq \frac{\alpha_i(\alpha_i - 1)}{2} \leq \frac{\alpha_i^2 - \alpha_i}{2}$$

This implies that

$$2\ell_i \leq \alpha_i^2 - \alpha_i$$

$$2\ell_i \leq (\alpha_i - \frac{1}{2})^2 - \frac{1}{4}$$

$$\sqrt{2\ell_i + \frac{1}{4}} + \frac{1}{2} \leq \alpha_i = |g_i|$$

By Lemma 0.4 this implies that

$$f(g_i) \geq \frac{\ell_i}{2} + \frac{\sqrt{2\ell_i + \frac{1}{4}} - \frac{1}{2}}{4}$$

For all i. Thus, as $f(G) = f(g_1) + \cdots + f(g_k)$ we have,

$$f(G) \geq \frac{\ell_1}{2} + \frac{\sqrt{2\ell_1 + \frac{1}{4}} - \frac{1}{2}}{4} + \cdots + \frac{\ell_k}{2} + \frac{\sqrt{2\ell_k + \frac{1}{4}} - \frac{1}{2}}{4}$$

$$f(G) \geq \frac{\ell_1 + \cdots + \ell_k}{2} + \frac{\sqrt{2\ell_1 + \frac{1}{4}} - \frac{1}{2} + \cdots + \sqrt{2\ell_k + \frac{1}{4}} - \frac{1}{2}}{4}$$

$$f(G) \geq \frac{m}{2} + \frac{\sqrt{2\ell_1 + \frac{1}{4}} + \sqrt{2\ell_2 + \frac{1}{4}} + \cdots + \sqrt{2\ell_k + \frac{1}{4}} - \frac{k}{2}}{4}$$

Now using the fact that $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ and factoring a k from the numerator we have,

$$f(G) \geq \frac{m}{2} + \frac{k\left(\sqrt{\frac{2\ell_1 + \frac{1}{4} + 2\ell_2 + \frac{1}{4} + \cdots + 2\ell_k + \frac{1}{4}}{k^2}} - \frac{1}{2}\right)}{4}$$

$$f(G) \geq \frac{m}{2} + k\left(\frac{\sqrt{2m + \frac{k}{4}}}{\sqrt{k^2}} - \frac{1}{2}\right) \cdot \frac{1}{4}$$

$$f(G) \geq \frac{m}{2} + \sqrt{2m + \frac{k}{4}} - \frac{k}{2} \geq \frac{m}{2} + \frac{\sqrt{2m + \frac{1}{4}} - \frac{1}{2}}{4}$$

Where the last inequality is true because as a function of k by the first-derivative test we can see that $\frac{m}{2} + \sqrt{2m + \frac{k}{4}} - \frac{k}{2}$ is decreasing for all $k \geq 1$ and thus attains it's maximum for k a positive integer at $k = 1$. And again this bound must hold for $f(m)$.

$$f(m) \geq \frac{m}{2} + \frac{1}{4}\left(\sqrt{2m + \frac{1}{4}} - \frac{1}{2}\right)$$

Remark. The Wikipedia article for the Maximum Cut [11] previously contained an incorrect lower bound citing the Edwards paper [3]:

$$f(m) \geq \frac{m}{2} + \frac{1}{4}\left(\sqrt{2m + \frac{1}{2}} - \frac{1}{4}\right)$$

However, it's clear that this inequality is not valid. Substituting $m = 3$ gives:

$$f(3) \geq \frac{3}{2} + \frac{1}{4}\left(\sqrt{6 + \frac{1}{2}} - \frac{1}{4}\right) \approx 2.07..$$

Since the maximum cut value must be an integer, this implies $f(3) \geq 3$. However, consider the triangle graph, which has $m = 3$ edges but a maximum cut of only 2. The Edwards paper [3] originally presented a correct bound, which was mistranslated on the Wikipedia page. Hence, I have corrected this error on the Wikipedia page.

6. Special Graphs for Max Cut

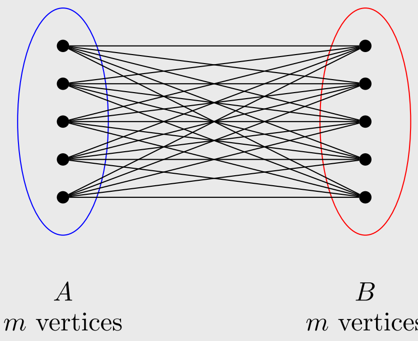
Now that we have proven a lower bound on the max cut of a graph G in general, we consider a few cases of special graphs where the max cut is solved.

