

Department of Computer Science and Information Systems
CSIS 6924 (Algorithms),
CSIS 0324 (Topics in Theoretical Computer Science)
Assignment 2

Deadline: November 25, 1:45pm.

We accept both printed and written answers.

Remember to put your name and university number in the answer script.

Submit it to assignment box A6 (cargo lift lobby of 3/F, Chow Yei Ching building).

Answer all questions. They carry the same marks.

For CSIS 0324 students, please note the additional questions overleaf.

1. [p961, Ex 36.5-5] HAMILTONIAN-PATH is defined as follows: given an undirected graph $G = (V, E)$, check whether G contains a path in G containing all vertices. That is, it has the same requirement as HAMILTONIAN-CYCLE, except that the first and the last vertices need not be adjacent. Prove that the problem is *NP*-complete. (Hint: reduce HAMILTONIAN-CYCLE to it.)
2. [p974, Ex 37.2-3] In this problem, we approximate the bottleneck traveling-salesman problem, defined as follows. Given a complete undirected graph $G = (V, E)$ and a non-negative integer cost $c(e)$ for each edge $e \in E$, we want to find a tour $H(G)$ of G that minimize the cost C of the most costly edge used. Again, we assume the triangle inequality holds for the costs. We denote the cost of the optimal solution as C^* .
 - (a) Given an arbitrary tree T , show how to construct a permutation $P_2(T)$ for all vertices in T , such that each vertex is included exactly once, all consecutive vertices in $P_2(T)$ are of distance at most 3 in T , and the first vertex in $P_2(T)$ is of distance at most 3 from the last vertex of $P_2(T)$ in T . (Hint: use recursion.)
 - (b) Given a graph G . Prove that the minimum spanning tree $M(G)$ of G contains only edges with costs no more than C^* .
 - (c) Show that the following algorithm achieves a ratio-bound of 3 on approximating the bottleneck TSP: given G , return $P_2(M(G))$.
3. [p984, Prob 37-2, modified] Let $G = (V, E)$ be an undirected graph. For any $k \geq 1$, define $G^{(k)}$ to be the undirected graph $(V^{(k)}, E^{(k)})$, where $V^{(k)}$ is the set of all ordered k -tuples of vertices from V and $E^{(k)}$ is defined so that (v_1, v_2, \dots, v_k) is adjacent to (w_1, w_2, \dots, w_k) if and only if for all i , either vertex v_i is adjacent to w_i in G , or $v_i = w_i$.
 - (a) Prove that the size of the maximum clique in $G^{(k)}$ is equal to the k -th power of the size of the maximum clique in G .
 - (b) Argue that if there is an approximation algorithm that has a constant ratio bound for finding a maximum-size clique, then there is a polynomial-time approximation scheme for the problem.

(N.B.: we can continue and shows that clique cannot be approximated with constant ratio bound, unless an NP -hard problem has sub-exponential running time. The result has been improved to show that there exists $\epsilon > 0$ so that clique cannot be approximated with any ratio bound less than n^ϵ , unless $P = NP$ [S. Arora, C. Lund, R. Motwani, M. Sudan and Szegedy, Proof verification and intractability of approximation problems. In the Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science, pages 13–22, 1992].)

Additional questions for CSIS 0324 students

1. [p928, Ex 36.2-4] Consider two languages L_1 and L_2 in NP .
 - (a) Prove that the language $L_1 \cup L_2$, $L_1 \cap L_2$ and $L = \{uw : u \in L_1, w \in L_2\}$ are all in NP . (Note: uw denotes the concatenation of the strings u and w .)
 - (b) Consider the following “proof” that $\overline{L_1}$ (the set of all strings not in L_1) is in NP : if there is a polynomial time algorithm that decides $\overline{L_1}$ in polynomial time using non-determinism, then we can invoke the algorithm, negate the result (either accept or reject), and thus decide L_1 . Thus $\overline{L_1}$ is in NP .

What is the problem of the above proof?

2. [Parts of p983, Prob 37-1] Suppose that we are given a set of n objects, where the size s_i of the i -th object satisfies $0 < s_i < 1$. We wish to pack all the objects into the minimum number of unit-size bins. Each bin can hold any subset of the objects whose total size does not exceed 1.

The first-fit heuristic takes each object in turn and places it into the first bin that can accommodate it. Let $S = \sum_{i=1}^n s_i$.

- (a) Argue that the optimal number of bins required is at least $\lceil S \rceil$.
- (b) Argue that the first-fit heuristic leaves at most one bin less than half-full.
- (c) Prove that the number of bins used by the first-fit heuristic is never more than $\lceil 2S \rceil$.
- (d) Prove a ratio bound of 2 for the first-fit heuristic.