CECS 528 MW, LO Assessment 3, Fall 2026, Dr. Ebert

IMPORTANT: For each problem (LO5,LO6,GK,Advanced, All LO Makeups) write your solution using a **SINGLE** and **SEPARATE SHEET OF PAPER (FRONT AND BACK IF NEEDED)**. Write your **FULL NAME** and **PROBLEM CODE** on each sheet. Questions? Need scratch paper? Raise hand.

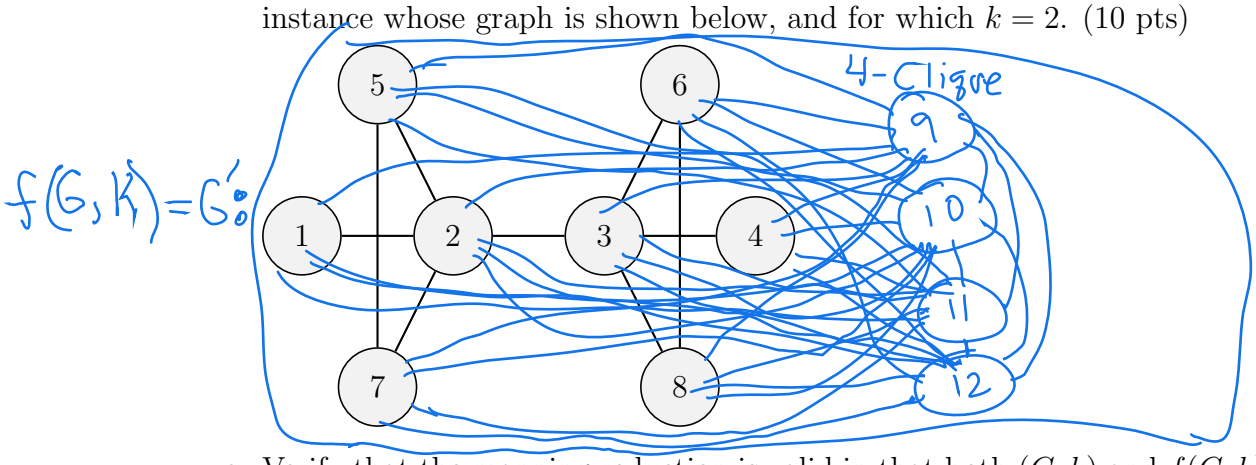
25 Points Each

LO5 Do the following.

a. Provide the definition of what it means to be a mapping reduction from computational problem A to computational problem B. (5 pts)

Solution. See definition from lecture notes.

b. For the mapping reduction f : `Clique` $\rightarrow$ `Half Clique`, draw $f(G,k)$ for the `Clique` instance whose graph is shown below, and for which $k = 2$. (10 pts)



c. Verify that the mapping reduction is valid in that both (G,k) and $f(G,k)$ are either both positive instances, or are both negative ones. Explain and show work. (10 pts)

Solution. $(G, k = 2)$ is a positive instance since, e.g., $\{2,5\}$ is a 2-clique for G. Also, $G' = f(G,k)$ is a positive instance of `HC` since $\{2,5,9,10,11,12\}$ is a half-clique for G'.

LO6 Do the following.

a. An instance of the `Graph Isomorphism` decision problem consists of two simple graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, where $V = \{1, \ldots, n\}$ is the vertex set common to both graphs, and $|E_1| = |E_2|$, meaning that both graphs have the same number of edges. Moreover, G_1 and G_2 are said to be **isomorphic**, denoted $G_1 \cong G_2$, iff there is some bijective function $f : V \rightarrow V$ which has the property that, for every pair of distinct integers $i, j \in V$, $(i,j) \in E_1$ iff $(f(i), f(j)) \in E_2$. Thus, a natural certificate for an instance of `Graph Isomorphism` is a bijective function $f : V \rightarrow V$. To show that `Graph Isomorphism` is an `NP` problem, do the following.

i. Provide the pseudocode for a verifier program for `Graph Isomorphism` that takes as inputs G_1, G_2, and bijection $f : V \to V$, and returns 1 iff, for every pair $i, j \in V$, $(i, j) \in E_1$ iff $(f(i), f(j)) \in E_2$.
Solution.

For each $i, j \in V$ with $i \neq j$,

If $(((i, j) \in E_1) \oplus ((f(i), f(j)) \in E_2))$, then return 0.

Return 1.

