6.893 Approximability of Optimization Problems	October 6, 1999
Lecture 9	
Lecturer: Madhu Sudan	*Scribe: Nati Srebro*

PCP and Hardness of Aproximation

In this lecture, we will see how results about the existence of efficient PCP verifiers can be used to prove hardness of approximation of specific optimization problems. We will do this by reducing any problem having an efficient PCP verifier to a "gapped" version of a hard problem.

We first introduce the PCP results we will use.

1 Overview of PCP results for NP

We introduce the following notation for verifiers: Verifier V on input x and random string R talking to oracle O outputs $V^O(x,R) \in \{\text{accept} = 1, \text{reject} = 0\}$. We say that the verifier is (r,q)-restricted if on input x of length n, the verifier tosses $r(n)$ coins and queries the proof oracle $q(n)$ times.

We can now define the parametric complexity class $\text{PCP}_{c,s}[r(\cdot), q(\cdot)]$, where c, s are **C***ompleteness* and **S***oundness* probabilities, and $r(\cdot), q(\cdot)$ are functions describing the **R**andom bits quota and oracle **Q**ueries quota as a function of the input word length.

$$\text{PCP}_{c,s}[r(\cdot), q(\cdot)] \doteq \left\{ L \subseteq \Sigma^* \;\middle|\; \begin{array}{l} \exists\ (r,q)\text{-restricted verifier } V \\ \forall_n \forall_{x\in\Sigma^n} \;\; x \in L \;\; \Rightarrow \;\; \exists_\pi \mathbf{E}_{R\in\{0,1\}^{r(n)}}\left[V^\pi(x,R)\right] \geq c \\ \qquad\qquad\;\; x \notin L \;\; \Rightarrow \;\; \forall_\pi \mathbf{E}_{R\in\{0,1\}^{r(n)}}\left[V^\pi(x,R)\right] < s \end{array} \right\}$$

Note that each oracle query is independent and sent to the same oracle. The oracle cannot change its response based on previous queries. Another way of thinking of this, instead of oracle access, is receiving a long proof for $x \in L$ written down, but only being allowed to look at q bits of the proof.

The PCP Theorem generally states that every language in NP has a PCP with a logarithmic number of random bits ($r(n) = O(\log n)$) and a constant number of queries ($q = O(1)$). One of the strongest known versions of the PCP Theorem, due to Håstad, makes only three oracle queries / proof lookups and behaves as follows:

Theorem 1 (Håstad) *For every $\epsilon > 0$ and every language $L \in NP$ there is a verifier that uses a logarithmic number $r = O(\log |x|)$ of random bits and a polynomially computable deterministic function*

$$\begin{array}{rrcl} V: & \Sigma^* \times \Sigma^{r(|x|)} & \to & \{0,1\} \times \mathbf{N} \times \mathbf{N} \times \mathbf{N} \\ & (x, R) & \mapsto & (b, i_1, i_2, i_3) \end{array}$$

and decides $x \in L$ iff $\pi_L[x,i_1] \oplus \pi_L[x,i_2] \oplus \pi_L[x,i_3] \stackrel{?}{=} b$ with completness $c = 1-\epsilon$ and soundness $s = \frac{1}{2}+\epsilon$:

$$\begin{array}{llclcl} \mathbf{E}_R(& \pi_L[x,i_1] \oplus \pi_L[x,i_2] \oplus \pi_L[x,i_3] = b & | & x \in L, \;\; V_L(x,R) = (b,i_1,i_2,i_3)) & \geq & c \\ \mathbf{E}_R(& \pi_L[x,i_1] \oplus \pi_L[x,i_2] \oplus \pi_L[x,i_3] = b & | & x \notin L, \;\; V_L(x,R) = (b,i_1,i_2,i_3)) & < & s \end{array}$$

