Problem Set 1

- Due Date: **15 Mar (Thurs), 2010**
- It is recommended that you try to solve all the exercises and problems, but you need to submit the writeup for only the problems.
- Collaboration is encouraged, but all writeups must be done individually.
- Indicate names of all collaborators.
- Refering sources other than the lecture notes is discouraged, since for some of the problems a Google search will reveal the solution. But if you do use an outside source (text books, lecture notes, any material available online), do mention the same in your writeup.

Exercises

1. **[Maximum Acyclic Subgraph (MAS)]**

   Give a 2-approximation for the following problem

   Input: directed graph $G = (V, E)$
   Output: A subset E' of edges $E' \subseteq E$ such that $G' = (V, E')$ is acyclic.
   Objective: Maximize $|E'|$.

2. **[2-approximation for Vertex Cover using DFS]**

   Consider the following approximation algorithm for finding the minimum vertex cover. Find a depth first search tree of the given graph and output the set S of all nonleaf vertices. Show that S is indeed a vertex cover for G and $|S| \leq 2 \cdot OPT$.

3. **[Tight example for 3/2-approximation for metric TSP]**

   In lecture, we discussed a 3/2-approximation for metric TSP due to Christofides. Convince yourself that there is a tight example for the analysis of this algorithm. More precisely, construct a graph (and an associated minimum spanning tree and a perfect matching on the degree one vertices of the MST), such that the approximation algorithm when run on this MST and matching gives a TSP that is almost 3/2 times the optimal TSP.

4. **[FPTAS and strongly NP-hard]**

   An optimization problem $\Pi = (\mathcal{X} \times \mathcal{Y}, \text{feas} : \mathcal{X} \times \mathcal{Y} \to \{0,1\}, \text{objective} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R})$ is said to be strongly NP-hard, if it is NP-hard and furthermore, the optimal value can take values only within a polynomial sized set. More precisely, Π is strongly NP-hard if Π is NP-hard and there exists a polynomial $p(\cdot)$ such that for every input instance $x \in \mathcal{X}$, $OPT(x) \in \{1, 2, \ldots, p(|x|)\}$.

   Show that if an optimization problem Π has an FPTAS, then it cannot be strongly NP-hard unless NP=P.

Problems

1. **[MAXFLOW = MINCUT]**

   In this problem, we will prove the MAXFLOW = MINCUT theorem using LP duality. Recall the definition of MAXFLOW and MINCUT from lecture.

   Let $G = (V, E)$ be a directed graph with s and t being the source and target vertices. The capacity of an edge is a mapping $c : E \to \mathbb{Z}^{\geq 0}$, denoted by $c(u, v)$. It represents the maximum amount of flow that can pass through an edge. A flow is a mapping $f : E \to \mathbb{R}^+$, denoted by $f(u, v)$, subject to the following two constraints:

   - $f(u, v) \leq c(u, v)$ for each $(u, v) \in E$ (capacity constraint)
   - $\sum_{v:\ (u,v)\in E} f(u, v) = \sum_{v:\ (v,u)\in E} f(v, u)$ for each $u \in V \setminus \{s, t\}$ (conservation of flows)

   The value of flow is defined by

   $$|f| = \sum_{v\in V} f(s, v) - \sum_{v\in V} f(v, s),$$

   where s is the source. It represents the amount of flow passing from the source to the target.

   The maximum flow problem is to maximize $|f|$ among all possible flows f, that is, to route as much flow as possible from s to t. The value of this maximum is called MAXFLOW.

   An $s - t$ cut $(S, \bar{S})$ is a partition of V such that $s \in S$ and $t \in \bar{S}$. The set of edges $E(S, \bar{S}) = \{(u, v) \in E | u \in S, v \in \bar{S}\}$ is the cut-set of the cut $(S, \bar{S})$. Note that if the edges in the cut-set of C are removed, $|f| = 0$. The capacity of an $s - t$ cut is defined by

   $$c(S, \bar{S}) = \sum_{(u,v)\in E(S,\bar{S})} c(u, v).$$

   The minimum cut problem is to minimize $c(S, \bar{S})$ among all possible cuts $(S, \bar{S})$, that is, to minimize the amount of capacity of an $s - t$ cut. The value of this minimum is called MINCUT.

   Prove that MAXFLOW=MINCUT.

   [Hint: You may use strong LP duality to prove the above theorem. That is, frame the MAXFLOW and MINCUT problems as linear programs and then use duality.
   LP Duality: Given a linear program (also called the primal program), "Max $c^T x$ subject to $Ax \leq b, x \geq 0$", the dual of this program is defined as "Min $b^T y$ subject to $A^T y \geq c, y \geq 0$". Weak duality states that any feasible primal solution is always less than a feasible dual solution. Strong LP duality states that if the primal has an optimal solution, then the dual also has an optimal solution and furthermore the objective value attained by both the primal and dual optimals are the same.]

