E0 249: Approximation Algorithms **Spring 2015**

Lecture 16: Apr 25, 2015

Lecturer: Arnab Bhattacharyya *Scribe: Abhijat Sharma & Suprovat Ghoshal*

16.1 Introduction

Recall the *Label Cover* problem introduced in the previous lecture, as a useful tool in proving hardness of approximation results:

Input: An undirected, unweighted bipartite graph $G = (V_1, V_2, E)$ (where V_1, V_2 denote the two partitions of vertices in G), an alphabet set Σ, and a set of relations $\{\pi_e \subseteq \Sigma \times \Sigma : e \in E\}$.
Output: An assignment (labeling) $L : V_1 \cup V_2 \mapsto \Sigma$ such that maximum edges are satisfied. An edge $e = (u, v) \in E$ ($u \in V_1, v \in V_2$), is said to be satisfied if $(L(u), L(v)) \in \pi_e$.

The *Unique Games* problem is only a special case of the Label Cover, where each relation π_e is a permutation relation, therefore from here on now, we will assume $\pi_e : \Sigma \mapsto \Sigma$ to be a bijection function. Therefore, in a Unique Games instance, an edge $e = (u, v)$ is said to be satisfied by a labeling L if $\pi_e(L(u)) = L(v)$. Let I be any Unique Games instance, we denote by $OPT_{UG} \in [0, 1]$ the maximum fraction of edges that can be satisfied in I by any labeling. We can also define the *Gap Version* of the Unique Games Problem as we have defined for other problems earlier:

Definition 16.1 ([a,b]-GAP-UG problem ($0 \leq a < b \leq 1$)): *Given an instance of the Unique Games Problem, the graph $G = (V_1, V_2, E)$, an alphabet set Σ and the set of relations $\{\pi_e \subseteq \Sigma \times \Sigma : e \in E\}$, with the promise that either $OPT_G < a$, or $OPT_G \geq b$, the goal is to distinguish between these two cases:*

- *YES instance : $OPT_{UG} \geq b$*
- *NO instance : $OPT_{UG} < a$*

Note that from the definition of the Unique Games instance, we can easily observe that $[s, 1]$-GAP-UG problem is easy to solve, for any $s \in [0, 1]$. If we are told that $OPT_{UG} = 1$ for a YES instance, it means there exists some labeling that satisfies every edge, and such a labeling can be easily found in polynomial time. This is because, we can assume that we have the correct label for one vertex $v \in V_1$(by trying out all possible labels to it), and then whenever we know a label for the vertex v_1, it uniquely fixes labels to its neighbors and so on. Thus a labeling can be found for each connected component of the constraint graph.

Hence, from the viewpoint of the *Unique Games Conjecture*, the interesting case is when $OPT_{UG} = 1 - \epsilon$ for some small positive constant $\epsilon > 0$. In this case, the above mentioned algorithm to find a good labeling does not seem to work and one may conjecture that finding even a labeling that satisfies a δ fraction of edges is an NP-hard problem, even when the graph is left-regular.

Conjecture 16.2 (Bipartite Unique Games Conjecture[1]) *For every $\epsilon, \delta > 0$, there exists a set Σ such that $[\delta, 1 - \epsilon]$-GAP-UG is NP-hard, on instances $\mathcal{I} = (G(V_1, V_2, E), \Sigma, \{\pi_e : \Sigma \mapsto \Sigma\}_{e \in E})$, where G is left-regular and π_e are arbitrary bijections.*

16.1.1 Applications of Conjecture 16.2

The unique games conjecture (UGC), often used in its various alternate equivalent formulations, has found many applications and connections between computational complexity, algorithms, analysis, and geometry[2].

- The main motivation for the conjecture is to prove inapproximability results for NP-complete problems that researchers have been unable to prove otherwise. The UGC states that a particular gap version of Unique Games is NP-hard, which, as we have seen earlier implies inapproximability of the Unique games problem. A gap-preserving reduction from the Unique Game problem then implies inapproximability results for other NP-complete problems. This is illustrated in section 16.2.

- It allows us to prove several lower bounds for special classes of constraint satisfaction problems, 2-CSPs (Eg: MaxCut and Max-2-SAT), which we were not able to approach using standard Label Cover reductions. This is because when the constraints involve two variables, it can be more easily related to the bipartite Unique Games problem.

- The inapproximability reductions from the Unique Game problem often use gadgets constructed from a boolean hypercube. The reductions can be alternately viewed as constructions of Probabilistically Checkable Proofs (PCPs) and the gadgets can be viewed as probabilistic checking procedures to check whether a given codeword is a *Long Code*. These type of reductions have been illustrated in section 16.3.