2 Hardness of Gapped Satisfiability

From this statement of the PCP Theorem, we can prove the hardness of approximating GAP-3SAT. First, lets recall the definition of GAP-3SAT. Here and later we use the notation $\#_x\phi$ to denote the number of clauses satisfied by the assignment x in a 3cnf formula ϕ. An instance of GAP-3SAT$_\gamma$ is a pair (ϕ, k) with YES instances satisfying $\exists x$ such that $\#_x\phi \geq k$, while NO instances satisfy $\forall x$, $\#_x\phi < \frac{k}{\gamma}$. If $\gamma = 1$, we get the decision version of the usual MAX 3SAT problem, while for $\gamma > 1$, GAP-3SAT$_\gamma$ is equivalent to γ approximation of MAX 3SAT.

Theorem 2 (Håstad)

$$\forall_{\epsilon>0} \textit{GAP-3SAT}_{\frac{8}{7}-\epsilon} \quad \textit{is NP-hard}$$

Proof For any language $L \in$ NP, we will describe a reduction f from L to GAP-3SAT$_{\frac{8}{7}-\epsilon}$. Let $V : (x, R) \mapsto (b, i_1, i_2, i_3)$ be the function used by Håstad's verifier for the language L. f will map a word x to a collection of $4 \cdot 2^{r(|x|)}$ clauses, four clauses for each possible random string R, over variables $\{y_i\}$, one variable for each bit location in a possible proof π. Since r is logarithmic in $|x|$, the total number of clauses is polynomial in $|x|$, and so is the total number of variables used.

We will use the following quartet of clauses over three variables:

$$Q(y_i, y_j, y_k) = \{(y_i \vee y_j \vee y_k), (y_i \vee \bar{y_j} \vee \bar{y_k}), (\bar{y_i} \vee y_j \vee \bar{y_k}), (\bar{y_i} \vee \bar{y_j} \vee y_k)\}$$

Note that at least three of the clauses are always satisfied, and all four are satisfied iff $y_i \oplus y_j \oplus y_k = 1$. If we have a union of such quartets, the number of satisfied clauses will thus directly correspond to the number of odd parity triplets of variables. In a similar way, $Q(\bar{y_i}, \bar{y_j}, \bar{y_k})$ checks for even parity.

Since parity checking is exactly what the Håstad verifier does, we will map each word x to the following set of clauses:

$$f(x) = \bigcup_{R \in \Sigma^{r(|x|)}} \begin{array}{ll} Q(y_i, y_j, y_k) & \text{if } V(x, R) = (1, i, j, k) \\ Q(\bar{y_i}, \bar{y_j}, \bar{y_k}) & \text{if } V(x, R) = (0, i, j, k) \end{array}$$

Each variable y_i corresponds to the ith bit in a proof. There is thus an implied mapping of a proof π to an assignment y_π, such that $y_i \leftarrow \pi[i]$. For each bit string R and word x, three of the corresponding clauses will always be satisfied, and the fourth will be satisfied iff the verifier would accept x given proof π and random bit string R.

For a word x there are $m \doteq 2^{r(|x|)}$ possible random bit strings, and a total of $4m$ clauses. The number of satisfied clauses $\#_{y_\pi} f(x)$ for the assignment corresponding to the proof π of $x \in L$ is thus exactly $\#_{y_\pi} f(x) = 3m + \mathbf{E}_R[V^\pi(x, R)]m$. (Note that we slightly overload the notation V both for the verifier itself, and for the internal mapping it uses).

We can use the PCP theorem to establish a gap between the maximal number of satisfied clauses for $x \in L$ and for $x \notin L$, thus allowing us to decide $x \in L$ using GAP-3SAT:

$$\begin{array}{rl} x \in L & \\ & \Downarrow \text{PCP theorem with completeness } c = 1 - \epsilon \\ \exists_\pi \mathbf{E}_R[V^\pi(x, R)] \geq 1 - \epsilon & \\ & \Downarrow \text{correspondence of proofs to assignments} \\ \exists_y \#_y f(x) \geq 3m + (1 - \epsilon)m = (4 - \epsilon)m & \end{array}$$