2. **[TSP with weights 1 and 2]**

   Let G be an undirected complete graph in which all the edge weights are either 1 or 2 (clearly, G satisfies the triangle inequality). Give a 4/3-approximation for TSP in this special class of graphs.

   [Hint: Start by finding a minimum 2-matching in G. A 2-matching is a subset S of edges so that every vertex has exactly 2 edges of S incident at it.]

3. **[asymmetric TSP]**

   Give an $O(\log n)$-approximation for the following problem.

   Input: Directed complete graph $G = (V, E)$ with a non-negative edge cost $c(u \to v)$ specified for every pair of vertices $(u, v) \in V \times V$ The edge costs satisfy the *directed triangle inequality*, i.e., for any three vertices u, v, w, $c(u \to v) \leq c(u \to w) + c(w \to v)$.
   Output: A cycle $\mathcal{C} = v_0, v_1, \ldots, v_n = v_0$ of the vertices such that each vertex appears exactly once along the cycle.
   Objective: Minimize $c(\mathcal{C}) = \sum_{i=1}^{n} c(v_{i-1} \to v_i)$.

   [Hint: You may use the fact that a minimum cost cycle cover (i.e., disjoint cycles covering all the vertices) can be found in polynomial time. Shrink the vertices and recurse.]

4. **[Primal-dual analysis for SETCOVER]**

   The SETCOVER problem is defined as follows.

   Input: A collection of n sets $S_1, \ldots, S_n$ such that $\bigcup_{i=1}^{n} S_i = U = \{1, 2, \ldots, m\}$.
   Output: A subcollection $I \subseteq [n]$ of sets such that $\bigcup_{i \in I} S_i = U$. Such an I is called a cover of U.
   Objective: Minimize the size of the subcollection $|I|$.

   There is a natural greedy algorithm for this problem along the following lines.

   (a) Add the largest set S_i to the cover.

   (b) Delete S_i from the collection of sets and all elements of S_i from the universe U to obtain a smaller sized SETCOVER instance.

   (c) Repeat the above procedure as long as the universe U is non-empty.

   In this problem, we will give a primal-dual analysis to demonstrate that this greedy algorithm gives a $\ln(\max_i |S_i|)$-approximation.

   Consider the following LP-relaxation of the SETCOVER problem. Here, we associate each set S_i with a indicator variable x_i which is 1 iff S_i is in the cover.

$$\begin{array}{rcll} \text{Minimize} & & \displaystyle\sum_{i=1}^{n} x_i & \\ \text{subject to} & & & \\ \displaystyle\sum_{i: j \in S_i} x_i & \geq & 1, & \forall j \in \{1, \ldots, m\} \\ 0 \leq & x_i & \leq 1, & \forall i \in \{1, \ldots, n\} \end{array}$$

The dual of this LP-relaxation is as follows:

$$\begin{array}{rrcll} \text{Maximize} & \displaystyle\sum_{j=1}^{m} y_j & & & \\ \text{subject to} & & & & \\ & \displaystyle\sum_{j \in S_i} y_j & \leq & 1, & \forall i \in \{1, \ldots, n\} \\ & y_j & \geq & 0, & \forall j \in \{1, \ldots, m\} \end{array}$$

This dual can be interpreted as the following packing problem: we wish to assign a non-negative weight y_j to each of the elements in the universe so that the sum of weights is as large as possible, with the constraint that the sum of weights of any set is at most 1.

We will now view each step of the greedy algorithm as updating a pair of solutions, one for the primal LP and one for the dual LP. All the dual solutions will be dual feasible solutions while all but the final primal solution will be primal infeasible solutions. Initially all the variables x_i and y_j are set to zero. Each time the greedy algorithm adds a set S_i to the cover, the variables are updates as follows:

- Primal: Set $x_i \leftarrow 1$.
- Dual: For all elements $j \in S_i$, set $y_j \leftarrow 1/(|S_i| \cdot B)$.

where B is a mysterious constant whose value will be set later. B will be chosen large enough that the dual constraints will always be satisfied.

(a) Show that each run of the greedy algorithm increments the primal solution by 1 and the dual solution by $1/B$. Hence, show using weak duality that the greedy algorithm is a B-approximation. (Recall that at termination both the primal and dual solutions are feasible.)

(b) Show that at the end of the greedy algorithm, for every set S_i

$$\sum_{j \in S_i} y_j \leq \frac{1}{B} + \frac{1}{2B} + \frac{1}{3B} + \cdots + \frac{1}{|S_i| \cdot B} = \frac{H_{|S_i|}}{B},$$

where H_k denotes the k-th harmonic number, $1 + 1/2 + 1/3 + \cdots + 1/k \sim \ln k$.