(8 pts)

ii. Given that the size parameters for `Graph Isomorphism` are $m = |E_1| = |E_2|$ and n, determine the big-O number of steps required by your verifier. Defend your answer. Make sure to include any operations that may be hidden/implied by the pseudocode and describe the number of steps that each requires. (5 pts)
Solution. The `for`-loop requires $O(n^2)$ iterations, and each iteration requires $O(1)$ steps assuming that i) f is implemented as an array, and ii) edge lookup can be done using lookup tables that require $O(m)$ steps to construct. Therefore, the number of steps equals $O(n^2 + m) = O(n^2)$.

b. Classify each of the following problems as being in `P`, `NP`, or co-`NP`. Note: at least three correct answers is necessary for passing this part of LO6. (3 points each).

i. An instance of `Integer Quadratic Equation` is a triple of integers (a, b, c) and the problem is to decide if the equation $ax^2 + bx + c = 0$ has an integer solution.

ii. An instance of `Steiner Tree Problem` is an undirected weighted graph $G = (V, E, w)$, a subset $U \subset V$ of vertices, and a nonnegative integer k. The problem is to decide if there is a tree subgraph of G that includes U (and possibly more vertices from $V - U$) and whose total weight does not exceed k. Note: this problem is *much harder* to solve than the one that is solved by Prim and Kruskal, because the vertices of the tree are note entirely fixed.

iii. An instance of the `Diophantine Quadratic Inequality` problem consists of three positive integers a, b, and c, the problem is to decide if, for every pair of positive integers (x, y), it is always true that $ax^2 + bx \neq c$.

iv. An instance of the `Graph Coloring` problem is a simple graph $G = (V, E)$ and a positive integer k. The problem is to decide if there is some way to color the vertices of G using k colors for which no two adjacent vertices have the same color. Note: if someone allowed you to color the vertices using a large box of crayola crayons, then $k = 64$ in that case. Just make sure to stay within the lines. ha! ha!.
Solution. P, NP, co-NP, NP

GK Do the following.

a. Provide formal definitions for `P`, `NP`, `co-NP`. (15 pts)
Solution. See lecture notes.

b. Explain why, for any decision problem $L \in$ `P`, it must also be true that $\overline{L} \in$ `P`. (10 pts)
Solution. Any algorithm that decides L can be altered so that each return statement returns the negation of what is returned when deciding L.

ADV Do the following.

a. Provide a mapping reduction f : `VC` $\rightarrow$ `IS` from the `Vertex Cover` to `Independent Set` decision problems. For full credit, function f must be rigorously defined and you must prove that the reduction is valid for any arbitrary instance of `VC`. (15 pts)

b. Prove that `P` $\subseteq$ `NP` $\cap$ co-`NP`. (10 pts)

LO MakeUps: Pass/No Pass

LO1 a. Use the Master Theorem to determine the growth of $T(n)$ if it satisfies the recurrence $T(n) = 4T(n/2) + n^2$. Defend your answer.

b. Use the Substitution method to establish that, if $T(n) = 4T(n/2) + n^2 \log n$, then $T(n) = \mathrm{O}(n^2 \log n)$.

LO2 a. Consider Karatsuba's algorithm for multiplying two n-bit binary numbers x and y, where we assume that n is even. Let x_L and x_R be the leftmost $n/2$ and rightmost $n/2$ bits of x respectively. Define y_L and y_R similarly. Let P_1 be the result of calling `multiply` on inputs x_L and y_L, P_2 be the result of calling `multiply` on inputs x_R and y_R, and P_3 the result of calling `multiply` on inputs $x_L + x_R$ and $y_L + y_R$. Then return the value $P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2$. Demonstrate Karatsuba's algorithm on $x = 35$ and $y = 42$, where both number are assumed to require six bits. Verify through calculation that the algorithm does in fact compute xy. Limit your demonstration to the root level of recursion. In other words, do not go deeper than level 0.

b. What is the divide-and-conquer recurrence for the divide-and-conquer algorithm that solves the `Minimum Positive Subsequence Sum` problem and what is the resulting running time? Defend your answer by desribing the algorithm using sufficient detail. The description should address each part of the recurrence.

LO3 a. The following two questions pertain to Kruskal's algorithm.

i. State the greedy choice that is being made in each round of the algorithm.

ii. Apply the algorithm to the following set of edges, where each triple in set E represents the two vertices that are adjacent, followed by the edge weight.

$$E = \{(1,2,130), (1,3,170), (1,9,120), (2,4,140), (2,6,70), (2,10,90), (3,6,190), (3,10,150)$$

$$(4,5,160), (4,6,180), (5,7,200), (5,10,50), (7,8,20), (7,9,30), (7,10,80), (8,9,110)\}.$$

b. For the edges provided in part a, suppose Prim's algorithm is used instead. Draw a plausible binary heap after Round 5 and assuming vertex 1 was the first vertex added to the tree (i.e. after five vertices have been added to the Prim tree and the first one added being vertex 1). Then redraw the heap after the root vertex is popped, list any priority increases that result from the pop and show a final heap that results after all priority increases have been executed in the order that they are listed.

LO4

Solve the following problems.

a. The dynamic-programming algorithm that solves the `Floyd Warshall` optimization problem defines a recurrence using the matrix entry d_{ij}^k. In words, what does d_{ij}^k equal? Hint: do *not* write the recurrence (see Part b).

b. Provide the dynamic-programming recurrence for d_{ij}^k.

c. Given the directed graph whose edge-cost matrix is

$$d^0 = \begin{pmatrix} 0 & 2 & 6 & 4 \\ 1 & 0 & 5 & 9 \\ 6 & 1 & 0 & 4 \\ 9 & 2 & 6 & 0 \end{pmatrix},$$

use the recurrence to compute d^1, d^2, d^3, and d^4.