While:

$$\begin{array}{rl} x \notin L & \\ & \Downarrow \text{PCP theorem with soundness } s = \frac{1}{2} + \epsilon \\ \forall_\pi \mathbf{E}_R[V^\pi(x, R)] < \frac{1}{2} + \epsilon & \\ & \Downarrow \text{correspondence of proofs to assignment} \\ \forall_y \#_y f(x) < 3m + (\frac{1}{2} + \epsilon)m = (\frac{7}{2} + \epsilon)m & \end{array}$$

Thus there is a gap of $\frac{4-\epsilon}{\frac{7}{2}+\epsilon} = \frac{8}{7} - \epsilon'$ in the ratio of satisified clauses depending on whether or not $x \in L$. More formally, this allows us to say that the instance $(f(x), (4 - \epsilon)m)$ is a YES instance of GAP-3SAT$_{\frac{8}{7}-\epsilon'}$ if $x \in L$, and a NO instance if $x \notin L$. Thus deciding GAP-3SAT$_{\frac{8}{7}-\epsilon'}$ is NP-hard (if L is NP-hard) and thus approximating MAX-3SAT to within $\frac{8}{7} - \epsilon'$ is NP-hard. ■

2.1 Hardness of GAP-3SAT under a general PCP result

Next we show that even a much weaker PCP result, without any assumptions on the structure of the query and decision rule, is sufficient to prove hardness of GAP-3SAT for some gap:

Theorem 3

$$(\exists\ q \in \mathbf{N},\ c, s \in \mathbf{R},\ c\ >\ s, \quad s.t. \quad NP \subseteq PCP_{c,s}[\log, q]) \quad \Rightarrow \quad \exists g > 1 \mathit{GAP\text{-}3SAT}_g \ \textit{ is NP-hard.}$$

Proof We will assume that the verifier asks non-adaptive queries. That is, that the q queries asked by the verifier are determined by x and R only, and not by answers to previous queries. We will later describe two possible ways in which this assumption can be removed.

For each random string R the Boolean verdict of the verifier is a function of at most q bits from π. The following proposition shows that an arbitrary Boolean function can be converted into a 3-cnf formula. The proof of the proposition is left as an exercise.

Proposition 4 *For every $q \in \mathbf{N}$ there exists $\nu(q), \mu(q) \in N$ such that for every binary function over q variables $h(y_1, y_2, \ldots, y_q)$, there exists a $\mu(q)$-clause 3CNF formula $g(y_1, \ldots, y_q, z_1, \ldots, z_\nu)$ over the variables y, and $\nu(q)$ additional auxiliary variables z, such that*

$$h(y) = 1 \Leftrightarrow \exists_z g(y, z) = \mu$$

and furthermore, at least all but one of the clauses can always be satisfied:

$$\forall_y \exists_z \#_{y,z} g(y, z) \geq \mu - 1$$

We map each word x to a collection $f(x)$ of $\mu(q)2^{r(|x|)} = \mu m$ clauses, over the variables y_i as before, and $\nu(q)2^{r(|x|)}$ additional variables. There are μ clauses for each r-bit string R. Note that each random string R will have ν private auxiliary variables.

This time the mapping between proofs and assignments is not unique, since the auxiliary variables are not directly specified by the proof. But for every proof π of $x \in L$ there exists an assignment y_π, z_π satisfying all μ clauses corresponding to random bit strings on which V accepts, and $\mu - 1$ of the clauses corresponding to random bit strings on which V rejects. Thus we get

$$\forall_\pi \begin{array}{lccl} \exists_z & \#_{y_\pi,z} f(x) & = & m\mathbf{E}_R[V^\pi(x,R)]\mu + m(1 - \mathbf{E}_R[V^\pi(x,R)])(\mu - 1) = m(\mu + E[V^\pi(x,R)] - 1) \\ \forall_z & \#_{y_\pi,z} f(x) & \leq & m\mathbf{E}_R[V^\pi(x,R)]\mu + m(1 - \mathbf{E}_R[V^\pi(x,R)])(\mu - 1) = m(\mu + E[V^\pi(x,R)] - 1). \end{array}$$