(c) Argue using parts 4a and 4b, that we can set $B = \ln(\max_i |S_i|)$ and thus, greedy gives a $\ln(\max_i |S_i|)$-approximation algorithm.

5. **[Greedy 2-approximation for KNAPSACK]**

Recall the KNAPSACK problem

Input: n items with sizes $s_1, \ldots, s_n$ and values $v_1, \ldots, v_n$ and a knapsack of total size B. Furthermore, assume $s_i \leq B$ for all i.
Output: A subset of items $I \subset [n]$ such that $\sum_{i \in I} s_i \leq B$.
Objective: Maximize $\sum_{i \in I} v_i$.

Consider the following modification to the greedy algorithm for KNAPSACK discussed in lecture. Sort the items by decreasing ratio of value to size. Let the sorted order of objects be $a_1, \ldots, a_n$. Find the lowest k such that the total size of the first $(k+1)$ elements exceeds B. Now, pick the more valuable of the sets $\{a_1, \ldots, a_k\}$ and $\{a_{k+1}\}$. Show that this algorithm achieves a 2-approximation.

6. **[weighted version of vertex cover]**

   Consider the following weighted version of Vertex Cover (W-VC).

   Input: Undirected graph $G = (V, E)$ with weights $w : V \to \mathbb{Z}^{\geq 0}$ on the vertices.
   Output: A cover $C \subseteq V$ of the vertices such that for every edge $(u, v) \in E$ either $u \in C$ or $v \in C$.
   Objective: Minimize the weight of the cover (i.e., $\sum_{v \in C} w(v)$).

   Observe that the 2-approximation algorithm for the unweighted version discussed in lecture does not extend to this weighted version. Design an alternate deterministic 2-approximation algorithm for W-VC.

   [Hint: First design a LP relaxation of the problem with variables for each vertex in the graph and then deterministically round the LP to obtain a 2-approximate solution.]

7. **[gap preserving reductions]**

   A reduction from one gap problem gap-A$_\alpha$ to gap-B$_\beta$ (for some $0 < \alpha, \beta < 1$) is said to be a gap preserving reduction if it maps YES instances of gap-A$_\alpha$ to YES instances of gap-B$_\beta$ and NO instances of gap-A$_\alpha$ to NO instances of gap-B$_\beta$. The existence of a gap preserving reduction from gap-A$_\alpha$ to gap-B$_\beta$ implies that if it is NP-hard to approximate problem A to within α, then it is NP-hard to approximate problem B to within β.

   For every $\alpha > 0$, show that there exists a and ε, β and a gap preserving reduction from gap-3SAT$_\alpha$ to gap-2SAT$_{1-\varepsilon,\beta}$. Hence, conclude that there exists a $\beta \in (0, 1)$ such that approximating MAX2SAT to within β is NP-hard.

   Note:The gap problems gap-3SAT$_\alpha$ and gap-2SAT$_{1-\varepsilon,\beta}$ are defined as follows.
   gap-3SAT$_\alpha$:

$$\begin{aligned} \mathsf{YES} &= \{\varphi | \varphi \text{ is a satisfiable 3CNF formula}\} \\ \mathsf{NO} &= \{\varphi | \varphi \text{ is a 3CNF formula such that no assignment satisfies more} \\ &\quad\ \text{than } \alpha \text{ fraction of the clauses}\} \end{aligned}$$

   gap-2SAT$_{1-\varepsilon,\beta}$:

$$\begin{aligned} \mathsf{YES} &= \{\varphi | \varphi \text{ is a 2CNF formula with an assignment that satsifies} \\ &\quad\ \text{at least } (1-\varepsilon)\text{-fraction of the clauses}\} \\ \mathsf{NO} &= \{\varphi | \varphi \text{ is a 2CNF formula such that no assignment satisfies more} \\ &\quad\ \text{than } \beta \text{ fraction of the clauses}\} \end{aligned}$$

8. **[three vs. two queries]**

   In class, we stated that Håstad proved the following strengthening of the PCP Theorem which shows that every language in NP has a PCP with 3 queries and soundness error almost 1/2.

   $$[\text{Håstad}]\ \forall \varepsilon > 0, \textsf{Circuit-SAT} \in PCP_{1-\varepsilon,1/2+\varepsilon}[O(\log n), 3].$$

   Suppose we were able to further strengthen the above result to prove that Circuit-SAT has a 2 query PCP (i.e., Circuit-SAT $\in PCP_{1,s}[O(\log n), 2]$ for some $0 < s < 1$), then show that then $NP = P$!

