CS 153 Current topics in theoretical computer science	**Spring 2010**

Solution Set 1

Posted: May 10

1. Let $S_1, S_2, \ldots, S_k$ be the sets selected by the greedy algorithm, and let U be the optimal k-cover. Let T_i be $S_i \setminus (S_1 \cup S_2 \cup \cdots \cup S_{i-1})$; i.e. T_i is the set of new elements covered at step i. Let U_i be the portion of U that remains uncovered by the greedy solution before the i-th step. Since we know that there exists a k-cover of U (and therefore of U_i), we deduce that $|T_i| \geq |U_i|/k$.

   We prove by induction on i that after the i-th greedy choice, the number of elements covered is at least $[1-(1-1/k)^i]|U|$. This is true for $i = 1$ by our above observation that $|T_1| \geq |U_1|/k$.

   Now for the induction step there are two cases: if $|U_i| \geq (1 - 1/k)^{i-1}|U|$, then we have that $|T_i| \geq |U_i|/k \geq (1 - 1/k)^{i-1}/k$ and by induction the first $i - 1$ steps cover at least $[1 - (1 - 1/k)^{i-1}]|U|$ elements. Summing these quantities gives at least $[1 - (1 - 1/k)^i]|U|$ covered elements after the i-th greedy choice, as desired.

   In the other case, we have $|U_i| < (1 - 1/k)^{i-1}|U|$. Before the i-th greedy choice, we have covered at least $|U \setminus U_i|$ elements (just within U). So after the i-th greedy choice we have covered at least

   $$|U| - |U_i| + |U_i|/k = |U| - (1 - 1/k)|U_i| \geq |U| - (1 - 1/k)^i|U| = [1 - (1 - 1/k)^i]|U|$$

   elements, as required.

   Using the fact that $(1 - 1/k)^k \leq 1/e$ for all $k \geq 1$, we find that the greedy algorithm covers at least $(1 - 1/e) \cdot OPT$ elements.

2. Consider a constraint $x_i + x_j + x_k = c$ in the given instance of MAX-3-LIN. If $c = 1$, this is satisfied exactly when 1 or 3 of the variables are 1. Our reduction produces the following majority clauses:

   $$\begin{aligned} &\text{MAJ}(\overline{x_i}, x_j, x_k) \\ &\text{MAJ}(x_i, \overline{x_j}, x_k) \\ &\text{MAJ}(x_i, x_j, \overline{x_k}) \\ &\text{MAJ}(\overline{x_i}, \overline{x_j}, \overline{x_k}) \end{aligned}$$

   Note these are symmetric with respect to permutations of the three variables. This is helpful when determining that exactly 3 clauses are satisfied when 1 or 3 of the variables are 1, and exactly 1 clauses is satisfied when 0 or 2 of the variables are 1.

   If $c = 0$, then our reduction produces the same set of 4 majority clauses, with all their variables negated.

Now, if a $(1-\epsilon)$ fraction of the linear constraints can be simultaneously satisfied, then that assignment satisfies a $(1-\epsilon)(3/4)$ fraction of the majority clauses.

On the other hand, suppose more than a $1/2+\delta/2$ fraction of the majority clauses can be simultaneously satisfied. Let p be the fraction of the groups of 4 majority clauses that have 3 clauses satisfied in this assignment (so a $(1-p)$ fraction have 1 clauses satisfied). Then we have

$$1/2+\delta/2 < \frac{(1-p)+3p}{4} = 1/4 + p/2$$

from which we conclude $p > 1/2+\delta$. But this implies that the assignment satisfies more than $1/2+\delta$ fraction of the linear constraints, violating soundness of MAX-3-LIN.

Thus we have described a reduction with completeness $c = (1-\epsilon)(3/4)$ and soundness $s = 1/2+\delta/2$, which implies that MAX-3-MAJ is **NP**-hard to approximate to better than s/c,which can be made arbitrarily close to $2/3$ by taking δ, ϵ sufficiently small.