Further, note that from any assignment (y, z) to the variables of $f(x)$, we can set up a corresponding proof π (by projecting on to the y part of the assignment). Further for this projected proof π, we find that the setting of z can satisfy at most μ clauses corresponding to random strings that are accepted by the verifier on proof π; while accepting at most $\mu - 1$ clauses corresponding to random strings that are not accepted by the verifier. Thus we find that

$$\begin{array}{lllll} x \in L & \Rightarrow & \exists_\pi E_R[V^\pi(x,R)] \geq c & \Rightarrow & \exists_{y,z} \#_{y,z} f(x) \geq m(\mu + c - 1) \\ x \notin L & \Rightarrow & \forall_\pi E_R[V^\pi(x,R)] < s & \Rightarrow & \forall_{y,z} \#_{y,z} f(x) < m(\mu + s - 1) \end{array}$$

Thus the instance $(f(x), m(\mu + c - 1)$ forms a YES instance of GAP-3SAT$_{\frac{\mu+c-1}{\mu+s-1}}$ if $x \in L$, and a NO instance a gap of $g = \frac{m(\mu+c-1)}{m(\mu+s-1)} = 1 + \frac{c-s}{\mu+s-1} > 1$ (for all $c > s$). ■

2.2 Adaptive queries

If the verifier asks adaptive queries (i.e., if the queried indices may depend on answers to previous queries), we cannot know ahead of time (without knowing π) which q bits will be queried and thus cannot set up the above clauses using Proposition 4.

There are two options here. One, we can enumerate all the possible queries the verifier may make before making any query. This leads to 2^q possible bits queried, which is still a constant if q is a constant. Thus we conclude:

Lemma 5

$$PCP_{c,s}[r, q] \subseteq \textit{non-adaptive-}PCP_{c,s}[r, 2^q - 1]$$

This allows us to use the earlier reduction to the new non-adaptive verifier and derive the same theorem. A more efficient option would be to note that the (adaptive) verifier's computation can still be expressed as a depth q decision tree. And to use this observation, one can prove an alternate version of Proposition 4 for depth q decision trees on n variables and show that the ensuing clauses have $\mu'(q)$ clauses and $\nu'(q)$ auxiliary variables. (In particular, these bounds are independent of n.) Details of this part omitted.

3 Hardness of Gapped Clique/Independent Set

We will now use the PCP Theorem to prove hardness of approximation for Maximum Clique and Maximum Independent Set. Since these two problems are directly reducible one to another, with solutions carrying the same value, and thus the reduction preserving the approximation ratio, they are totally equivalent as far as approximation. We present the hardness results using Maximum Independent Set.

Recall that an instance of GAP-IS$_\gamma$ consists of a pair (G, k) with YES instances consisting of graphs G that have an independent set of size at least k, while NO instances are graphs that have no independent sets of size $\frac{\lceil k}{\gamma\rceil}$.

Theorem 6 (Feige, Goldwasser, Lovasz, Safra, and Szegedy)

$$NP \subseteq PCP_{c,s}[\log, q] \Rightarrow \textit{GAP-IS}_{\frac{c}{s}} \textit{ is NP-hard}$$

Proof We will map the decision formulas of $V(R, x)$ to vertices and edges in a graph (as will be described bellow), mapping every word x into a graph. This mapping will be a reduction from the language verified by V to GAP-IS.

For each x the graph $G(x) = G_{(V,x)}$ will have a vertex for each *accepting* "transcript" $(R, Q) \in \{0,1\}^{r(\|x\|)} \times \{0,1\}^q$ of V's computation. I.e., for each random string R, we enumerate the 2^q possible binary sequences that represent possible sequence of answers to V's oracle queries. For each such sequence Q, we include the pair (R, Q) as a vertex of $G(x)$ if V accepts the sequence Q on random string R.

