CECS 528 TuTh, 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

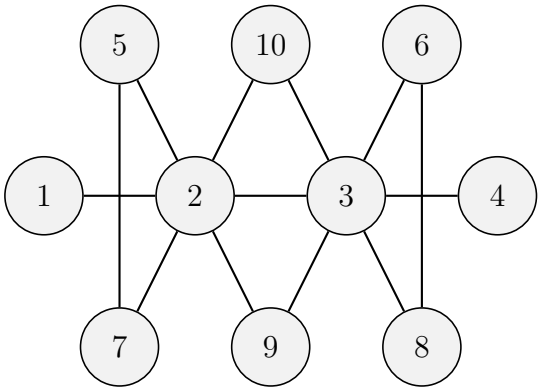
LO5 Do the following.

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 : `VC` $\rightarrow$ `HVC`, draw $f(G, k)$ for the `Vertex Cover` instance whose graph is shown below, and for which $k = 4$. (10 pts)

Solution. $f(G, k)$ is the same graph as G but with $n - 2k = 2$ isolated triangles added with the first having vertices 11,12, and 13, and the second having vertices 14, 15, and 16.



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. $(G, k = 4)$ is a positive instance of `VC` since, e.g., vertex set $\{2, 3, 5, 6\}$ covers all edges of G. Also, $G' = f(G, k)$ is a positive instance of `HVC` since $\{2, 3, 5, 6, 11, 12, 14, 15\}$ is a half vertex cover for G' since it has 8 vertices and $G' = f(G, k)$ has 16 vertices.

LO6 Do the following.

a. An instance of the `Set Cover` decision problem consists of a triple $(\mathcal{S}, m, k)$, where $\mathcal{S} = \{S_1, \ldots, S_n\}$ is a collection of n subsets, where $S_i \subseteq \{1, \ldots, m\}$, for each $i = 1, \ldots, n$, and a nonnegative integer k. The problem is to decide if there are k subsets $S_{i_1}, \ldots, S_{i_k}$ for which

$$S_{i_1} \cup \cdots \cup S_{i_k} = \{1, \ldots, m\}.$$

Thus, a natural certificate for an instance of `Set Cover` is a subcollection

$$\{S_{i_1}, \ldots, S_{i_k}\} \subseteq \mathcal{S}.$$

To show that `Set Cover` is an `NP` problem, do the following.

i. Provide the pseudocode for a verifier program for `Set Cover` that takes as inputs $(\mathcal{S}, m, k)$ and certificate $\{S_{i_1}, \ldots, S_{i_k}\}$, and returns 1 iff

$$S_{i_1} \cup \cdots \cup S_{i_k} = \{1, \ldots, m\}.$$

Solution.

Initialize array `covered` so that `covered`$[r] = 0$ for each $r \in \{1, \ldots, m\}$.
For each $j \in \{1, \ldots, k\}$,
  For each $l \in \{1, \ldots, |S_{i_j}|\}$,
    `covered`$[S_{i_j}[l]] = 1$. $//S_{i_j}$ is represented as an array.
For each $r \in \{1, \ldots, m\}$,
  If `covered`$[r] == 0$, then return 0.
Return 1.

(8 pts)

ii. Given that the size parameters for `Set Cover` are m and $n = |\mathcal{S}|$, 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. Initialize `cover`: $O(m)$. The outer loop requires $O(n)$ iterations, the inner loop requires $O(m)$ steps, and updating the `cover` array requires $O(1)$ steps. Thus, a total of $O(mn)$ steps is required to execute the nested loops. Finally, checking if `covered`$[r] == 0$ for some $r \in \{1, \ldots, m\}$ requires $O(m)$ steps. Therefore, total number of steps equals $O(m + mn) = O(mn)$.

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 decision problem `Zero Matrix Power (ZMP)` consists of an $n \times n$ matrix A having integer entries and the problem is to decide if $A^n = 0$.

ii. An instance of `Cubic Diophantine` is a triple (a, b, c, d) of natural numbers and the problem is to decide if there no triple of positive integers $x, y, z > 0$ for which

$$ax^3 + bx^2 + cx = d.$$

Hint: this problem is not in P.

iii. An instance of the `Fallible` decision problem is a Boolean formula F and the problem is to decide if there is an assignment α to the variables of F for which $F(\alpha) = 0$.

iv. An instance of the `Substring` decision problem is a pair of binary strings (s_1, s_2), and the problem is to decide if s_1 is a substring of s_2. For example, abba is a substring of bbabbaa because the latter has a abba within it.
Solution. P, co-NP, NP, P

GK Do the following.

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

Solution. See lecture notes.

b. Explain why every `P` problem is also an `NP` problem, as well as also a co-`NP` problem. (10 pts)

Solution. Every problem L in `P` has a program that decides it in a polynomial number of steps. This program serves as an NP-verifier for L since, on input x, if it returns 1, then it proves that x is a positive instance of L. Similarly, it also serves as a co-NP-verifier since, if the program returns false, then proves that x is a negative instance of L.