- For many problems, the UGC rules out every polynomial time algorithm to compute a good approximate solution. One might also be interested in exploring the limits of techniques such as linear programming and semi-definite programming (SDP), in approximating a particular problem. In fact, there have been several results which show that reduction from the Unique Game problem to a target problem can in fact be used to construct an (unconditional, explicit) *integrality gap* instance for the target problem.

$$
\begin{array}{lll}
\underset{x}{\text{Maximize}} & \frac{1}{|E|} \sum_{e=(u,v)\in E} \sum_{\sigma\in\Sigma} \langle x_{u,\sigma}, x_{v,\pi_e(\sigma)} \rangle & \\
\text{subject to} & \sum_{\sigma\in\Sigma} \|x_{v,\sigma}\|^2 \; = 1, & \forall v \in V_1 \cup V_2, \sigma \neq \sigma' \in \Sigma \\
. & < x_{v,\sigma}, x_{v,\sigma'} > \; = 0, & \forall v \in V_1 \cup V_2, \sigma \neq \sigma' \in \Sigma. \\
 & < x_{u,\sigma}, x_{v,\sigma'} > \; \geq 0, & \forall u, v \in V_1 \cup V_2, \sigma \neq \sigma' \in \Sigma.
\end{array}
$$

where $x_{u,\sigma}$ are vectors for all $u \in V_1 \cup V_2, \sigma \in \Sigma$, and $\langle a, b \rangle$ denotes the dot-product of vectors a and b. The above SDP can be solved in polynomial time, and no natural distribution of variables is known for which it would be hard to solve the SDP.

Reductions from unique games are broadly of two kinds : *low-tech* reductions which follow naturally from the definitions of the problems and *high-tech* reductions which involve the use of high tech gadgets such as *Dictator Tests*. In the following section, we see a example of a low-tech reduction : the hardness of approximation for Multicut problem.

16.2 Inapproximability of Multicut

For proving the inapproximability of the Multicut problem, we first describe an alternate equivalent formulation of UGC[1], in terms of the *Max-2Lin-K* problem, which is defined as follows:

Input: A system of m linear equations, in n variables $x_1, x_2, ..., x_n$, each equation of the form $x_i - x_j = a_{ij} (\text{mod } k)$.
Output: An assignment of $x_1, x_2, ..., x_n$ to values from $\{0, 1, ..., k-1\}$, that satisfies maximum number of equations.

Conjecture 16.3 (Linear Unique Games Conjecture) *For every $\epsilon, \delta > 0$, there exists an integer k such that $[\delta, 1-\epsilon]$-GAP-Max-2Lin-k is NP-Hard.*

We use this alternate version of UGC to prove an inapproximability result for the Multicut problem. Recall the definition of the Multicut problem:

Input: Given an undirected graph $G = (V, E)$ and a set of pairs of vertices $\{(s_1, t_1), (s_2, t_2), ..., (s_k, t_k)\}$.
Output: A subset of edges $S \subseteq E$ of minimum size such that after removing all the S edges, there is no path from s_i to t_i for all $1 \le i \le k$, in the new graph $(V, E \setminus S)$.

Theorem 16.4 *For any constant $\alpha > 0$, α-approximation of Multicut does not exist, assuming the Unique Games Conjecture.*

Proof: We prove this by describing a reduction from $[\delta, 1-\epsilon]$-GAP-Max-2Lin-k to the $[\epsilon, \frac{1-\delta}{2}]$-GAP-Multicut problem. Given an instance $\mathcal{I}$ of Max-2Lin-k, let $OPT_{Max2LinK}(\mathcal{I})$ denote the maximum fraction of the m equations that can be satisfied in instance $\mathcal{I}$. We would like to have a reduction of instance $\mathcal{I}$ to $\mathcal{I}'$, an instance of the Multicut problem (and $OPT_{Multicut}(\mathcal{I}')$ would denote the fraction of edges in the optimal cut), such that the following holds:

- **Completeness:** If $OPT_{Max2LinK}(I) \ge 1 - \epsilon$ then $OPT_{Multicut}(I') \le \epsilon$.
- **Soundness:** If $OPT_{Max2LinK}(I) < \delta$ then $OPT_{Multicut}(I') > (1-\delta)/2$.

Therefore, we describe the required reduction, and then in-turn prove the above two properties.