Theorem 6.1

For a complete graph K_n the max cut of K_n is m^2 if $n = 2m$ is even, and if $n = 2m + 1$ is odd then the max cut of K_n is $m(m + 1)$.

Proof

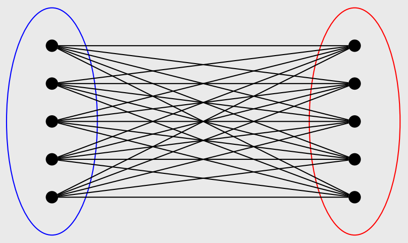
Proof. If $n = 2m \quad m \in \mathbb{N}$ then partition half the vertices into A and half into B. Because the graph is complete, every vertex in A will be connected to every vertex in B, and vice versa, giving m^2 edges in the cut.



If you partition in any other way, then without loss of generality A has $j < m \quad j \in \mathbb{N}$ vertices where B has remaining $2m-j$ vertices, in which case there will be $j \cdot (2m-j)$ edges in the cut. We want to show that $m^2 > j(2m-j)$ to show that the max cut of G is m^2.

$$m^2 > j(2m-j) \iff$$
$$m^2 - 2jm + j^2 > 0 \iff$$
$$(m-j)^2 > 0$$

This is true for all $j \neq m$ i.e for all $j < m$ which completes the proof for complete graphs with an even amount of vertices. If instead $n = 2m+1$ then we place x vertices in A and $(n-x)$ vertices in B, then the number of edges in the cut given by this partition will be $x(n-x)$ where $x, n \in \mathbb{N}$ and we wish to maximize this as a function of x.



A
x vertices

B
$n-x$ vertices

$$f(x) = xn - x^2$$
$$f'(x) = n - 2x$$
$$f''(x) = -2$$

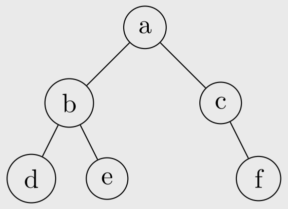
Since we cannot attain a critical point at $n = 2x$ since n is odd, we use the fact that f is concave down, which implies that the nearest maximum we can attain are at the nearest integer values to $\frac{n}{2} = x$, which are $x = m$ and $x = m+1$. In both scenarios, sorting m vertices into A and $2m+1-m = m+1$ into B or $m+1$ into A and $2m+1-m-1 = m$ into B we get $m(m+1)$ edges in the cut, which is maximal as we have shown above, completing the proof.

Theorem 6.2

The max cut for a tree graph with m edges is m.

Proof

Proof. For a tree graph, there are no cycles.



So for any vertex a we can place it without loss of generality in A and place its adjacent vertices in B, and then place the neighbors of those adjacent vertices back into A. We can continue this process, never losing any edges as if any of a's neighbors were adjacent to a vertex connected to a then this would form a cycle. Therefore, we can partition the graph in this fashion to attain a cut with m edges.

Theorem 6.3

The max cut of a planar graph G is solvable in polynomial time. [2]

Proof

Proof. To see a proof of this theorem see [2].

Now that we have proven some theorems about the max cut problem in a general setting we begin with the presentation of the Goemans-Williamson algorithm [10].

7. Goemans-Williamson

The Goemans-Williamson algorithm is an algorithm that recovers at least 87.8..% of the max cut of any graph G in polynomial time. Additionally, this percentage is not as random as it seems, if the unique games conjecture (an unproven conjecture in theoretical computer science) is true, unless $P = NP$ this is the best guaranteed approximation of the max cut by any polynomial time algorithm. [6] We now present a proof and formulation of Goemans's and Williamson's algorithm. Recall Lemma 4.1:

Lemma ▶ 4.1

[9] For a graph G with n vertices, and a_{ij} being the entry in the i'th row and j'th column of the adjacency matrix of G,

$$\text{maxcut } G = \frac{1}{4} \max_{x_i \in \{-1,1\}} \sum_{i,j=1}^{n} a_{ij}(1 - x_i x_j)$$

