Unique Games Conjecture and MAX-CUT Application

Xingyou Song

Abstract

In this report, we survey briefly the PCP theorem, as well as a variant useful for 2-CSP problems, which is the Unique Games Conjecture. We provide their statements and intuitions, and show an example of a recent paper on the hardness of approximation of the well known MAX-CUT approximation algorithm, by citing lemmas, including the Majority is Stablest Theorem, related to the Fourier analysis of certain types of cuts on a hypercube-like graph.

Contents

1 PCP AND UGC

1.1 PCP

In the ordinary language of NP, the model consists of a prover and a verifier, (P, V). A language $L \in NP$ iff there is a deterministric verifer V and an arbitrarily powerful prover P, such that two conditions hold:

- Completeness: If $x \in L$, P can write a proof of length poly($|x|$) that V will always accept.
- Soundness: If $x \notin L$, P cannot write a proof of length poly($|x|$) that V will accept.

However, the difference here between NP and a PCP is the following: A PCP (Probabilistically Checkable Proof): The prover P writes down a poly($|x|$) length proof. Then the verifier V does deterministic poly($|x|$) computation ϕ, but is allowed log($|x|$) bits of randomness. Using the randomness, V looks at C bits from the proof, and uses the test ϕ on these C bits and decides to accept or reject.

- Completeness: If $x \in L$, there exists a proof that V will accept with probability 1.
- Soundness: If $x \notin L$, V will always accept with probability at most $1/2$ no matter what the poly($|x|$) length is.

Intuitively, when given a normal proof in NP, there may only be one small, specific error in the proof, that may fool the verifier if he did not look at this error (e.g. a childhood proof using algebra, where dividing a variable, that is actually valued 0, will prove statements like $0 = 1$), which implies the verifier must scan the entire proof.
Comparatively, it is a surprising fact that only looking at small, random bits in a proof, the verifier can suspect if the proof is incorrect. It is proven, from the PCP Theorem, that $\mathbf{NP} = \mathbf{PCP}(O(\log n), O(1))$, proven in [D07]. The gist of this is due to essentially, error correcting codes. One may convert a proof in NP into a proof in PCP by applying "error correction". In this, we show an example by looking at the long codes.
The PCP is useful due to hardness of approximation as well; the PCP theorem also states that:

Theorem 1. *There exists a $\rho < 1$ such that for every $L \in \mathbf{NP}$, there exists a polynomial time function f mapping strings to 3CNF formulas (essentially 3-SAT) such that:*

$$x \in L \implies \text{val}(f(x)) = 1$$

$$x \notin L \implies \text{val}(f(x)) < \rho$$

where val stands for maximum ratio of clauses satisfiable. (The completeness can also be relaxed, as seen in the proof of hardness of approximation of MAX-CUT; the importance is the gap between soundness and completeness). This is due to the fact that we can just construct a PCP that searches for a random clause, and let the maximum value be the probability that the PCP accepts.

Applied to MIN-VERTEX-COVER, this gives a bound on the approximation ratios achieved. Intuitively, this f is the error correction; it transforms a maximization problem into a 3-SAT problem, and encodes a single error into many different positions in the proof uniformly.

Unfortunately, however, the PCP theorem is not strong enough to provide bounds on many optimization algorithms that are equivalent to their performance, given the current state of algorithms. Specifically, it is not enough to solve $2 - CSP$ problems, but is effective on $q - CSP$ problems where $q \geq 3$. Instead, Knot proposed the Unique Games Conjecture for the application on $2 - CSP$ problems, which is still unproven to be true or false.

1.2 Unique Games Conjecture

1.2.1 Statement

The Unique Games Conjecture (UGC) can be stated in two ways.

The first way is to state it in terms of 2-prover games. Consider the game in which we have 2 Provers and 1 Verifier. The verifier asks, from a possible set of questions V_1 to the first prover, and a set of questions V_2 to the second prover. Each prover i, given a question $q_i \in V_i$, will return $L_i(q) = a_i \in A_i$ the set of all possible answers, and answer to the question q, where $L_i : V_i \to A_i$.