Reduction: Given the Max-2Lin-k instance $\mathcal{I}$ having m equations in n variables $x_1, x_2, ..., x_n$, we define the Multicut instance $\mathcal{I}'$, the graph $G = (V, E)$ having $|V| = nk$ vertices and $|E| = mk$ edges. For each variable x_i $(1 \le i \le n)$, we have k corresponding vertices in $|V|$, labelled (x_i, a) where $a \in \{0, 1, ..., k-1\}$. For the edges in $|E|$, consider each equation $x_i - x_j = c_{ij}$, and form an edge between vertices (x_i, a) and (x_j, b) if and only if $a - b = c_{ij}$ (mod k). Finally, after the graph is defined, to complete the Multicut instance I', we define the folowing $s - t$ pairs:

$$\{((x_i, a), (x_i, b)) : a \neq b, 1 \le i \le n\} \tag{16.1}$$

[1] The equivalence of this version to the bipartite unique games conjecture is non-trivial

Completeness: Let $\mathcal{I}$ be a YES-instance of the $[\delta, 1-\epsilon]$-GAP-Max-2Lin-k problem, and therefore there exists an assignment L, assigning values from $\{0, 1, ..., k-1\}$ to the set of variables $X = \{x_1, x_2, ..., x_n\}$ such that at-least $(1-\epsilon) \cdot m$ equations are satisfied. For some $c \in \{0, 1, ..., k-1\}$, we define a set of vertices in the Multicut instance $\mathcal{I}'$,

$$V_c = \{(x_i, a) : x_i \in X, a = L(x_i) + c\}$$

We can observe that each of these vertex-sets (which we will refer to as *clusters*), contain exactly n vertices, and there are k such clusters by definition. Hence, all the edges of the graph can be classifed as *intra-cluster* (having both end-points in the same cluster), and *inter-cluster* (edges going accross different clusters). Also, for every $x_i \in X$, each cluster contains exactly one vertex labelled (x_i, a) where $a \in \{0, 1, ..., k-1\}$ and all other vertices in that particular cluster correspond to other variables. This means that if we select all the inter-cluster edges to form a cut, it will definitely disconnect all the $s-t$ pairs required as in instance $\mathcal{I}'$ (Equation 16.1), and thus the number of such edges form an upper bound on the value $OPT_{Multicut}(\mathcal{I}')$.

Claim 16.5 *If $OPT_{Max2LinK}(\mathcal{I}) \geq (1-\epsilon) \cdot m \cdot k$, then there are at-most $\epsilon \cdot m \cdot k$ inter-cluster edges in the reduced instance $\mathcal{I}'$.*

Proof: Let $(x_i, a) \in V_{c_1}$ and $(x_j, b) \in V_{c_2}$ be the end-points of one such inter-cluster edge accross clusters V_{c_1}, V_{c_2}. So, we know that $a = L(x_i) + c_1$ and $b = L(x_j) + c_2$. Also, because there is an edge between them, $a - b = c_{ij}$ (mod k, where the corresponding equation in instance $\mathcal{I}$ was $x_i - x_j = c_{ij}$. Substituting values of a and b in the above equation gives us:

$$L(x_i) - L(x_j) = c_{ij} - (c_1 - c_2)(\text{mod } k) \tag{16.2}$$

From equation 16.2, $L(x_i) - L(x_j) \neq C_{ij}$ whenever $c_1 \neq c_2$, which corresponds to the unsatisfied equation $x_i - x_j = c_{ij}$ (mod k). And it is easy to observe that each unsatisfied equation will correspond to k equations of the form as Equation 16.2. As we know, the assignment L satisfies at-least $(1-\epsilon)m$ equations, i.e., at-most ϵm equations are not satisfied. which implies there are at-most $\epsilon \cdot m \cdot k$ inter-cluster edges, and hence the claim is proved, which eventually also proves the completeness result. ■

Soundness: Now, for soundness we need to prove that if $OPT_{Max2LinK}(\mathcal{I}) < \delta$ then $OPT_{Multicut}(\mathcal{I}') > (1-\delta)/2$. Instead we prove the counter positive, that is we assume there is a cut of size less than $(1-\delta)|E|/2$ in the reduced instance $\mathcal{I}'$, then we show that there exists an assignment $L : X \mapsto \{0, 1, ..., k-1\}$ to $\mathcal{I}$ that satisfies more than δm equations. Let C be the cut of size less than $(1-\delta)|E|/2$, we remove all the C edges from the input graph, to get the graph $G = (V, E \setminus C)$. It will not be a connected graph, and therefore let $V_1, V_2, .., V_l$ be its connected components (clusters). As each cluster can contain at-most one vertex representing an input variable $x_i \in X$, size of each V_j is at-most n $(1 \leq j \leq l)$, which means $l \geq k$ as there are total nk vertices in the instance $\mathcal{I}'$.

We randomly permute the l connected components, and then for each variable x_i, find the first component V_{C_i} that contains a representative vertex (x_i, a). Then, we assign $L(x_i) = a$. We do this for all variables in the input instance $\mathcal{I}$, $x_1, x_2, ..., x_n$.

