ECE GY 9343 Final Exam (2022 Spring)
Version X

Name: **NetID:**

Session(Circle one): Session A (Prof. Yong Liu) Session B (Prof. Pei Liu) Online

---

Answer ALL questions. Exam is closed book. No electronic aids. However, you are permitted two cheat sheets, two sides each sheet. Any content on the cheat sheet is permitted.

Multiple choice questions may have **multiple** correct answers. You will get **partial credits** if you only select a subset of correct answers, and will get zero point if you select one or more wrong answers.

Requirements for in-person exam

Please answer all questions on the question book. You should have enough space. Since we scan all the submissions in a batch, DON'T WRITE ANSWERS ON THE BACK of any page.

Make sure you write your NetID on the bottom of each page (abc1234). Don't tear off any page. If you need extra scrape papers, we can give to you. However, it will not be graded.

Requirements for remote exam

You should only have two electronic devices on your desktop. 1) A computer where you read the exam paper from; 2) A smartphone or a tablet with a camera that you use to join the Zoom Session. Make sure both devices are plugged into a power source all the time.

Your cheat sheets must be on paper, not on your computer. You must write your solutions on paper, not on a tablet/iPad. If you use a pencil to answer, please use a pencil of 2B or darker.

Close all other windows/tabs on your computer. You should only use it to read the exam questions. Avoid typing on the computer once you downloaded and saved the questions.

Put your Zoom device on your side. Turn on your camera and set it up so that I have a good view of your work area, your face, your hands and computer screen and keyboard. Make sure your device is charged. Change your Zoom name to your Net ID, then your full name. Mute your microphone but leave the **speaker on** so you can hear announcements. If the proctor identifies something unusual, we will announce it on the speaker for three times. If you fail to respond, your grade will be zero for this exam.

Always keep your video on. **Don't move your camera** unless the device fells. You might want to use a device holder, or at least use a pile of books to hold the device. If you have questions, please unmute yourself and ask the question by voice. If you need to use the restroom, send us a chat message that you need to use restroom and then you can leave. Send us another message when you come back from the restroom.

Submission **deadline is 15 minutes** once exam ends. You can use Apple Notes App or Adobe Scan to get a single PDF document. You can temporarily turn off the camera at the end of the exam to scan the exam papers. If you miss the deadline, we CANNOT take your exam papers unless you email us right away. The timestamp for email submission should no later than 15 minutes. We can take JPEG/PDF submissions via email. DON'T WAIT TILL the last minute and tell us you don't have enough time to scan!

1. **(20 points) True or False**

   (a) **[✓ T] or F:** Worst case tree height for AVL tree with n nodes is $\theta(\log(n))$.

   (b) **[✓ T] or F:** To save time on determining whether an edge (u, v) exists in a graph, one should use adjacency matrix, instead of adjacency list, to represent the graph.

   (c) **[✓ T] or F:** For an undirected graph with n vertices to be acyclic, the maximum number of edges is $n - 1$.

   (d) **T or [✓ F]:** There is only one valid topological ordering of vertices in a directed acyclic graph.

   (e) **T or [✓ F]:** A dynamic programming algorithm has to find all optimal solutions of the problem to be solved.

   (f) **[✓ T] or F:** 0-1 knapsack problem is NP-Complete, but can be solved by dynamic programming;

   (g) **T or [✓ F]:** Kruskal's algorithm for minimum spanning tree does not work on a graph with negative weight edges.

   (h) **[✓ T] or F:** For any Huffman code, if we exchange the codewords for the two least frequent symbols, the new code is still optimal.

   (i) **[✓ T] or F:** The least weighted edge must be in the minimum spanning tree of a connected graph.

   (j) **T or [✓ F]:** If all the edge weights are distinct, there can be only one shortest path between any pair of nodes;

2. (**4 points**) Which of the following statements about depth-first search (DFS) are true?

   (a) [✓ Complexity of DFS on a dense graph is $\theta(E)$;]

   (b) DFS can generate shortest path in a graph;

   (c) [✓ In a DAG, if there is an edge (u, v), then $f(u) > f(v)$;]

   (d) [✓ For an undirected graph, DFS tree has no cross edges;]

   (e) None of the above.

3. (**4 points**) Which of the following graph problems CANNOT be solved in linear time $(O(|V| + |E|))$?

   (a) Determining a topological order of the vertices for a directed acyclic graph;

   (b) Determining if an undirected graph is connected;

   (c) Determining if an undirected graph has at least one loop;

   (d) [✓ Finding the minimum spanning tree of a connected undirected graph;]

   (e) None of the above.

4. (**4 points**) Which of the following algorithms does NOT require a heap for its efficient implementation?

   (a) Huffman coding algorithm;

   (b) Dijkstra's algorithm;

   (c) Prim's algorithm;

   (d) [✓ Floyd–Warshall algorithm;]

   (e) None of the above.

5. (**4 points**) In the activity selection problem, if a new activity is added to the input activity set, which has the earliest finishing time among all activities. Compare with the solution of the original problem, which of the following are true?