Given a tuple (q_1, q_2, a_1, a_2), the verifier then verifies if the tuple is TRUE or FALSE using a verification Γ. i.e. we want to see if $\Gamma(q_1, q_2, a_1, a_2) = TRUE$. We are interested in the maximum probability of success for the provers, in this experiment. The unique games conjecture specifically asks, however, that there is a unique answer a_2 from the second prover given a_1, that will make the verifier state TRUE. Also, $A_1 = A_2$.

In this sense, there exists a permutation function π such that the verifier only accepts if $\pi(a_1) = a_2$.

In the other statement of the conjecture, this is translated to having a graph $G = (V, E)$, in which each edge $(i, j) = e$ has a permutation (or essentially matching) π associated with it. Every vertex is then to be colored from a common label set $[R]$. Note that for any coloring of all of the vertices, some edges may not satisfy correct matchings. We are interested in the maximum possible number of edges satisfied.

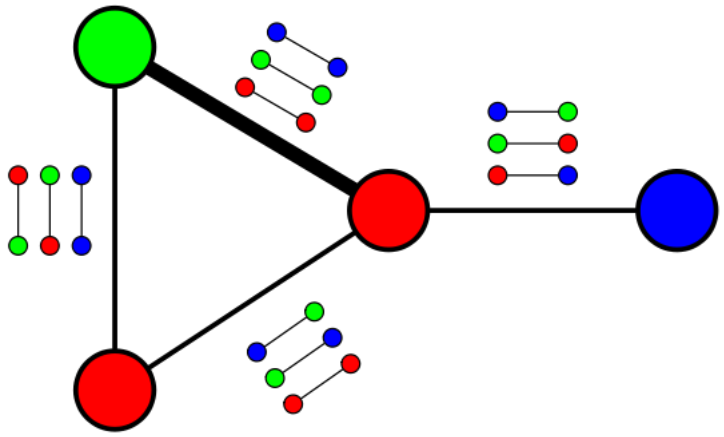

Figure 1: The permutations on edges are shown; given a coloring, the shaded edge cannot be satisfied.

The unique games conjecture states that given constants (γ, η), there exists a large enough R such that it is NP-hard to determine whether the maximum number of edges satisfied to the total number of edges, is at least $1 - \eta$ or at most γ.
Other formulations of UGC may be in terms of equations; e.g. attempting to satisfy the maximum possible number of equations of the form

$$X_i \equiv X_j + c_k \mod R$$

if R is prime. Clearly any variable choice in $\{0, 1, ..., R - 1\}$ implies that the other variable in the equation is chosen uniquely to satisfy the equation.

Note that if $P = NP$ then the UGC is clearly false, since testing whether a solution satisfies some amount of edges is trivial. Therefore proving the UGC is true would require one to first prove $P \neq NP$. Furthermore, it is easy to merely detect if it is possible to satisfy all the edges; just take a vertex, color it in $[R]$, and propagate across over the entire graph due to the bijection assumption across each edge.

1.2.2 Importance

The intuition for what the UGC means is that for a UGC instance, we can encode each variable (or vertex assignment) as a collection of object assignments in a different problem P(e.g. max-cut problem, or vertex label problem, graph coloring, etc.). If there is a good solution to P, then there is a good solution to the UGC instance. Naturally, the edges (or permutations) will represent how well constraints are satisfied within collections of objects in P.

For a cutting problem, we can "blow up" a UGC instance into a cutting instance by representing each vertex X in the UGC by a series of vertices $v_{X,1}, ..., v_{X,k}$. Supposing that we had a cut $(S, V - S)$, this implies we gave a binary k-bit vector to X, and work from there to get a good sequence of colorings in the UGC. This will indeed be our strategy in the proof of the hardness of approximation of MAX-CUT.