Consider an equation in $\mathcal{I}$, $x_i - x_j = c_{ij}$ (mod k). We try to upper-bound the probability that $L(x_i) - L(x_j) \neq c_{ij}$ after the above assignment algorithm. Note that if there is an edge between vertices $(x_i, L(x_i))$ and $(x_j, L(x_j))$, then $L(x_i) - L(x_j) = c_{ij}$. We know that size of the cut $|C| < (1-\delta)/2$, then let us partition the cut edges as follows: Let $(1-\delta_{ij})/2$ be the fraction of edges of the form $(x_i, a), (x_j, b)$ for some $a, b \in \{0, 1, ..., k-1\}$ that go accrosss clusters (and therefore are part of the cut C). Thus, $\sum_{i,j \in [n]} (1-\delta_{ij})/2 = (1-\delta)/2$.

We call a cluster *good* if it contains representatives of two variables x_i, x_j connected by an edge. If an edge $(x_i, a), (x_j, b)$ is not going accross clusters, it will belong to a *good* cluster, thus fraction of *good* clusters, $n_g = 1-(1-\delta_{ij})/2 = (1+\delta_{ij})/2$, and similarly, fraction of *bad* clusters, $n_b \leq (1-\delta_{ij})/2+(1-\delta_{ij})/2 = 1-\delta_{ij}$, with respect to the variables x_i, x_j. To bound the probability that $L(x_i) - L(x_j) \neq c_{ij}$, we first estimate the following:

$$\begin{aligned}
&\mathbb{P}[\text{first cluster containing representatives of } x_i, x_j \text{ is bad}] \\
&= 1 - \frac{n_g}{(n_g + n_b)} \\
&\leq 1 - \frac{(1+\delta_{ij})/2}{(1+\delta_{ij})/2 + (1-\delta_{ij})} \\
&= \frac{2}{3-\delta_{ij}}.(1-\delta_{ij}) \\
&\leq (1-\delta_{ij})
\end{aligned}$$

Thus, $\mathbb{P}[L(x_i) - L(x_j) = c_{ij}] = \mathbb{P}[\text{first cluster containing representatives of } x_i, x_j \text{ is good}] \geq \delta_{ij}$. Summing over all $i, j \in [n]$, we derive that the fraction of equations satisfied by our labelling L, $OPT_{Max2LinK}(\mathcal{I}) \geq \sum_{i,j\in[n]} \delta_{ij} = \delta$, and hence the soundness property holds for this reduction. ■

The second type of reductions are more technically involved : the goal in such reductions is to come up with a PCP verifier that performs special type of tests. In the following sections, we present the hardness of approximation of MAX-CUT as an illustration of the use of such techniques.

16.3 Inapproximability of MAX-CUT

Now we shall look at the hardness of approximation for the MAX-CUT problem. We begin by recalling the problem definition of MAX-CUT :

- *Input* : Undirected Graph $\mathcal{G}(V, E)$
- *Goal* : To compute the set $S \subseteq V$ which maximizes cardinality of the set

$$\{(u, v)|(u, v) \in E, u \in S, v \notin S\}$$

We recall that MAX-CUT has an polynomial time algorithm with an approximation factor of α_{GW} where

$$\alpha_{GW} = \min_{\rho\in[-1,1]} \frac{2 \arccos \rho}{(1-\rho)\pi} \simeq 0.878$$

which was obtained by solving a semidefinite program followed by Goemans-Williamson rounding. In this section we shall prove that assuming the Unique Games Conjecture, it is NP-Hard to solve MAX-CUT with an approximation factor of $\alpha_{GW}+\epsilon$, for any $\epsilon > 0$. We start by introducing the gap version of the MAX-CUT problem $[s, c]$-GAP-MAX-CUT (where $s \leq c$) which is as follows : it outputs *YES* if the optimal solution results in a cut of size $\geq c|E|$ and outputs on *NO* if the optimal (and therefore all) solutions result in cuts of size $< s|E|$. The main theorem stating the hardness of approximation is as follows :

Theorem 16.6 *Assuming the bipartite unique games conjecture, for any $-1 < \rho < 0$ and $\epsilon > 0$, $\left[\frac{\arccos \rho + \epsilon}{\pi}, (1-\epsilon)\frac{1-\rho}{2}\right]$-GAP-MAX-CUT is NP-HARD .*

From the above theorem, the hardness of approximation result follows immediately :

Corollary 16.7 *Assuming the bipartite unique games conjecture, there is no $(\alpha_{GW} + \epsilon)$-approximation algorithm for MAX-CUT for any $\epsilon > 0$ unless* $\mathrm{P} \neq \mathrm{NP}$

This follows directly from theorem 16.6. In the following section, we introduce some concepts and notations from Boolean functional analysis which would be essential for the proof of the hardness result.