Since the maximum cut problem is NP-hard, and given its equivalence to this quadratic program, the quadratic program is also NP-hard. However, in an attempt to approximate the maximum cut, we can instead attempt to approximate the maximum of this quadratic program by relaxing it into a semi-definite program. Consider if instead of the x_i in the sum taking on binary values 1 or -1 we allowed x_i to be vectors in $\mathbb{R}^n$ with $||x_i|| = 1$ and instead tried maximizing what we will call SDP G:

$$\text{SDP } G = \frac{1}{4} \max_{||x_i||=1} \sum_{i,j=1}^{n} a_{ij}(1 - \langle x_i, x_j \rangle)$$

As described earlier in the formulation of the quadratic program, any term in the summand is either 2 if $x_i \neq x_j$ and $a_{ij} = 1$, or 0 otherwise. We can recover any non-zero term in this summand in our new relaxed program by choosing $x_i = (1, 0, 0, \ldots, 0)$ and $x_j = (-1, 0, 0, \ldots, 0)$ then $\langle x_i, x_j \rangle = -1$ and $a_{ij}(1 - \langle x_i, x_j \rangle) = 2$. Thus, it is clear that any maximum of the quadratic program can be recovered by this relaxation which is maximizing over a larger space, so we have another lemma:

Lemma 7.1

$$\text{SDP } G \geq \text{maxcut } G$$

Proof

Proof. The proof follows completely from our above reasoning that any maximum value attained by the quadratic program equivalent to the maximum cut can be attained by choices of x_i in SDP G.

$$\text{SDP } G = \frac{1}{4} \max_{||x_i||=1} \sum_{i,j=1}^{n} a_{ij}(1 - \langle x_i, x_j \rangle) \geq \frac{1}{4} \max_{x_i \in \{-1,1\}} \sum_{i,j=1}^{n} a_{ij}(1 - x_i x_j) = \text{maxcut } G$$

The purpose of relaxing our quadratic program was to make it computable in polynomial time, which would be the case if our new program was linear. However, it is not yet clear why SDP G is linear. We introduce a new decision variable p_{ij} for each $i, j \in V$ and we want that $p_{ij} = \langle x_i, x_j \rangle$ in order to make use of this new variable in SDP G. We aim to show that our new program is linear, allowing us to solve it efficiently.

Entries of a symmetric matrix P have the form $p_{ij} = \langle x_i, x_j \rangle$ if and only if $P = X^T X$ for some matrix $X \in R^{V \times V}$ which is equivalently the quality of a matrix being positive semi-definite. This implies that for such matrix P,

$$z^T P z \geq 0 \quad \forall z \in \mathbb{R}^n$$

And, that for any fixed $z \in \mathbb{R}^n$ we have that $\sum_{i,j \in V} p_{ij} z_i z_j \geq 0$, which *is linear* in p_{ij}, our decision variable. Thus, we have a new lemma:

Lemma 7.2

Our relaxed program SDP G is equivalent to,

$$\frac{1}{4} \max_{||x_i||=1} \sum_{i,j=1}^{n} a_{ij}(1 - \langle x_i, x_j \rangle) = \frac{1}{4} \max \sum_{i,j=1}^{n} a_{ij}(1 - p_{ij})$$

Subject to the conditions:

$$\sum_{i,j \in V} p_{ij} z_i z_j \geq 0 \quad \forall z \in \mathbb{R}^n, \qquad p_{ij} = p_{ji}, \qquad p_{ii} = 1 \quad \forall i \in V$$

Where the first two conditions ensure that this is a semi-definite program as P is a symmetric positive semi-definite matrix, and condition three ensures that all $||x_i|| = 1$.

In general, semi-definite programs can be solved in polynomial time with methods like the interior point method and the ellipsoid method which we will not get into the specifics of; but the interested reader can read chapter 13 of Graphs and Geometry by Lovász [7] to learn more.

However, it is important to note that a solution to our semi-definite program will be a list of n vectors in $\mathbb{R}^n$, not a cut of our graph. Surprisingly, the most difficult part of using this relaxation for the max cut is translating the result back into a meaningful cut. The key idea used to "round" each vector in the solution of our SDP back to a binary value 1 or -1 is to use a method called Randomized Hyperplane Rounding. Essentially, rounding vectors in $\mathbb{R}^n$ to 1 or -1 depending on what side of a random hyperplane they land on. This raises the question: What is a random hyperplane in $\mathbb{R}^n$ and how do we choose one? One method is as follows. Sample n independent standard random variables with mean 0 and variance 1, $y_1, \ldots y_n$ and let $\vec{n} = \frac{(y_1, \ldots, y_n)}{||(y_1, \ldots, y_n)||}$ and have $\vec{n}$ be the normal vector defining your random hyperplane in $\mathbb{R}^n$. Now that we have a random hyperplane defined by its normal vector $\vec{n}$ we round our vectors x_i for $i \in V$ that solved our SDP into two vertex sets A and B as follows:

$$A = \{i \in V | \langle x_i, \vec{n} \rangle \geq 0\}$$

$$B = \{i \in V | \langle x_i, \vec{n} \rangle < 0\}$$

Which partitions the vertices according to which side of the hyperplane with normal vector $\vec{n}$ they lie on. Then, Goemans-Williamson algorithm can be stated as follows.

Theorem 7.1

The expected weight of the cut given by partitioning the vertices of G into disjoint sets A and B using the SDP relaxation detailed above and randomized hyperplane rounding is greater than or equal to $0.878 \cdot \text{maxcut } G$.

$$\mathbb{E}[\#E(A, B)] \geq 0.878 \cdot \text{maxcut } G$$

We will present the proof of the theorem after another key lemma.

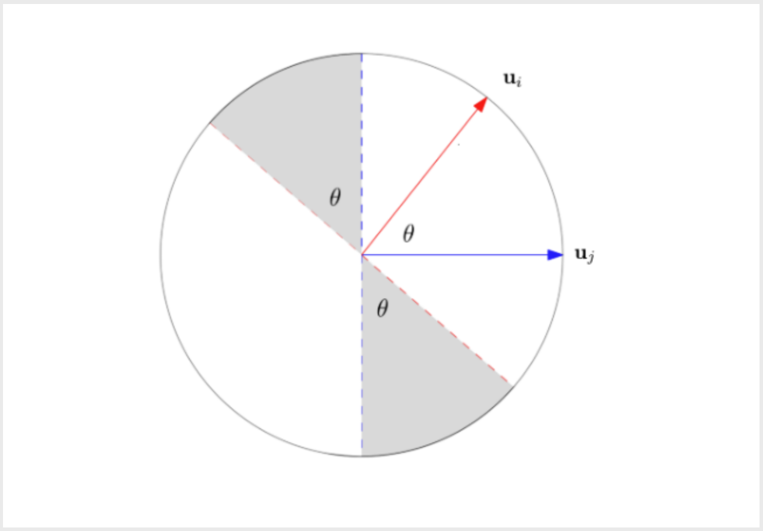
Lemma 7.3

The probability that any edge (i, j) is in the cut is

$$\mathbb{P}\left[(i,j) \text{ in the cut }\right] \geq .878 \cdot \left[\frac{1}{2}(1 - \langle x_i, x_j \rangle)\right]$$

Proof

Proof. Let $(i, j) \in E$ and consider the 2D subspace spanned by x_i and x_j, the solution vectors in $\mathbb{R}^n$ from the SDP relaxation. Vectors x_i and x_j are placed on different sides of the cut if and only if they're split by the projection of $\vec{n}$ onto this subspace. Let θ be the angle between x_i and x_j then one has that the probability edge (i, j) is in the cut is $\frac{\theta}{\pi}$. The following image is provided for clarification.



[5]

Additionally, one has that

$$\langle x_i, x_j \rangle = ||x_i|| ||x_j|| \cos(\theta) = \cos(\theta)$$

Thus,

$$(1 - \langle x_i, x_j \rangle) = (1 - \cos(\theta))$$

We leave out of the proof but point out that it is sufficient to prove that,

$$\mathbb{P}\left[(i,j) \text{ in the cut }\right] = \frac{\theta}{\pi} \geq .878 \cdot [(1 - \cos(\theta))] \quad \forall \theta \in [0, \pi]$$

Now we finally prove Theorem 7.1.

Proof

Proof. First recall that in Lemma 4.1 we noted that the constant factor of 1/4 was due to a quadruple counting of edges in the cut, partially due to that summing over the vertices will return a value of 2 for both edge (i,j) and (j,i) although these are the same edge. Reformulating Lemma 4.1 slightly to instead sum over the edges we have,

$$\mathbb{E}[\#E(A,B)] = \frac{1}{2} \sum_{(i,j)\in E} a_{ij} \mathbb{P}\left[(i,j) \text{ in the cut }\right]$$