Furthermore, the UGC is essentially the proof of the barrier to approximating certain algorithms such as MAX-CUT and MAX 2-SAT. The logic behind disproving the UGC is essentially the idea that there exist more clever algorithms to solve such problems, than mere SDP. A similar barrier existed before the emergence of the SDP method by Goemans-Williamson, in which it was proven that linear programming could only approximate such problems up to 1/2, however, obviously this was broken. The question therefore remains, on whether there really do exist better algorithms than SDP for 2-CSP's.

1.3 State of the Art

Within the past years, work has been done by using parallel repetition to change the large labelling size into more equations involving binary bits, but this has failed, shown in [FKD07].

Recently, there has been a series of SDP algorithms, $\{SDP_1, SDP_2, ..., SDP_i...\}$ that attempt to approximate the UGC problem in n^i time. The best known algorithm, inspired by sum of squares, called the Lasserre SDP Hierarchy in [T15], currently will run in $n^{2^{2\sqrt{\log n}}}$ time, in which there are no current UGC instances that counter it.

2 MAXCUT

In [CST 97], there is a reduction from weighted MAX-CUT to unweighted MAX-CUT, so it makes sense to show only the unweighted case.

2.1 Goeman's Algorithm

Given a graph $G = (V, E)$, we wish to partition the graph into $S, V - S$ such that $E(S, V - S)$ is maximized. In [GW 95], this was solved using SDP.

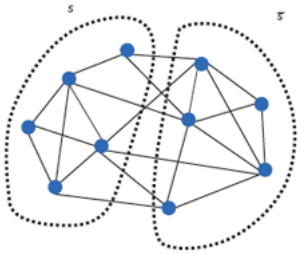

Figure 2: Given a graph as shown above, we partition the vertices into two sets, and count the number of edges in between.

In terms of integer programming, this is equivalent to the following program:

$$OPT(G) = \max \sum_{\{u,v\}\in E} \frac{1}{2}(1 - y_u y_v) \tag{1}$$

where the constraints are $y_i \in \{-1, 1\} \ \ \forall i \in V$. Supposing $|V| = n$, we may relax this programming by replacing y_i with an n dimensional vector $x_i \in \mathbb{R}^n$ on the sphere; i.e. the new semi-definite program,

$$SDP(G) = \max \sum_{\{u,v\}\in E} \frac{1}{2}(1 - \langle x_u, x_v\rangle) \tag{2}$$

where $||x_i||^2 = 1 \ \ \forall i \in V$. Note that $OPT(G) \leq SDP(G)$, because if we had a construction that partitioned the vertices into two sets, then we can allow their respective vectors to be one of two opposite vectors $x, -x$, giving the same values. After $x_1, ..., x_V$ are set, we take a random hyperplane through the origin, then partition the vertices by the side that the x_i are in.
Geometrically, if $\rho = \langle x_u, x_v\rangle$, then the probability that (u, v) is cut is $(\arccos \rho)/\pi$. Also, note that $SDP(G) = \sum_{ij}(\frac{1}{2} - \frac{1}{2}x_i \cdot x_j) \geq OPT(G)$. Taking weighted sums and comparing term by term, the approximation ratio will then be

$$\frac{E}{OPT(G)} \geq \frac{E}{SDP(G)}\alpha_{GW} = \min_{-1\leq\theta\leq 1} \frac{(\arccos\theta)/\pi}{\frac{1}{2} - \frac{1}{2}\theta} = 0.8785... \tag{3}$$

where E is the expected value of edges cut. Thus, this algorithm achieves at least a α_{GW} ratio of the maximum cut in G.

An application of the UGC will show that α_{GW} is the maximum possible ratio achievable in polynomial time. The key is finding the construction such that the ratio holds with a hyperplane cut, which is essentially a graph that can be embedded onto a sphere. The proof of the MAX-CUT hardness of approximation relies on this construction by using a hypercube as an approximation.