3. (a) We produce a graph with 2^r "clouds" of 2^f vertices each. Each cloud is labeled with one of the 2^r possible sequences of coin tosses of the verifier. Each vertex within a cloud is labeled with one of the 2^f distinct answers that would cause the verifier to accept. We have an edge between vertex (α, β) and vertex (α', β') iff $\alpha \neq \alpha'$ and answers β and β' are consistent (i.e. any bits in β and β that correspond to the same proof symbol are equal). Since the PCP system has completeness 1, we know that in the positive case, there is a clique of size 2^r, consisting of the single vertex in each cloud that is consistent with the proof (because every answer derived from this proof is consistent with every other one). For the soundness direction, we first note that any clique can contain at most one vertex from each cloud. Thus if there is a clique of size greater than $s2^r$, then there is a proof that causes the verifier to accept with probability greater than s (any proof consistent with the answers forming the clique suffices). Thus in the negative case there is no clique larger than $s2^r$.

   (b) We describe the new verifier. It uses R bits of randomness to select $x \in [2^R]$. It then invokes the original verifier with randomness $E(x, y)$ for each $y \in [D]$, and accepts if every one of these invocations would accept. The new verifier uses the required randomness R, and has free-bit complexity at most Df since it invokes the old verifier D times. Any proof causing the original verifier to accept with probability 1 will also cause the new verifier to accept with probability 1, so the completeness of the new verifier is 1 as required. For soundness, we fix a proof π and consider the set $B \subseteq [2^R]$ of "bad" random strings; i.e. those for which the new verifier accepts. Since we are in the soundness case, we know that at most $s2^r$ of the random strings for the original verifier cause it to accept. Since every $x \in B$ must have $E(x, y)$ among these strings for all $y \in [D]$, it must be that $|B| \le K$, as otherwise B is a set that violates the definition of a (K, s)-disperser. So the soundness of the new verifier is at most $K/2^R$ since there are most K bad strings from among 2^R total.

   (c) We start with the **FPCP** system of Theorem 1.1, with randomness r, and apply the previous part, using a $(2^{R\epsilon}, 2^{-\ell})$ disperser $E : [2^R] \times [D] \to [2^r]$ with $D \le O(R/\ell)$ and $M = 2^r$ (we need to take $R = cr$ for some constant c for Theorem 1.2 to apply). Applying the first part, we obtain a graph of size $2^{R+D\overline{f}\ell}$, for which it is **NP**-hard to distinguish between a clique of size at least 2^R and a clique of size at most $2^{R\epsilon}$. The ratio is $2^{R(1-\epsilon)}$ for arbitrarily small ϵ, so we only need to ensure that the size of the graph is at most $2^{R(1-\delta)}$ for arbitrarily small δ, and we are done (then the ratio is $N^{1-\epsilon'}$ for arbitrarily small ϵ', where N is the size of the graph). This means we need $D\overline{f}\ell < \delta R$. But notice that $D\overline{f}\ell = O(R\overline{f})$. So by taking $\overline{f}$ sufficiently small (as allowed by Theorem 1.1), we are done.

4. (a) Since $A' \cup B'$ constitutes a cover, by the defining property of (m, ℓ) set systems, we have that the union must contain C_j and $\overline{C_j}$ for some j. Since B consists only of uncomplemented sets and A consists only of complemented sets, we must have $\overline{C_{\pi(i)=j}} \in A'$ and $C_j \in B'$. So, our randomized procedure is as follows: pick i to be a random i from among the $\overline{C_{\pi(i)}}$ in A', pick j to be a random j from among the C_j in B'. We have just argued that exists a good pair from among A' and B', so the probability we hit it is at least $(1/|A'|) * (1/|B'|) \geq 4/\ell^2$.

   (b) We first note that the starting instance of label cover can be assumed to be balanced (i.e., $|V_1| = |V_2|$) by first repeating the left node set $|V_2|$ times (with each copy inheriting the same edges and constraints), and then repeating the left node set $|V_1|$ times (with each copy inheriting the same edges and constraints). It is easy to see that both transformations preserve perfect completeness, and that after each transformation, any assignment to the new graph satisfying s fraction of the edges must satisfy an s fraction of some copy of the original graph, so soundness is exactly preserved.