16.3.1 Preliminaries

Let $x \in \{\pm 1\}^k$ be a k-bit binary string. Fix a $\rho \in (-1, 0)$. We say that y is ρ correlated to x if y is generated from x as follows : for each $i \in [k]$,

$$y_i = \begin{cases} x_i & \text{w.p. } \frac{1+\rho}{2}, \\ -x_i & \text{w.p. } \frac{1-\rho}{2} \end{cases} \tag{16.3}$$

that is for a fixed $x \in \{\pm 1\}^k$, y is generated by flipping each bit with probability $\frac{1-\rho}{2}$. We denote the distribution of y's generated from x using (16.3) as $y \sim_\rho x$. Given a boolean function $f : \{\pm 1\}^k \longrightarrow \{\pm 1\}$, we define its *Noise Sensitivity* NS_ρ to be as follows :

$$\mathrm{NS}_\rho(f) = \mathbb{P}_{\substack{x \in \{\pm 1\}^k \\ y \sim_\rho x}} \Big(f(x) \neq f(y) \Big)$$

Furthermore, we define the influence of the i^{th} variable on function f as

$$\mathrm{Inf}_i(f) = \mathbb{P}_{x \in \{\pm 1\}^k} \Big(f(x) \neq f(x^i) \Big)$$

where x^i is the same as x except the i^{th} bit which is flipped. Now we state the following result from boolean functional analysis which was conjectured in connection to both hardness of approximation and social choice theory and later proven by E.Mossel et. al. :

Theorem 16.8 Majority Is Stablest *: For any $-1 < \rho < 0$ and $r > 0$, there exists $\beta > 0$ such that if $f : \{\pm 1\}^k \longrightarrow \{\pm 1\}$ has $\mathrm{Inf}_i(f) \leq \beta$ for all $i \in [k]$, then*

$$\mathrm{NS}_\rho(f) \leq \frac{\arccos \rho}{\pi} + r$$

In other words, among all the functions that have low influence on all the bit positions, the majority function has the largest noise sensitivity which is $\frac{\arccos \rho}{\pi} + r$. However, we would be using the contrapositive version of the statement directly: it states that if a function has high noise sensitivity, then it is bound to have atleast one high influential variable (i.e., $\mathrm{Inf}_i(f) > \beta$ for some $i \in [k]$). The phrasing of the above theorem might seem contradictory in the sense that the theorem states that among all functions with low influence, the *majority* function is the most sensitive to noise (and hence, the least stable). However, this happens when $\rho < 0$; for $\rho \geq 0$ the theorem is reversed i.e., the majority function would then be the most stable. For completeness, we prove the theorem for a sub-class of functions (the class of halfspace-cut functions).

Proof Sketch: Let $f(x) = \operatorname{sgn}(a \circ x) = \operatorname{sgn}(\sum_{i\in[k]} a_i x_i)$ where $a_i \in \{\pm 1\} \quad \forall \quad i \in [k]$. Also, without loss of generality, let $\|a\|_2^2 = 1$. For large enough k, the a_i's would be small. Indeed, if $\mathrm{Inf}_i(f) \leq \beta \quad \forall i \in [k]$, then it can be shown that $|a_i| = O(\beta)$ for all $i \in [k]$. Now, we construct b_i's from a_i's as in (16.3) so that $b \sim_\rho a$. Now we observe that

$$\begin{aligned}
\mathrm{NS}_\rho(f) &= \mathbb{P}_{\substack{x\in\{\pm1\}^k \\ y\sim_\rho x}}(f(x) \neq f(y)) \\
&= \mathbb{P}_{\substack{x\in\{\pm1\}^k \\ b\sim_\rho a}}(\operatorname{sgn}(a\circ x) \neq \operatorname{sgn}(b \circ x)) \\
&\overset{(1)}{=} \mathbb{P}_{\substack{g\sim\mathcal{N}(0,1)^n \\ b\sim_\rho a}}(\operatorname{sgn}(a\circ g) \neq \operatorname{sgn}(b\circ g)) \\
&\overset{(2)}{=} \frac{\arccos\rho}{\pi}
\end{aligned}$$

where (1) is due to the fact that a sum of i.i.d. random variables converges in limit to a gaussian distribution (central limit theorem) and (2) follows from the analysis of Goemans-Williamson algorithm (Lecture 11) along with the fact that the a_i s are small in magnitude.