2.2 MAX k-CUT, $k > 2$

A similar algorithm was devised for when the number of sets we want is larger than 2. The MAX k-CUT problem is a generalization; we wish to seek a partition $P =$

$\{S_1, ..., S_k\}$ of V such that

$$w(P_k) = \sum_{\{u,v\} \in E} \mathbf{1}[i \neq j | u \in S_i, v \in S_j]$$

is maximized. [FJ 94] proposed a randomized algorithm in which

$$\mathbb{E}(w(P_k)) \geq \alpha_k w(P_k^*)$$

where P^* was the optimal k-cut, where $\alpha_2 = \alpha_{GW}, \alpha_3 \geq 0.800, \alpha_4 \geq 0.85..$ and $\alpha_k = (1 - \frac{1}{k}) + O(2k^{-2} \ln k)$ where clearly $\alpha_k \to 1$ as $k \to \infty$ (note the drop from $k = 2$, but then steadily increasing). The heuristic used is that, instead of using integer programming to describe the program, the SDP consists of having $y_j \in \{a_1, ..., a_k\}$ describe the indicator set belonging. The construction for $\{a_1, ..., a_k\}$ consists of taking the vertices $b_1, ..., b_k$ of the $\triangle^k$ in $\mathbb{R}^{k-1}$; letting the centroid of this simplex be c, and letting $\{a_1, ..., a_k\}$ be the normalized versions of $b_i - c \ \forall i \leq k$.

Then using symmetry,

$$\langle a_i, a_j \rangle := \begin{cases} = -1/(k-1) \text{ if } i \neq j \\ = 1 \text{ if } i = j \end{cases}$$

Then it follows that the formulation will be

$$\max \frac{k-1}{k} \sum_{i<j} w_{i,j}(1 - \langle y_i, y_j \rangle)$$

where $y_i \in \{a_1, ..., a_k\}$.

The natural relaxation of this program will then to relax the locations of a_i to random points, but to maintain the dot product bound, extending the length $\{v_1, ..., v_n\}$. In other words, the SDP will be

$$\max \frac{k-1}{k} \sum_{i<j} w_{i,j}(1 - \langle v_i, v_j \rangle)$$

given that $||v_i||_n = 1 \ \forall i$, and $\langle v_i, v_j \rangle \geq -1/(k-1) \ \forall i \neq j$

The next step, involves choosing random vectors $r_i, ||r_i||_n = 1 \ \forall i \leq k$, and partitioning them according to the closest r_i for each v_j. The analysis of such an algorithm is simulated by using Gaussians due to the fact that all of the vectors lie on the sphere, from which the constants are produced.

2.3 Inapproximability Proof

The general strategy will be to start from a UGC instance U, constructing a graph G, applying a hypothetical max-cut algorithm to get a good cut ratio, which in turn gives a good satisfying solution to U, which contradicts the UGC. (There is an extension to MAX k-CUT that also achieves the exact same constants but we will not cover this here).

Assuming the UGC, we can take an instance of the Unique Label Covering Problem that has either large val (completeness) or low val (soundness). For a large enough label $[R] = \{1, 2..., R\}$, according to the UGC, it is NP-hard to distinguish between those two cases.

- **Completeness:** there exists a proof for which the PCP accepts with large probability c.
- **Soundness:** there is no proof for which it accepts with probability higher than s. This PCP can be reduced to a MAXCUT instance in which a proof corresponds to a cut. If we could efficiently approximate MAXCUT on this graph with a approximation ratio better than s/c, then we would be able to distinguish between cases 1 and 2, contradicting the NP-hardness of UGC.
- **Assumptions**:

  When given a proof with bits $\{-1, 1\}$, this will correspond to a graph with vertices assigned $\{-1, 1\}$ in order to form the cut of the graph.