Then by linearity of expectation and Lemma's 7.1, and 7.3 it follows that,

$$\begin{aligned}
\mathbb{E}\left[\#E(A,B)\right] &= \frac{1}{2} \sum_{(i,j)\in E} a_{ij} \mathbb{P}[\text{(i,j) in the cut }] \\
&= \frac{1}{2} \sum_{(i,j)\in E} a_{ij} \frac{\theta}{\pi} \\
&\geq .878 \cdot \frac{1}{2} \sum_{(i,j)\in E} (1 - \langle x_i, x_j \rangle) \\
&= .878 \cdot \frac{1}{4} \sum_{i,j=1}^{n} (1 - \langle x_i, x_j \rangle) \\
&= .878 \cdot \text{SDP } G \\
&\geq .878 \cdot \text{maxcut } G
\end{aligned}$$

Which completes the proof.

8. Conclusion

The maximum cut problem is extremely applicable with applications in theoretical computer science, network design, machine learning and even theoretical physics. Math students can benefit from reading this paper due to its exploration of several advanced, powerful, and relatively new problem-solving, including:

- **Randomized Hyperplane Rounding**: The technique used to round solutions of the semi-definite program in Goemans-Williamson's maximum cut algorithm which can be used to relax solutions from other SDP's and provide approximations to many problems.
- **Semi-definite programming (SDPs)**: Semi-definite programming is a powerful generalization of linear programming which is "The most substantial tool in combinatorial optimization in the last 50 years." [7]
- **Probabilistic method**: A non-constructive problem-solving method often used to prove the existence of certain structures within graphs when creating specific examples is extremely difficult.

While its connection to the $P = NP$ conjecture may seem daunting, improvements on the Goemans-Williamson algorithm seem to not be impossible, as algorithms like Burer-Monteiro-Zhang (BMZ) and others have been observed to outperform Goemans-Williamson in practical settings, although the reasons why are currently unknown. Further research can be done on developing algorithms that guarantee cuts of at least $\left(\frac{1}{2} + \varepsilon\right) \cdot \text{maxcut}$, because as we have seen in the proof of our initial algorithm, a method guaranteeing $\left(\frac{1}{2}\right) \cdot \text{maxcut}$ is very easy, but achieving even slight improvements has proven extremely challenging.

References

[1] Bela Bollobas and A. Scott. "Better bounds for Max Cut". In: *Bolyai Soc. Math. Stud.* 10 (Jan. 2002).

[2] Glencora Borradaile. "Lecture 2: MAXCUT for planar graphs". In: (2014).

[3] C. S. Edwards. "Some Extremal Properties of Bipartite Subgraphs". In: *Canadian Journal of Mathematics* 25.3 (1973), pp. 475–485. DOI: 10.4153/CJM-1973-048-x.

[4] Richard M. Karp. "Reducibility Among Combinatorial Problems." In: *Complexity of Computer Computations*. Ed. by Raymond E. Miller and James W. Thatcher. The IBM Research Symposia Series. Plenum Press, New York, 1972, pp. 85–103. ISBN: 0-306-30707-3. URL: http://dblp.uni-trier.de/db/conf/coco/cocc1972.html#Karp72.

[5] Lily Li Kevan Hollbach. "Goemans and Williamson (1995) — "Goemans and Williamson Algorithm for MAXCUT"". In: (2018).

[6] Subhash Khot et al. "Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs?" In: *SIAM Journal on Computing* 37.1 (2007), pp. 319–357. DOI: 10.1137/S0097539705447372. eprint: https://doi.org/10.1137/S0097539705447372. URL: https://doi.org/10.1137/S0097539705447372.

[7] L. Lovász. *Graphs and Geometry*. American Mathematical Society colloquium publications. American Mathematical Society, 2019. ISBN: 9781470453541. URL: https://books.google.com/books?id=0XsOywEACAAJ.

[8] "P versus NP problem". In: *Wikipedia* (2024).

[9] Stefan Steinerberger. "MATH563 Lecture Notes". In: (2024).

[10] Bradley H. Theilman and James B. Aimone. "Goemans-Williamson MAXCUT approximation algorithm on Loihi". In: *Proceedings of the 2023 Annual Neuro-Inspired Computational Elements Conference*. NICE '23. , San Antonio, TX, USA, Association for Computing Machinery, 2023, pp. 1–5. ISBN: 9781450399470. DOI: 10.1145/3584954.3584955. URL: https://doi.org/10.1145/3584954.3584955.

[11] https://en.wikipedia.org/wiki/Maximum_cut.