ADV Do the following.

a. An instance of `Set Tri-Partition (STP)` is a set S of nonnegative integers and the problem is to decide if there are subsets A, B, C of S for which

$$A \cup B \cup C = S$$

and

$$A \cap B = A \cap C = B \cap C = \emptyset.$$

Provide a mapping reduction $f : \texttt{SP} \rightarrow \texttt{STP}$ from the `Set Partition` to the `Set Tri-Partition` decision problem. For full credit, function f must be rigorously defined and you must prove that the reduction is valid for any arbitrary instance of `SP`. (15 pts)

Solution. Let M denote the sum of all members of S. If M is odd, then S is a negative instance of `SP`, Thus, assigning $f(S) = S' = S \cup \{M\}$ maps a negative instance to a negative instance of `STP`, since M itself accounts for more than $\frac{1}{2} > \frac{1}{3}$ of the total sum of S'. On the other hand, if M is even, then $f(S) = S' = S \cup \{M/2\}$ is a valid reduction, since S is a positive instance iff $f(S, t)$ is a positive instance, since the positivity of both depend on whether or not S can be partitioned into two sets A and B for which both sets sum to $M/2$ which is the sum of subset $C = \{M/2\} \subseteq S'$.

b. Suppose you have an efficient algorithm $\mathcal{A}$ that will take as input a simple graph $G = (V, E)$ and a nonnegative integer $k \geq 0$ and will output 0 or 1 depending on whether or not G has a vertex cover of size k. In the case that G does have a VC of size k, explain how $\mathcal{A}$ could be used to compute the actual cover. Assume that $\mathcal{A}$ is like a black box and that you have no way of examining or learning anything about how $\mathcal{A}$ works. All you can do is use $\mathcal{A}$. (10 pts)

Solution. For each vertex $v \in V$ add a new vertex u to G and connect it to v. Call the new graph G'. If $\mathcal{A}(G', k) == 0$, it means that v is not in the k cover since, otherwise, (u, v) would have been covered by the existing k cover. Since, v is not in the cover. Repeating this for all vertices of G will determine which vertices are in the cover.

LO MakeUps: Pass/No Pass

LO1 a. Use the Master Theorem to determine the growth of $T(n)$ if it satisfies the recurrence $T(n) = 8T(n/2) + n^3 \log^2 n$. 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. Demonstrate the `Median-of-Five Find Statistic Algorithm` when applied to the array

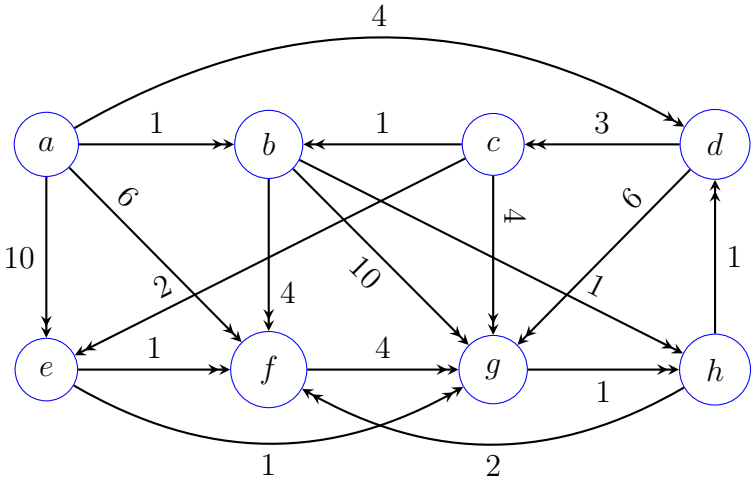
$$a = 86, 96, 35, 16, 36, 94, 55, 50, 77, 8, 44, 15, 19, 32, 55, 98, 68, 78, 70, 7, 3, 20, 87$$

with $k = 7$, lower $= 0$, and upper $= 22$. Compute the pivot M, the arrays a_{left} and a_{right}, and the updated lower and upper values. Hint: although you do not need to show the step by step details, nevertheless you should be applying the partitioning algorithm from Hoare's `Quicksort`.

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 Do the following.

a. Consider the following instance of the `Single-Source Distances` problem where vertex a is the source vertex.



Make a table whose rows are the vertices and whose eight columns shows the neighboring distance of each vertex from the source, and the parent of the vertex along the path that creates the neighboring distance. For each of the eight rounds, circle the neighboring distance that wins the round and use the parent information to add the winning vertex to the Dijkstra Distance Tree. "Dash out" the remaining table cells for the winning vertex.

b. For the weighted graph with edges

$$(a, f, 6), (b, d, 3), (c, f, 2), (c, e, 5), (d, e, 4), (e, f, 1),$$

Show how the forest of disjoint-set data structure trees changes when processing each edge in Kruskal's sorted list of edges. When merging two trees, use the convention that the root of the merged tree should be the one having *lower* alphabetical order. For example, if two trees, one with root a, the other with root b, are to be merged, then the merged tree should have root a. Redraw the forest each time an edge is processed.

LO4

Solve the following problems.

a. The dynamic-programming algorithm that solves the `Edit Distance` optimization problem defines a recurrence for the function $\mathrm{d}(i, j)$. In words, what does $\mathrm{d}(i, j)$ equal? Hint: do *not* write the recurrence (see Part b).

b. Provide the dynamic-programming recurrence for $\mathrm{d}(i, j)$.

c. Apply the recurrence from Part b to the words u = baccbb and v = abbbcc. Show the matrix of subproblem solutions and use it to provide an optimal solution.