  For an instance U, with label size R, we will construct a graph G vertices $v_{x,b}$, where $x \in U$, and b spans across all of $\{-1, 1\}^R$. Assume that U is bipartite, and therefore $U = X \cup Y$, where X, Y holds vertices $x_1, ..$ and $y_1...$ respectively. Assume that the vertices on the X side are regular, i.e. have the same degree. (This type of construction, due to the central limit theorem, will eventually be the same as a sphere.)
- **Definitions:** Let ρ be a parameter. Define the "noise"-operator $N_\rho(b)$, on input a $b \in \{-1, 1\}^{|R|}$, outputs a $b' \in \{-1, 1\}^{|R|}$, such that $b'_i = (-1)b_i$ with probability ρ, and $b'_i = b_i$ with probability $1 - \rho$. Also, given a permutation $\pi : [R] \to [R]$, then denote $(b \circ \pi)_i = b_{\pi(i)}$. Furthermore, denote the labelling of a vertex $x \in U$ to be $\ell(x) \in [R]$.
- **Setup:** Now, suppose we weighted the edges of G with the following experiment: randomly select an x_k, with random neighbors $y_i, y_j \in Y$. Suppose π, π' were the permutations on the edges $(x_k, y_i), (x_k, y_j)$. Now, pick a random $b \in \{-1, 1\}^{|R|}$, and receive $b' = N_\rho(b)$. Weight the edge $(v_{y_i, b\circ\pi}, v_{y_j, b'\circ\pi'})$ with the probability of $f_{y_i}(b \circ \pi) \neq f_{y_j}(b' \circ \pi')$. So in some sense, there is higher connectivity between those y_i, y_j's that agree with each other on their neighboring x_k's labellings. (In fact, this is the reason for the construction).

  Given a cut S of graph G, this will assign a -1 or 1 to each vertex $v_{y,b}$. This essentially gives a $2^{|R|}$ bit to each y based on how the vertices $v_{y,b}, b \in \{-1, 1\}^{|R|}$ were selected. This is also representable by a function $f_y : \{-1, 1\}^{|R|} \to \{-1, 1\}$ where

$$f_y := \begin{cases} f_y(b) = 1 \text{ if } b \in S \\ f_y(b) = -1 \text{ if } b \notin S \end{cases}$$

- **PCP:** The PCP will then take a random edge using the distribution above, and output YES if the edge was cut. (Therefore, the PCP probability of verification is dependent on the cut), while the MAX-CUT algorithm will attempt to optimize the total weight of edges cut. The probability of PCP accepting is the value of the weighted MAX-CUT.

- **Remarks:** The intuition here is to be able to measure how well the edges "agree with each other"; i.e. admit correct labellings without having contradictions with each other. This is captured in the notion of "influence" of a function, as well as the information encapsulated by permuting the bits on a string b.

2.3.1 Completeness

We first generate a cut on G based on the labelling of U. If $1-\eta$ fraction of the edges are satisfiable in U, then we can generate a cut on G by assigning $\{-1, 1\}$ to $v_{y,b}$ by using the Long Code of the labelling of x, i.e.

$$v_{y,b} \text{ is marked by } b_{\ell(y)} \in \{-1, 1\}$$

This gives a cut on the graph. The probability that given a random $x \in X$, two neighbors y, y', the edges (x, y), (x, y') are both satisfied is then $(1-\eta)^2 \geq (1-2\eta)$. If the PCP takes a random edge given the distribution above (or essentially experiment), then it expects to have at least $c = (1-2\eta)\rho$ edges cut.

Note: Notice the huge redundancy of the "Long" code here. This error correction method is representative of the PCP-type reductions that are seen.

2.3.2 Soundness

We go the opposite way now; we generate a labelling of U based on the MAXCUT. For a given bit-string to y, we will use this to deduce an optimal labelling $\ell(y) \in [R]$ that will lead to a good value in the UGC game. Essentially we will decode the entire cut string (of size $2^{|R|}|Y|$) in order to give labellings, for the y's, that give good agreements with the corresponding x's connected to those y's.