   As suggested, we produce the instance with universe $E \times U$ and sets

$$S_{u,i} = \bigcup_{v:(u,v)\in E} \{(u,v)\} \times \overline{C_{\pi_{(u,v)}(i)}}$$

   for each $u \in V_1$ and $i \in [m]$, and

$$S_{v,i} = \bigcup_{u:(u,v)\in E} \{(u,v)\} \times C_i$$

   for each $v \in V_2$ and $i \in [m]$.

   In the completeness case, if $A : (V_1 \cup V_2) \to [m]$ is a labeling satisfying all edges, then we claim that $S_{u,A(u)}$ for $u \in V_1$ together with $S_{v,A(v)}$ for $v \in V_2$ consistute a cover. Indeed, for each edge (u, v) with associated constraint $\pi = \pi_{u,v}$, $S_{u,A(u)}$ contains the set $\{(u, v)\} \times \overline{C_{\pi(A(u))}}$ and $S_{v,A(v)}$ contains the set $\{(u, v)\} \times C_{A(v)}$. Since this edge is satisfied by assignment A, we know that $\pi(A(u)) = A(v)$ and hence the entire subset $\{(u, v)\} \times U$ is covered. Since this holds for all edges, we have a cover of size $|V_1 \cup V_2|$.

   For soundness, define a set of candidate labels for each vertex as follows: for $u \in V_1$, let $D(u)$ be those i for which $S_{u,i}$ is in the cover, and for $v \in V_2$, let $D(v)$ be those j for which $S_{v,j}$ is in the cover. A natural decoding procedure is to pick a label for each vertex randomly from the associated set of candidates.

   Consider the set of vertices for which there are fewer than $\ell/2$ candidates. Call such a vertex "good". For an edge (u, v) for which both u and v are good, the analysis reduces to the situation described in the first part: namely, the subset $\{(u, v)\} \times U$ must be covered by the $S_{u,i}$ and $S_{v,j}$ sets in the cover, because no other sets include elements with (u, v) as their first component. Thus the sets $\overline{C_{\pi_{(u,v)}(i)}}$ for $i \in D(u)$ and C_j for $j \in D(v)$ must cover U, but there are at most ℓ of them. So, by the first part, the randomized decoding procedure satisfies this edge with probability at least $4/\ell^2$.

By an averaging argument, at most $1/4$ of the graph nodes can have candidate sets larger than $\ell/2$ (since the overall number of sets is at most $\ell/8$ times the number of vertices). Since the graph was assumed to be balanced, at most p_1 fraction of left-nodes are not good, and at most p_2 fraction of right-nodes are not good with $p_1 + p_2 \leq 1/2$.

By regularity, picking a random edge is the same as picking a left-node and a random neighbor, which is the same as picking a random right-node and a random neighbor. Thus when selecting a random edge the probability of picking the left endpoint in the non-good set is at most p_1 and (separately) the probability of picking a right endpoint in the non-good set is at most p_2, and we just argued that $p_1 + p_2 \leq 1/2$. Altogether the probability we pick an edge with both endpoints "good" vertices is at least $1/2$. Whenever this happens, our randomized decoding procedure succeeds with probability at least $4/\ell^2$. The total fraction of edges satisfied (in expectation) is thus $2/\ell^2$.

(c) Let n be the size of the instance of 3-SAT to which we will apply Theorem 1.3. Set $\epsilon = \Theta(1/(\log^3 n))$. We obtain an instance of LABEL COVER with size $N = n^{O(\log\log n)}$ and alphabet size at most $m = (\log n)^{O(1)}$.

Set $\ell = \gamma \log n \log\log n$, where γ is a small enough constant to ensure that $\epsilon < 2/\ell^2$ (the inequality holds asymptotically, so such a constant γ exists). This ensures soundness of the set cover instance via the previous part. The required (m, ℓ) set system can be constructed in $n^{O(\log\log n)}$ time, by Theorem 1.4. By the previous part, we have a soundness/completeness ratio of $\Omega(\ell)$. Since $\log N = O(\log n \log\log n)$, we have achieved a hardness ratio of $\Omega(\log N)$ as required. Any approximation algorithm achieving better than this ratio can be used to solve 3-SAT (and hence all of **NP**) in time $n^{O(\log\log n)}$ by performing this reduction (which runs in time $n^{O(\log\log n)}$) and then running the approximation algorithm in the resulting SET COVER instance.