■

Now we are ready to prove Theorem 16.6. The proof is via a reduction from the GAP-UG to MAX-CUT. We construct a 2 bit PCP verifier with $O(\log n)$ randomness for the bipartite unique games. The verifier function would then essentially be a boolean function which takes as input two locations (chosen using the random bits) and accept *iff* the bits at the two locations are unequal. From the instance of the unique games, we construct a GAP-MAX-CUT instance where the vertices are all possible locations of the proof that could be queried using the $O(\log n)$ random bits, and we put a edge between between two vertices whenever they are queried together. The PCP verifier is constructed in a manner such that if the proof is correct, then there are lots of pairs of locations for which the verifier when evaluated at those locations would return true, and hence the GAP-MAX-CUT instance would have a large number of edges between locations equalling $+1$ and the locations equalling -1, which in turn would imply that the GAP-MAX-CUT instance has a large cut.

16.4 Proof of Theorem 16.6

The proof is via the following reduction

$$\Big[\delta, 1-\delta\Big]\text{-GAP-UG} \longrightarrow \Big[\frac{\arccos\rho + \epsilon}{\pi}, (1-\epsilon)\frac{1-\rho}{2}\Big]\text{GAP-MAX-CUT}$$

where $-1 < \rho < 0$. The proof is broken into four parts. First, we look at the construction of the PCP verifier for GAP-UG following which we look at the construction of the GAP-MAX-CUT instance and prove the completeness direction of the reduction. Finally, we give a high level proof for the soundness direction of the reduction.

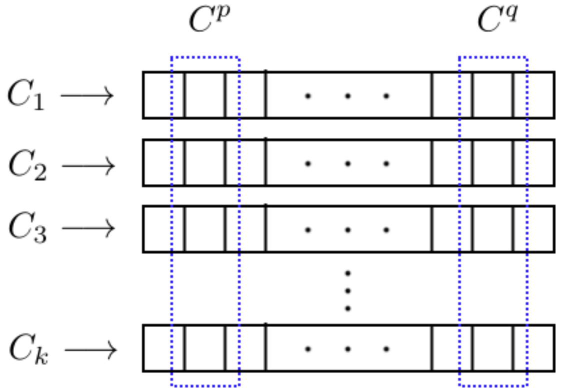

Figure 16.1: Long Code

16.4.1 Long Code Encoding

The satisfying proof consists of encodings of the correct labels of each vertex in the GAP-UG instance into bit strings. For soundness, the PCP verifier needs to check the following aspects :

- *Correctness of Encoding* : The proof consists of valid encodings of the labels.
- *Correctness of Labels* : The labels themselves are such that a large fraction of the constraints in the GAP-UG instance are satisfied

For the first part, we need an encoding scheme such that it is possible to decide (with high probability) whether the codes for the labels are valid codewords by querying only a few (constant number of) locations. In other words, we need the code to be *locally testable* and are highly redundant.

Let $\Sigma = \{1, 2, \ldots k\}$. Suppose we wanted to construct a such a code for Σ such that it is maximally redundant i.e., the length of codewords (say l_0) is as large as possible. Let $C \in \{\pm 1\}^{k \times l_0}$ be the long code for Σ i.e., the i^{th} row C_i is the encoding for label i. How large can l_0 be ? Since the maximum no. of distinct columns of length k is atmost 2^k, it is easy to see that if $l_0 > 2^k$, then by pigeon hole principle, C would have two columns C^p and C^q which are identical to each other (see figure for illustration), and the information from the duplicate column would be useless to the verifier. Therefore, the length of the maximally redundant code can be at most $l_0 = 2^k$. One example of such codes is the *Long Code* which is defined by the set of *Dictator functions* $\{f_i\}_{i \in [k]}$. The i^{th} function $f_i : \{\pm 1\}^k \longrightarrow \{\pm 1\}$ evaluated at x returns the i^{th} bit of x. Formally, the i^{th} dictator function is defined as

$$f_i(x) = x_i$$

where x_i is the i^{th} bit of x. In the following section, we see how the long code is used for the construction of the PCP verifier.

16.4.2 Construction of the PCP verifier

The long code encoding of label $i \in \Sigma$ is given by

$$C_i = (f_i(x), x \in \{\pm 1\}^k)$$

where elements of $\{\pm 1\}^k$ are ordered in some fixed way. For a *YES* instance of $[\delta, 1-\delta]$-GAP-UG, where there is a labelling $L : V_1 \cup V_2 \longrightarrow [k]$, such that atleast $1-\delta$ fraction of the edges are satisfied, the satisfying proof consists of $C_{L(u)} \quad \forall u \in V_1 \cup V_2$. We observe the following properties of the dictator functions which follow from their definitions :

- $\mathcal{P}.1$: $\mathrm{NS}_\rho(f_i) = \frac{1-\rho}{2}$
- $\mathcal{P}.2$: $\mathrm{Inf}_i(f_i) = 1$