   (a) [✓ The maximum number of activities can be scheduled together will not reduce;]

   (b) [✓ This new activity is in at least one of the optimal solutions to the new problem;]

   (c) Any optimal solution cannot include the activity with the latest finish time;

(d) The maximum number of activities that can be selected is always increased by one;

(e) None of the above.

6. (**4 points**) Which of the following statements about shortest path are true?

(a) In Bellman-Ford algorithm, the edges can be relaxed in any given order;

(b) Greedy shortest path algorithms can work on a graph with negative weight edges;

(c) In a directed acyclic graph, shortest path distance between any pair of vertices is always well-defined;

(d) If path $P1$ is the shortest path between u and v, and $P2$ is the shortest path between v and w, then the shortest path between u and w is $P1 + P2$.

(e) None of the above

7. (**4 points**) Which of the following statements are true?

(a) Halting Problem is NP;

(b) Any NP problem can be solved in polynomial time by a non-deterministic algorithm;

(c) NP-Complete problems are not in NP-Hard problem set;

(d) If a polynomial time algorithm is found for a NP-complete problem, all NP-complete problems are solvable in polynomial time;

(e) None of the above.

8. (**6 points**) When building an AVL tree from scratch, keys are inserted in the order of:

$$31 \to 20 \to 12 \to 26 \to 28 \to 27,$$

plot the AVL tree after each key is inserted, and mark the type of rotation taken, if any, at each step.

9. (**6 points**) Construct the optimal Huffman code for the following set of symbols with associated frequencies: $\{a : 26, b : 4, c : 17, d : 38, e : 33, f : 15, g : 7, h : 14\}$

10. (**8 points**) For the weighted undirected graph in Figure 1,

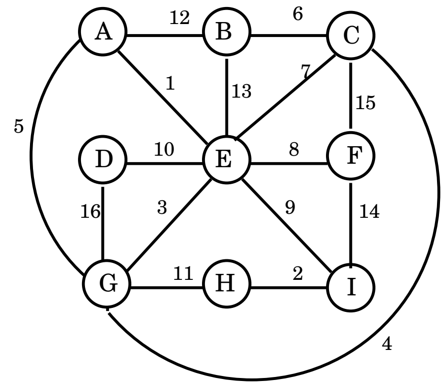

Figure 1: Weighted Undirected Graph for Question 10

(a) (**4 points**) what are the FIRST FIVE (5) edges added to the minimum spanning tree by the Prim's algorithm, starting at vertex A, in the order they are added?

(b) (**4 points**) what are the LAST FIVE (5) edges added to the minimum spanning tree by the Kruskal's algorithm, in the order they are added?

11. (**10 points**) Let A and B be two sequences of digits, each digit takes value from 0 to $k-1$. If a common subsequence between A and B has l digits, it can be treated as an l-digit number. The largest common subsequence is the one with the largest value. For example, with $k = 10$, $A = \{1, 9, 2\}$, B=$\{9, 1, 2\}$, the longest common subsequences are $\{1, 2\}$ and $\{9, 2\}$, but the largest common subsequence is just $\{9, 2\}$.

(a) (**6 points**) Design a dynamic programming algorithm to find the largest common subsequence between two sequences of n and m digits respectively. Your algorithm should output the value of the largest common subsequence. What is the complexity of your algorithm?

(b) (**4 points**) Apply your algorithm to find the largest common subsequence between $\{1, 2, 3, 9, 4, 6, 7, 5\}$ and $\{2, 8, 3, 1, 4, 6, 9, 5\}$, with $k = 10$.

Answer: a) From the lecture, we know that any common subsequence between $A[1 \cdots n]$ and $B[1 \cdots m]$ must be one of the following:

(a) a common subsequence between $A[1 \cdots n-1]$ and $B[1 \cdots m]$;

(b) a common subsequence between $A[1 \cdots n]$ and $B[1 \cdots m-1]$;

(c) a common subsequence between $A[1 \cdots n-1]$ and $B[1 \cdots m-1]$ concatenated by x, if $A[n] = B[m] = x$.

Let $L[n, m]$ be the largest common subsequence between $A[1 \cdots n]$ and $B[1 \cdots m]$. It satisfies the following recurrence:

$$L[n, m] = \max\{L[n-1, m], L[n, m-1]\}, \quad \text{if } A[n] \neq B[m] \tag{1}$$

$$L[n, m] = \max\{L[n-1, m], L[n, m-1], k * L[n-1, m-1] + x\}, \quad \text{if } A[n] = B[m] = x \tag{2}$$

Dynamic programming algorithm can be developed using the above recurrence, starting with base case of $L[i, 0] = L[0, j] = 0$, for $1 \leq i \leq n$ and for $1 \leq j \leq m$.

The complexity of the algorithm is simply $\theta(nm)$. **1 points**.