Suppose there was a cut that went over the limit proposed by Goemans; i.e. suppose there was a cut that cut at least $\frac{1}{\pi}\arccos(1-2\rho) + \epsilon$ where ϵ the excess.

2.4 Fourier Analysis of functions, Stability

In the proof, one needs to look at the spectrum of basic functions and see how much they contribute to the total function. As shown, one candidate function is is the Long Code function; i.e.

$$f_y(b) = b_i$$

if y was labelled $i \in [R]$. However, other functions may exist; i.e. the XOR function;

$$f_y(b) = -b_i \cdot b_j$$

for some i, j. (Here we replace the XOR by a negative multiplication). In order to use this broad spectrum, we use Fourier analysis with respect to the expected value. More formally, consider functions $f : \{-1, 1\}^R \to [-1, 1]$ (here, we are only outputting $-1, 1$ however). Consider the inner product as

$$\langle f, g \rangle_E = \mathbb{E}_{b \in \{-1,1\}^R}[f(b)g(b)]$$

taken uniformly over all such n-bit strings. Then it is known, that

$$f(x) = \sum_{S \subseteq [R]} \hat{f}(S)\chi_S(x)$$

where $\chi_S(x) = \prod_{i \in S} x_i$ (The XOR functions over all subsets) and $\hat{f}(S) = \langle f, \chi_S \rangle_E$. Furthermore, note that $\langle f, f \rangle_E = \sum_S \hat{f}(S)^2$.
For the other notions, define $T_{1-2\rho}(b) = \mathbb{E}[f(N_\rho(b))]$; then the algebra works out to $T_{1-2\rho}(b) = \sum_S (1-2\rho)^{|S|} \hat{f}(S)\chi_S$.
Now define the noise stability with respect to ρ of f:

$$\mathbb{S}_{1-2\rho}(f) = \langle f, T_{1-2\rho} f \rangle = \sum_{S \in [R]} (1-2\rho)^{|S|} \hat{f}(S)^2$$

The intuition here is that smaller $|S|$ allows us to see more clearly the coefficients $\hat{f}(S)$ due to the factor $(1-2\rho)^{|S|}$ that will mitigate $\hat{f}(S)$ with larger $|S|$. When $\rho = 1/2$, the noise operator becomes purely random, which essentially means that there is zero stability.

2.5 Influence, Majority is Stablest

Furthermore, define the influence at i to be

$$\mathrm{Inf}_i(f) = \sum_{i \in S} \hat{f}(S)^2$$

This intuitively calculates how much a single bit location influences the entire function.

A more specific definition consists of the k-degree influence with coordinate i, which is

$$\mathrm{Inf}_i^{\leq k}(f) = \sum_{i \in S,\ |S| \leq k} \hat{f}(S)^2$$

Then the **Majority is Stablest** theorem can be stated as: Given any ϵ and parameter $\rho < 1/2$ (essentially, we want the bits to be more similar to each other), there exists a $\delta = \delta(\epsilon, \rho)$ such that
If

$$\mathbb{E}(f) = 0$$

$$\mathrm{Inf}_i(f) \leq \delta \ \ \forall i \in [R]$$

then

$$\mathbb{S}_{1-2\rho}(f) \leq 1 - \frac{2}{\pi} \arccos(1-2\rho) + \epsilon$$

Intuitively, this states that if the influence from single for variables is small, then the stability (which is basically an amplification on the single variable functions) is small too. The proof involves the central limit theorem, but we will not go into that here.