$G(x)$ will have an edge between (R, Q) and (R', Q') iff they *contradict*. That is, if there is a query $\pi[i]$ that will be asked by V on both (x, R) and (x, R') and it has different responses in Q and in Q'.

Note that a proof π fixes a q-bit answer sequence $Q_{\pi,R}$, for every random string R. Now, for any proof π, let

$$S_\pi = \{(R, Q_{\pi,R}) \,|\, V^\pi(x, R) = 1\}.$$

Note that S_π is an independent set since all (R, Q)'s in S_π agree with π and so do not constradict each other. Thus $\forall_\pi \text{MAXIS}(G(x)) \geq \|S_\pi\| = m\mathbf{E}_R[V^\pi(x, R)]$ (where $m = 2^{r(|x|)}$ as before). Using the completeness of the verifier V we get:

$$x \in L \Rightarrow \exists_\pi \mathbf{E}_R[V^\pi(x, R)] \geq c \Rightarrow \text{MAXIS}(G(x)) \geq mc.$$

For the converse direction, we argue first that for any independent set S of $G(x)$ and for any random string R, S has at most one vertex of the form (R, Q). To see this consider any pair of strings Q and Q' and let i be the first index where they differ. Now since any one query of the verifier is a fixed function of x, R and the history of previous queries and this history is the same after the first $(i-1)$ queries for Q and Q', we note that the i-th query issued by the verifier is the same in the transcripts Q and Q'. But then the answers to the i-th query are different. This implies that Q and Q' contradict each other and thus (R, Q) is adjacent to (R, Q') and thus both can't be in S.

Furthermore, S specifies a partial proof, specifying bits of the proof corresponding to all the oracle queries made by any of the Rs in S. This proof can be extended arbitrarily to a proof π_S such that $\forall_{(R,Q)\in S} V^{\pi_S}(x, R) = 1$. Thus we find

$$\forall_{S \text{ IS of } G(x)} E_R[V^{\pi_S}(x, R)] \geq \frac{\|S\|}{m}$$

$$\Rightarrow \forall_{S \text{ IS of } G(x)} \text{MAXIS}(G(x)) \leq \|S\| \leq m E_R[V^{\pi_S}(x, R)].$$

Using the soundness of V we get:

$$x \notin L \Rightarrow \text{MAXIS}(G(x)) < ms.$$

Thus the instance $(G(x), mc)$ forms a YES instance of $\text{MAXIS}_{c/s}$ if $x \in L$ and a NO instance if $x \notin L$. Notice that the reduction runs in polynomial time as long as $r(\cdot)$ and $q(\cdot)$ are logarithmically growing functions. ■

Note that unlike for GAP-3SAT, the magnitude of q does *not* affect the gap size. As usual, since the hardness of gapped decision implies hardness of approximation, we get:

Corollary 7 *If $NP \subseteq PCP_{c,s}[\log, q]$, then approximating MAXIS to within a ratio of $\frac{c}{s}$ is NP-hard.*

4 Proving hardness for any ratio

Theorem 8 (Johnson) *If there exists a $r < \infty$ IS is r-approximable in polynomial time, then for every $r > 1$ IS is r-approximable in polynomial time.*

To prove the theorem, we will use exponentiation of the graph under the following definition of a product of graphs:

Definition 9 *The* product $G \times H$ *of graphs $G(V, E)$ and $H(W, F)$ has vertex set $V \times W$ and edge set*

$$\begin{array}{rll} E' \doteq & \{\overline{(v, w_1)(v, w_2)} \mid v \in V, w_1, w_2 \in W, & \overline{w_1 w_2} \in F\} \\ \cup & \{\overline{(v_1, w)(v_2, w)} \mid v_1, v_2 \in V, w \in W, & \overline{v_1 v_2} \in E\} \\ \cup & \{\overline{(v_1, w_1)(v_2, w_2)} \mid v_1, v_2 \in V, w_1, w_2 \in W, & \overline{v_1 v_2} \in E \vee \overline{w_1 w_2} \in F\} \end{array}$$