Note that $\mathrm{NS}_\rho(f_i)$ is large (since $\rho \in (0, -1)$) and the influence of the i^{th} position on f_i is maximum, which agrees with the contrapositive version of the *MIS* theorem : a function with high noise sensitivity must have atleast one influential variable. Furthermore, let $x \in \{\pm 1\}^k$. Given a permutation $\pi : \{\pm 1\}^k \longrightarrow \{\pm 1\}^k$, we define x_π as

$$x_\pi = (x_{\pi^{-1}(1)}, x_{\pi^{-1}(2)}, \ldots x_{\pi^{-1}(k)})$$

We observe that

$$f_i(x) = x_i = x_{\pi(\pi^{-1}(i))} = f_{\pi(i)}(x_\pi)$$

This, along with the noise sensitivity property $\mathcal{P}.1$ immediately suggests the following test for the verifier :

- Pick $u \in V_1, v \in V_2$ uniformly at random such that $(u, v) \in E$
- Fix an x and draw $y \sim_\rho x$ as in (16.3). If $f_u(x) \neq f_v(y_{\pi_{uv}})$ accept, otherwise reject.

where f_u is the dictator function corresponding to the label $L(u)$. For the completeness criteria, we assume that the proof input to the verifier is correct and the variables are assigned labels such that atleast $1-\delta$ fraction of the edge constraints are satisfied. Then the probability that L satisfies (u, v) is $\geq 1-\delta$ and hence

$$\begin{aligned}\mathbb{P}_{\substack{u,v\\ y \sim_\rho x}}\Big(f_u(x) \neq f_v(y_{\pi_{uv}})\Big) &\geq \mathbb{P}_{\substack{u,v\\ y \sim_\rho x}}\Big(f_u(x) \neq f_v(y_{\pi_{uv}}) | (L(u), L(v)) \in \pi_{uv}\Big)\mathbb{P}_{u,v}\Big((L(u), L(v)) \in \pi_{uv}\Big)\\ &\geq \frac{1-\rho}{2}(1-\delta) \geq \frac{1-\rho}{2}(1-\epsilon)\end{aligned}$$

So the verifier satisfies the completeness condition. However, the test fails miserably for the soundness criteria : take the following instance of a bad proof where u's are labelled as $1, 1, \ldots 1$ (all the positions are 1's) and v's are labelled as $-1, -1, \ldots, -1$ (all the positions are -1's). Clearly these are not long code encodings but

$$\mathbb{P}_{\substack{u,v\\ y \sim_\rho x}}\Big(f_u(x) \neq f_v(y_{\pi_{uv}})\Big) = 1$$

that is , the bad encoding always passes the proposed test. The test fails because it does not really check the validity of the encoding. Hence we use a more subtle approach which is as follows :

- Select $u \in V_1$ and two of its neighbors $v_1, v_2 \in V$ uniformly at random
- Fix an x and sample $y \sim_\rho x$ as in (16.3). If $f_{v_1}(x_{\pi_{uv_1}}) \neq f_{v_2}(y_{\pi_{uv_2}})$ accept, otherwise reject.

In the following sections, we look at the construction of the GAP-MAX-CUT instance and then we analyze the completeness and the soundness properties of the reduction.

16.4.3 Construction of the GAP-MAX-CUT instance

Given a GAP-UG instance $G(V_1, V_2, E), \Sigma, \{\pi_e\}_{e \in E}$, we construct the GAP-MAX-CUT instance $\tilde{G}(\tilde{V}, \tilde{E})$ as follows :

- For each location p_i of the GAP-UG proof that can be queried by the PCP verifier, we include a vertex $\tilde{v}_i \in \tilde{V}$
- For each pair of locations p_i, p_j queried together, we include edge $(\tilde{v}_i, \tilde{v}_j) \in \tilde{E}$

Consider the cut consisting of edges between proof locations equalling $+1$ and the proof locations equalling -1. By construction, it is easy to see that the fraction of edges crossing the cut is the same as the probability that the PCP verifier accepts the proof. In the following sections, we prove the completeness and soundness properties of the reduction.

16.4.4 Completeness

Let the GAP-UG instance have a labelling L such that $\geq 1 - \delta$ fraction of the constraints are satisfied. Then

$$
\begin{aligned}
&\mathbb{P}_{\substack{u,v_1,v_2\\ y\sim_\rho x}}\Big(f_{v_1}(x_{\pi_{uv_1}}) \neq f_{v_2}(y_{\pi_{uv_2}})\Big)\\
&\geq \mathbb{P}_{y\sim_\rho x}\Big(f_{v_1}(x_{\pi_{uv_1}}) \neq f_{v_2}(y_{\pi_{uv_2}}) | (L(u), L(v_1)) \in \pi_{uv_1}, (L(u), L(v_2)) \in \pi_{uv_2}\Big)\\
&\qquad\qquad\qquad\qquad \times \mathbb{P}_{u,v_1,v_2}\Big((L(u), L(v_1)) \in \pi_{uv_1}, (L(u), L(v_2)) \in \pi_{uv_2}\Big)\\
&\overset{(1)}{\geq} \frac{1-\rho}{2}(1-2\delta) \geq \frac{1-\rho}{2}(1-\epsilon)
\end{aligned}
$$