However, when $\rho > 1/2$, the theorem becomes reversed, and instead, we get that

$$\mathbb{S}_{1-2\rho}(f) \geq 1 - \frac{2}{\pi}\arccos\rho - \epsilon$$

The more important theorem to be used here, will be the case when $\rho > 1/2$. Then there will be small enough $\delta(\epsilon, \rho)$ and large enough $k = k(\epsilon, \rho$ such that if

$$\mathrm{Inf}_i^{\leq k}(f) \leq \delta \quad \forall i$$

then

$$\mathbb{S}_{1-2\rho}(f) \geq 1 - \frac{2}{\pi}\arccos(1-2\rho) - \epsilon$$

The method of attack here now is to assume that there did exist a MAX-CUT algorithm that violates α_{GW} bound; then there are some corresponding f functions belonging to some vertices in U in the ULC that will also violate the $\mathbb{S}_\rho$ bound as shown above, and therefore there will exist an i such that $\mathrm{Inf}_i(f) > \delta$. Then we may find a good labelling with this set.

2.6 Constructing the counter example

It is shown through calculation that

$$\text{Probability of Acceptance} = \mathbb{E}_{y,y'\sim x}\left[\frac{1}{2} - \frac{1}{2}f_y(b\circ\pi_{x,y})f_{y'}(N_\rho(b)\circ\pi_{x,y'})\right]$$

$$= \frac{1}{2} - \frac{1}{2}\mathbb{E}_x[\mathbb{S}_{1-2\rho}(g_x)]$$

where $g_x(b) = \mathbb{E}_{y\sim x}[f_y(b\circ\pi_{x,y})]$. $g(b)$ is essentially, given a bit vector b, taking a random neighbor y of x, and outputting the corresponding partition set.
Now if we have that as well, Probability of Acceptance $\geq (\arccos(1-2\rho))/\pi + \epsilon$, then by Markov's Inequality,

$$\Pr\left[\mathbb{S}_{1-2\rho}(g_x) \leq 1 - \frac{2}{\pi}\arccos(1-2\rho) - \epsilon\right] \geq 1 - \frac{\mathbb{E}_x[\mathbb{S}_{1-2\rho}(g_x)]}{1 - \frac{2}{\pi}\arccos(1-2\rho) - \epsilon} =$$

$$\frac{\epsilon}{1 - \frac{2}{\pi}\arccos(1-2\rho) - \epsilon} \geq \frac{\epsilon}{2}$$

Call these $x \in X$ "good". Due to Majority is Stablest theorem, each g_x has a coordinate j_x with k-degree influence at least δ; label each of these "good" x with $j_x \in [R]$. The motivation for this is that if g_x had a high influence from some coordinate, then the cut partition for x's neighbor's hypercubes will be decided heavily on this coordinate, which implies there are some correlations between the coordinate and the cut. Similarly, other x's with similar neighbors also will have some coordinates that affect the cut value; this implies that in some intuitive notion, labelling these x's will "agree with each other".

More formally with the math, it can be shown that this influence on j_x implies a similar result, in expectation, for the neighbors $y \sim x$ of x. That is,

$$\delta \leq \text{Inf}_{j_x}^{\leq k}(g_x) \leq \mathbb{E}_{y \sim x}[\text{Inf}_{\pi_{x,y}(j_x)}^{\leq k} f_y]$$

Now for each $y \in Y$, denote the candidate set of labels to be:

$$\text{Cand}[y] = \{i \in [R] : \text{Inf}_i^{\leq k}(f_y) \geq \delta/2\}$$

For a neighbor $y' \sim x$ where x was "good", note that $\pi_{x,y}(j_x) \in \text{Cand}[y]$ for at least $\delta/2$ fraction of neighbors $y \sim x$. This is because assuming otherwise, this implies that by definition, we take the cases when the inverse is in the candidate set, with the fact that influence is at most 1, and when the inverse is NOT in the candidate set, which by definition means the influence of the inverse is at most $\delta/2$:

$$\mathbb{E}_{y \sim x}[\text{Inf}_{\pi_{x,y}(j_x)}^{\leq k} f_y] \leq (\delta/2) \cdot 1 + (1 - \delta/2) \cdot (\delta/2) \leq \delta$$

which is a contradiction.

