Assignment sheet 4

Due date: January 20, 2021

---

1. Show that the dual of the dual of a linear program is the original program itself.

2. Show that any language in the class NP can be decided by an algorithm with running time $2^{O(n^k)}$ for some constant k.

3. The complexity class co-NP is the set of those languages L such that $\overline{L} = \Sigma^* - L$ is in NP. Show that if $NP \neq$ co-NP, then $P \neq NP$.

4. Given a 3CNF formula ϕ, Max-SAT is the problem of counting the maximum number of clauses that can be satisfied by any true/false assignment to the variables in ϕ. Show that the following is a factor 1/2 approximation algorithm for Max-SAT.

   Let τ be an arbitrary truth assignment and let τ' be its complement, i.e., a variable is *true* in τ if and only if it is *false* in τ'.

   - Compute the weight of the clauses satisfied by τ and τ', then output the better assignment.

5. The TSP problem is the following: given a complete undirected graph $G = (V, E)$ on n vertices with non-negative edge costs, find a minimum cost cycle visiting every vertex exactly once. Show that for any polynomial time computable function $\alpha(n)$, TSP cannot be approximated within a factor of $\alpha(n)$ in polynomial time, unless P = NP.

   (You can assume that the NP-hardness of the *Hamiltonian cycle* problem, which asks for a cycle that visits every vertex exactly once. Show that an $\alpha(n)$-approximation algorithm for TSP can be used to solve the Hamiltonian cycle problem in polynomial time.)

6. Consider the *metric-TSP* problem, i.e., the input is as in the above problem and edge costs satisfy triangle inequality. In other words, for any 3 vertices x, y, z we have: $c(x, y) \leq c(x, z) + c(y, z)$. Consider the following algorithm:

   (a) Find an MST T of G.

   (b) Double every edge of T to obtain an *Eulerian graph*.
   [This is a graph where every vertex has even degree.]

   (c) Find an Eulerian tour τ of this graph.
   [This is a cycle (need not be a simple cycle, i.e., vertices may be revisited) that traverses every edge in the graph exactly once.]

   (d) Output the tour C that visits vertices of G in the order of their first appearance in τ.

   Show that C is a 2-approximation of the optimal TSP cycle, i.e., a min-cost cycle of length n in G.

7. Show that the following problem cannot be approximated within any polynomial time computable factor α, unless $P = NP$.

   Given a graph $G = (V, E)$ with positive weights on its edges, and a positive integer k, find a subset S of vertices of cardinality k such that the total weight of edges in the subgraph induced by S is minimized.

8. Consider the LP-rounding algorithm discussed for 2-approximation algorithm for the weighted vertex cover problem. Recall that for any i: if $x_i^* \geq 1/2$, then we included vertex i in our set C and we showed that C is indeed a vertex cover and its weight is at most $2 \cdot \mathsf{OPT}$.

   Consider the following rounding rule: if $x_i^* > 0$, then include vertex i in C. It is easy to see that such a set C is indeed a vertex cover. Show that the weight of C is also within $2 \cdot \mathsf{OPT}$.

9. The following problem is called the *multiway cut* problem and it is NP-hard for $k \geq 3$: Given a set of terminals $S = \{s_1, \ldots, s_k\} \subseteq V$, a multiway cut is a set of edges whose removal disconnects the terminals from each other. The multiway cut problem asks for the minimum weight such set of edges. (Note that the case $k = 2$ is the minimum s-t cut problem.)

   Design a simple $2 - 2/k$ factor approximation algorithm for this problem.

   (Hint: First compute a minimum weight cut for each s_i that disconnects s_i from the rest of the terminals.)

10. We wish to solve the minimum vertex cover problem on input instances that consist of a graph G, together with a valid vertex colouring of G with 3 colours. Give a 4/3-approximation algorithm for the minimum cardinality vertex cover problem on such instances.

    (*Hint:* You can use the fact that we can compute a minimum cardinality vertex cover in a bipartite graph in polynomial time.)