Lemma 10

$$MAXIS(G \times H) = MAXIS(G) \cdot MAXIS(H)$$

Proof The product of every independent set of G and independent set of H is clearly an independent set in $G \times H$. Let K be an independent set in $G \times H$, and let $I \doteq \{v | \exists_w (v, w) \in K\}$, $J \doteq \{w | \exists_v (v, w) \in K\}$ be the projections of K. The projections are independent sets: if some $v_1, v_2 \in I$ are connected in G then the vertices they originated from in K must be connected in $G \times H$, contradicting K's independence. ■

Proof (of Theorem 8) From Lemma 10, we conclude that $\text{MAXIS}(G) = \sqrt[k]{\text{MAXIS}(G^k)}$, and thus an approximation ratio of r on G^k translates to an approximation ratio of $\sqrt[k]{r}$ on G. And so if we can approximate MAXIS to some constant ratio $r < \infty$, we can approximate it to any constant ratio $r' > 1$ by calculating the MAXIS of the $\left\lceil \frac{\log r}{\log r'} \right\rceil$-th power of the graph. For a constant r' the exponent is constant and so the reduction runs in polynomial time. ■

Corollary 11 *Max Independent Set is not approximable to within any constant ratio.*

4.1 Gap expansion using PCP

The same idea can also be carried out in PCP language.

Lemma 12

$$\forall_{k \in N} PCP_{c,s}[r,q] \subseteq PCP_{c^k,s^k}[kr,kq]$$

Proof Given a language $L \in \mathrm{PCP}_{c,s}[r,q]$ with V as the PCP verifier, let $V^{(k)}$ be a new verifier which repeats the actions of the verifier V k times independently, accepting only if all iterations accept. The verifier $V^{(k)}$ tosses k times as many random coins as V does, and makes k times as many (potentially repetitive) queries as V. Since responses to queries are independent of previous queries, the acceptance probability, both for $x \in L$ and for $x \notin L$, will be raised to the kth power. Thus $V^{(k)}$ places L in $\mathrm{PCP}_{c^k,s^k}[kr,kq]$. ■

The lemma above now implies that GAP-IS$_{c^k/s^k}$ is NP-hard for every k and thus (by an appropriate choice of k), GAP-IS$_r$ is NP-hard for every constant r. Notice that since q is a constant, one can afford to use k that is logarithmic in n and still have a reasonable value of the second parameter. However such a choice of k seems to blowup the first parameter to super-logarithmic values. Fortunately, the state-of-the-art results on recycling of randomness, allows us to reduce the randomness to just $r + O(k)$ instead of rk, without losing significantly in the completeness to soundness ratio. For instance, the following theorem (see the paper of Bellare, Goldreich and Sudan for details and history) can be shown along these lines.

Theorem 13

$$\forall_{k \in N} PCP_{c,s}[r,q] \subseteq PCP_{\frac{c^k}{2},2s^k}[r+2k,kq].$$

Combining this theorem with Håstad's verifier that makes 3 queries and has completeness to soundness ratio of $2-\epsilon$, and using $k = \frac{r}{\epsilon}$, one finds that it is hard to approximate the clique size to within $(2-\epsilon)^k$ in a graph with $N = 2^{(5+\epsilon)k}$ is NP-hard. Roughly, this implies that approximating clique to within $N^{\frac{1}{5}}$ in an N-vertex graph is NP-hard. Thus, amplifying in the PCP world seems to be much more beneficial, since we can resort to the high-powered results in recycling of randomness. Subsequent works of Håstad, and Trevisan show how to recylce the query bits as well, thereby improving clique hardness to within $N^{\frac{1}{2}-\epsilon}$ under deterministic reductions and to within $N^{1-\epsilon}$ under randomized reductions.