Then having every y pick randomly from its candidate set, and picking randomly any label if its candidate set is empty, the probability is at least the inverse of the cardinality $|\text{Cand}[y]|$. But to bound this, note that $\sum_i \text{Inf}_i^{\leq k} f_y \leq k$ and every element's value in the candidate set is $\geq \delta/2$, which implies $|\text{Cand}[y]| \leq 2k/\delta$.

Then if we run the random experiment in which we pick a random $x \in X$; continue if it is "good", then pick a random neighbor $y \sim x$, continue if it has x in its candidate set; then pick a random label in the candidate set for y, we see that there will be, in expectation, at least $\gamma = (\epsilon/2)(\delta/2)(\delta/2k)$ edges satisfied, which creates a $\delta = \sqrt{8k\gamma/\epsilon}$.

This now contradicts the UGC.

2.7 Finishing the proof

Therefore, we have

$$s = \min_{\rho \in (1/2,1]} \arccos(1 - 2\rho)/\pi + \epsilon$$

and

$$c = \rho(1 - 2\eta)$$

Finishing the proof, we see that as $\eta \to 0, \epsilon \to 0$,

$$\frac{s}{c} \leq \frac{\arccos(1 - 2\rho)/\pi}{\rho} \leq \alpha_{GW}$$

If the algorithm ever did grant a higher ratio, then this would violate the PCP theorem. Furthermore, a similar method, based on decoding longer codes (i.e. not just $1, -1$ assignments) will prove the hardness of approximation for MAX-q-cuts as well, matching with the same exact constants shown in section 2.2, in [KKMO05].
Thus, this shows that the PCP and the UGC, although unproven, have very powerful implications to many NP-HARD approximation problems.

References

[T15] Ning Tan On the Power of Lasserre SDP Hierarchy. 2015

[FKD07] Uriel Feige, Guy Kindler, Ryan O'Donnell. Understanding Parallel Reptition Requires Understanding Foams. 2007

[P08] Prasad Raghavendra. Optimal Algorithms and Inapproximability Results for Every CSP? 2008

[D07] Irit Dinur The PCP Theorem by Gap Amplification 2007

[DM 12] Anindya De, Elchanan Mossel Majority is Stablest: Discrete and SoS 2012.

[FJ 94] Alan Frieze, Mark Jerrum Improved Approximation Algorithms for MAX k-CUT and MAX BISECTION. 1994

[CST 97] Pierlugigi Crescenzi, Riccardo Silvestri, Luca Trevisan On Weighted vs Unweighted Versions of Combinatorial Optimization Problems 1997

[K 02] Subhash Khot On the Power of Unique 2-Prover 1-Round Games 2002

[AB 09] Sanjeev Arora, Boaz Barak Computational Complexity 2009

[RS 10] Prasad Raghavendra, David Steurer Graph Expansion and the Unique Games Conjecture 2010

[KKMO05] Subhash Khot, Guy Kindler, Elchanan Mossel, Ryan O'Donnell. Optimal Inapproximability Results for MAX-CUT, and Other 2-Variable CSPs? 2005

[L04] Luca Trevisan Inapproximability of Combinatorial Optimization Problems 2004

[BPS 10] Anne Berry, Romain Pogorelcnik, and Genevieve Simonet. An Introduction to Clique Minimal Separator Decomposition. 2010.

[GJ 79] Michael R. Garey, David S. Johnson. Computers and Intractability. 1979

[GW 95] Michael X. Geomans and David P. Williamson. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. 1995.

[MS 15] Andrea Montanari and Subhabrata Sen. Semidefinite Programs on Sparse Random Graphs and their Application to Community Detection. 2015.

[O75] Amin Coja-Oghlan. Spectral Techniques, Semidefinite Programs, and Random Graphs. 1975

[FJ94] Alan Frieze and Mark Jerrum. Improved Approximation Algorithms for MAX k-CUT and MAX Bisection. 1994

[L14] Luca Trevisan. Lecture Notes on Expansion, Sparsest Cut, and Spectral Graph Theory. 2014, University Lecture Notes.