   Thus, Håstad's PCP is optimal with respect to the number of queries till the status of the P vs. NP question is resolved.

9. **[inapproximability of clique via graph products]**

   In class, we proved the following theorem showing the inapproximability of clique. 3-COLOR $\in PCP_{c,s}[r, q]$ implies it is NP-hard to approximate MAXCLIQUE to within a factor s/c as long as $2^{r+q} = \text{poly}(\cdot)$. This resulted in the following inapproximability result for MAXCLIQUE assuming the PCP Theorem (3-COLOR $\in PCP_{1,1/2}[O(\log n), O(1)]$).

   $$\exists \alpha \in (0, 1),\ \text{it is NP-hard to approximate CLIQUE to within } \alpha \tag{1}$$

   We then applied sequential repetition on the PCP (i.e., $PCP_{c,s}[r, q] \subseteq PCP_{c^k,s^k}[kr, kq]$ for all $k \in \mathbb{Z}^{\geq 0}$) to obtain the following strengthening of the above result.

   $$\forall \alpha \in (0, 1),\ \text{it is NP-hard to approximate CLIQUE to within } \alpha \tag{2}$$

   In this problem, we will discuss an alternative approach to prove this result using graph products. For a graph $G = (V, E)$ we define the square of G, $G^2 = (V', E')$, as follows: The vertex set V' equals V^2, the set of pairs of vertices of G. Two distinct vertices (u_1, u_2) and (v_1, v_2) are adjacent in E' if and only if $(u_1, v_1) \in E$ and $(u_2, v_2) \in E$.

   (a) Prove that the squaring operation defined above satisfies $\omega(G^2) = (\omega(G))^2$ where $\omega(G)$ denotes the size of the largest clique in G.

   (b) Use (a) to given an alternate proof of (2) from (1).

10. **[recycling randomness via random walks on an expander]**

    In lecture, we showed that by sequential repetition of PCPs (i.e., $PCP_{c,s}[r, q] \subseteq PCP_{c^k,s^k}[kr, kq]$ for all $k \in \mathbb{Z}^{\geq 0}$) can be used to improve the hardness factor of approximating clique (also see earlier problem). In this problem, we will discuss a more efficient way to perform repetition by recycling randomness using expander walks.

    Let $G = (V, E)$ be an (n, d, λ)-expander with $\lambda < d$ i.e, G is a d-regular graph on n vertices such that the second largest eigenvalue (in absolute value) has absolute value at most λ.. Let $B \subseteq V$ be a set of vertices with $|B| = \mu n$, where $0 < \mu < 1$. Suppose we pick a uniformly random vertex in G and then perform a t-step random walk in G starting from this vertex. We wish to upper-bound the probability p that all vertices encountered along this random walk are in the set B.

(a) Let A denote the normalized adjacency matrix of G, and let P denote the matrix corresponding to *projection* onto B; in other words, P is the $n \times n$ diagonal matrix with 1's in the positions corresponding to B. Show that $p = \|P(AP)^t \pi\|_1$, where π is the vector $(1/n, \ldots |V| \text{times} \ldots, 1/n)$ (ie., the probability distribution of a random vertex in V), and $\|z\|_1$ denotes the l_1-norm of z (i.e., $\|z\|_1 = \sum_{i=1}^{n} |z_i|$).

(b) The matrix 2-norm of a matrix C is defined to be $\|C\|_2 = \max_{y \neq 0} \|Cy\|_2 / \|y\|_2$. Show that $p \leq \mu \|PAPAP \ldots AP\|_2 \leq \|AP\|_2^t$.

(c) Show that $\|AP\|_2 \leq \sqrt{\mu + (\lambda/d)^2}$, and conclude $p \leq (\mu + (\lambda/d)^2)^{t/2}$.

[Hint: given arbitrary $y \neq 0$, write $z = Py$ and decompose $z = z_{\|} + z_{\perp}$ where $z_{\|}$ is the component of z along the all one's vector and $z_{\perp}$ the orthogonal component.]

Extra Credit: show that in fact $\|PAP\|_2 \leq (\lambda/d) + \mu(1 - \lambda/d)$ and show how this can be used to conclude the sharper upper bound $p \leq \mu(\lambda/d) + \mu(1 - \lambda/d)^t$.

(d) Use the earlier part (c) to conclude that $\mathrm{PCP}_{1,1-s}[r, q] \subseteq \mathrm{PCP}_{1,2^{-k}}[r + O(k), O(kq)]$ for all $k \in \mathbb{Z}^{\geq 0}$.

(e) Conclude from (d) (setting $k = \log n$) that it is NP-hard to approximate to within $n^{-\delta}$ for some $\delta \in (0, 1)$.