12. (**10 points**) In Homework 10, Q3, we developed a greedy algorithm for making changes with the smallest number of coins out of quarters (25¢), dimes (10¢), nickels (5¢), and pennies (1¢). Prove the greedy algorithm is optimal by following the three steps below:

(a) (**3 points**) greedy algorithm is optimal for making changes with only nickels (5¢), and pennies (1¢);

(b) (**3 points**) greedy algorithm is optimal for making changes with only dimes (10¢), nickels (5¢), and pennies (1¢);

(c) (**4 points**) greedy algorithm is optimal for making changes with quarters (25¢), dimes (10¢), nickels (5¢), and pennies (1¢)

Answer: a) To make changes for x cents, if an optimal solution is not greedy, i.e., the number of nickels is less than $\lfloor x/5 \rfloor$, then number of pennies will be larger than 5, by replacing 5 pennies with one nickel, the number of coins reduces by 4, contradiction! So optimal solution must be the greedy solution.

Answer: b) If $x < 10$, we will only use nickels and pennies, from a), greedy is optimal.

For $x \geq 10$, if an optimal solution is not greedy, i.e., the number of dimes is less than $\lfloor x/10 \rfloor$, then the total value of nickels and pennies should be larger than or equal to 10; based on a), we will have at least 2 nickels, by replacing these two nickels with one dime, the number of coins reduces by 1, contradiction! So optimal solution must be the greedy solution.

Answer: c) If $x < 25$, we will only use dimes, nickels and pennies, from b), greedy is optimal.

if $x \geq 25$, if an optimal solution is not greedy, then the total value y of dimes, nickels and pennies should be larger than or equal to 25:

1) if $25 \leq y < 30$, according to b), we will have at least two dimes, and one nickel, replace them with one quarter, the total number of coins reduces by 2; contradiction!
2) if $y \geq 30$, according to b), we will have at least three dimes, replace them with one quarter plus one dime, the total number of coins reduces by 1; contradiction!

So optimal solution must be the greedy solution.

13. (**16 points**) Given an unweighted directed graph $G = (V, E)$, we define the Strongly Connected Component (SCC) distance from node u to node v as the least number of SCCs that one has to visit on the way from u to v. If u and v are in the same SCC, their SCC distance is zero, and if v is not reachable from u, the SCC distance from u to v is ∞.

(a) (**8 points**) design an $O(|V| + |E|)$ algorithm to find the SCC distances from a given source node s to all the other nodes in G;

(b) (**4 points**) show that the SCC distance from any node u to any node v equals to the shortest path length from u to v if and only if G is a DAG;

(c) (**4 points**) if we further define the weighted SCC distance from u to v as the total number of nodes in all the visited SCCs on the way from u to v, design another $O(|V| + |E|)$ algorithm to find the longest weighted SCC distances from a given source node s to all the other nodes in G.

(*You can use any algorithm we learned in class without writing out the detailed algorithm.*)

Answer: a) Let $G^c = (V^c, E^c)$ be the component DAG for G, and C_u and C_v be the SCCs of u and v, respectively. The SCC distance from u to v is just the shortest path distance from C_u to C_v in G^c. The following is the algorithm:

1) Run Kosaraju/Tarjan's algorithm to find SCCs of G, build the component DAG G^C, *complexity* $O(|V| + |E|)$;
2) Run BFS, starting from C_s, in G^c to find the shortest paths to all the other SCCs *complexity* $O(|V^c| + |E^c|) = O(|V| + |E|)$,
3) Return shortest path distance from C_s to C_v as the SCC distance from s to v.

The total complexity is $O(|V| + |E|)$.

Answer: b) ($\Rightarrow$**, ONLY IF**) If there a cycle in G, all nodes on the cycle belong to the same SCC, SCC distances between them will be zero. So if G is not DAG, SCC distance is not the same as the shortest path length;

($\Leftarrow$**, IF**) On the other hand, if G is a DAG, each node becomes an independent SCC, then G^c is isomorphic to G, the SCC distance equals the shortest path distance.

Answer: c) Let n_C be the number of nodes in SCC C. In the component DAG G^c, assign $-n_C$ as the link weights for all links going to node C. Let C_s be the SCC of node s. For any other node v, find the shortest path distance $d(C_s, C_v)$ from C_s to C_v in G^c using topological sort+one pass Bellman Ford relaxation, if $d(C_s, C_v) = \infty$, there is no path from s to v, the weighted SCC distance is 0, $d(C_s, C_v) < 0$, $-d(C_s, C_v)$ is the weighted SCC distance. (since the link weights are all negative, if C_v is reachable from C_s, $d(C_s, C_v) < 0$ always.)

Finding shortest path algorithm in weighted DAG is $O(|V^c| + |E^c|) = O(|V| + |E|)$.

If you use up the space under any particular problem, you can write your answer here. On the page of the problem, tell us part of your work is here and write down the page number of this page and. Don't tear off any pages.

If you use up the space under any particular problem, you can write your answer here. On the page of the problem, tell us part of your work is here and write down the page number of this page and. Don't tear off any pages.

If you use up the space under any particular problem, you can write your answer here. On the page of the problem, tell us part of your work is here and write down the page number of this page and. Don't tear off any pages.