for a small enough δ, where (1) is due to the fact that the GAP-UG instance is left regular, and the neighbors of u are chosen uniformly at random. Hence, in the GAP-MAX-CUT instance the induced graph has atleast $\frac{1-\rho}{2}(1-\epsilon)$ fraction of the edges crossing the cut. This completes the first direction.

16.4.5 Soundness

For the soundness direction, we assume the contra-positive version of the statement i.e., there exists a cut of size atleast $\frac{\arccos \rho}{\pi} + \epsilon$. From this cut, we construct a labelling for the GAP-UG instance such that atleast δ fraction of the edge constraints are satisfied. We observe that

$$\mathbb{P}(\text{ verifier accepts }) = \text{ fraction of edges crossing the cut } \geq \frac{\arccos \rho}{\pi} + \epsilon \tag{16.4}$$

Futhermore, the probability of acceptance can be rewritten as

$$\begin{aligned}
\mathbb{P}(\text{ verifier accepts }) &= \mathbb{E}_{u,v_1,v_2}\mathbb{E}_{y\sim_\rho x}[\frac{1}{2}-\frac{1}{2}f_{\pi_{uv_1}}(x_{\pi_{uv_1}})f_{\pi_{uv_2}}(y_{\pi_{uv_2}})] \\
&= \mathbb{E}_u\mathbb{E}_{y\sim_\rho x}[\frac{1}{2}-\frac{1}{2}\mathbb{E}_{v_1\sim u}[f_{\pi_{uv_1}}(x_{\pi_{uv_1}})]\mathbb{E}_{v_2\sim u}[f_{\pi_{uv_2}}(y_{\pi_{uv_2}})]] \\
&= \mathbb{E}_u\mathbb{E}_{y\sim_\rho x}[\frac{1}{2}-\frac{1}{2}g_u(x)g_u(y)] \qquad \text{where } g_u(x)=\mathbb{E}_{v_1\sim u}[f_{\pi_{uv_1}}(x_{\pi_{uv_1}})] \\
&= \mathbb{E}_u\mathbb{E}_{y\sim_\rho x}[\mathbb{1}(g_u(x)\neq g_u(y))] \\
&= \mathbb{E}_u[\mathbb{P}_{y\sim_\rho x}(g_u(x)\neq g_u(y))] \\
&= \mathbb{E}_u[\mathrm{NS}_\rho(g_u)]
\end{aligned}$$

Now, using a Markov type argument on equation (16.4), we can argue that for atleast $\frac{\epsilon}{2}$ fraction of the u's,

$$\mathrm{NS}_\rho(g_u) \geq \frac{\arccos\rho}{\pi}+\frac{\epsilon}{2}$$

Let $U=\{u_1,u_2,\dots u_l\}$ be the set of such u's. Recall that the contrapositive version of *MIS* states that if the noise sensitivity of a boolean function is high, then it must have atleast one influential variable. Therefore, for each $u\in U$, there exists an influential variable for g_u. We call that influential variable i_u. We label each u as $L(u)=i_u$. Labelling vertices for $v\in V_2$ is more technical, and we present only the high level view of it. It involves making the following observations :

- For a fixed $u\in U$, atleast $\frac{\beta}{2}$ fraction of its neighbors $v\in V_2$ have $\pi_{uv}(i_u)$ as their influential variable in f_v, where β is the parameter from *MIS* theorem.
- Furthermore, for each of the f_v's, there are not too many influential variables.

Hence for a $v\in V_2$, if we make a list of significant variables of f_v and assign one of the variables as the label to v uniformly at random, then there is a significant probability of v being assigned the label $L(v)=\pi_{uv}(i_u)$, which in turn would imply that the atleast δ fraction of the edge constraints in the GAP-UG instance would be satisfied, for a small enough δ. This completes the reduction from $[\delta,1-\delta]$-GAP-UG to $\left[\frac{\arccos\rho}{\pi}+\epsilon,(1-\epsilon)(\frac{1-\rho}{2})\right]$-GAP-MAX-CUT.

16.5 References

[1] S. Khot, On the power of unique 2-prover 1-round games. *In Proc. 34th ACM Symposium on Theory of Computing, 2002.*

[2] S. Khot, On Unique Games Conjecture, *In Proc. 25th IEEE Conference on Computational Complexity, Cambridge, 2010.*