CSCI-B 505 Applied Algorithms (3 cr.)

Dr. H. Kurban
Computer Science
School of Informatics, Computing, and Engineering

Indiana University, Bloomington, IN, USA

December 13, 2021

Directions

- This final exam has a total of 18 questions:
  - 6 multi-choice questions (Problem 1.1 - 1.6)
  - 10 non-multi choice problems (Problem 2 - 11)
  - 2 programming questions (Programming Problem 1&2)
- The exam is worth 100 points.
- All the work should be your own. In other words, the exam is individual and collaboration is not allowed.
- Write your answers and Python programs in a Jupyter Notebook file. Upload the .pdf and .ipynb versions of it to Canvas as your final exam. Write clearly and neatly.
- You are not allowed to post questions related to the final exam on Canvas (Piazza/Discussion). If you think that any of the questions is ambiguous, answer it as you understand and explicitly explain your approach.
- This take-home final exam is due Thursday Dec 16, 2021 09:00a.m (ET). OBSERVE THE TIME. Absolutely no final exam will be accepted after that time.
- **Good luck!**

Multi Choice

Problem 1.1 [2 pt.]

Consider an undirected graph $G = (V, E)$ with edge costs that are all positive and distinct. Let T be a minimum spanning tree for graph G. Let P be a shortest path from s to t, where $s, t \in V$. Which of the following statements is correct?

(A) Suppose each edge weight is increased by 1, the minimum spanning tree T will not change

(B) Suppose each edge weight is increased by 1, the shortest path P between node s and t will not change

(C) The edge with heaviest edge cost cannot be in minimum spanning tree T

(D) The edges included in shortest path P will also be in minimum spanning tree T

Problem 1.2 [2 pt.]

Which of the following statements about a tree of n nodes is wrong?

(A) A tree is a connected graph with at most $n - 1$ edges

(B) Removing the root node is the only way to disconnect the tree

(C) Adding one edge in the tree will create exactly one cycle

(D) There is one unique path between every pair of vertices

Problem 1.3 [2 pt.]

Which of the following statements about a Binary Search Tree with distinct n elements is wrong?

(A) The left child is always lesser than its parent

(B) The smallest element is always at the leaf of the tree

(C) A search operation takes $O(n)$ time

(D) Every non-root node has exactly one parent

Problem 1.4 [2 pt.]

The algorithm in concern has a time complexity of $O(n^2)$. The algorithm takes 40 seconds to run with input size 10. How much time it will require to run with input size 30?

(A) 2 minutes

(B) 4 minutes

(C) 5 minutes

(D) 6 minutes

Problem 1.5 [2 pt.]

Consider the following unsorted array of strings, $\mathcal{R}$ = {cat, him, ham, bat}. What will be the order of elements of $\mathcal{R}$ after the first iteration of LSD Radix sort if we want to sort the array in an ascending order?

(A) {ham, him, cat, bat}

(B) {him, ham, cat, bat}

(C) {him, ham, bat, cat}

(D) {ham, him, bat, cat}

Problem 1.6 [2 pt.]

Using `Brute-Force` pattern matching algorithm, which pattern below will give you the fastest match with the given string "`abacaabaccabacabaabb`"?

(A) `bacab`

(B) `cabac`

(C) `baabb`

(D) `bacca`

Problem 2 [5 pt.]

Let $G = (V, E)$ be a connected directed graph with non-negative edge weights. Let s and t be vertices of G. We delete c (not a constant number) edges in G to get a subgraph H. Suppose we want to add back exactly one deleted edge into H, so that the shortest path from s to t in the resulting graph is as short as possible.

Describe an algorithm that chooses the best edge to reinsert. Your algorithm should run in $O(m \log n)$ time in which m is the number of edges and n is the number of vertices in G.

Hint 1: Dijkstra's algorithm runs in $O(m \log n)$ time on graph G.

Hint 2: Number of deleted edges c is not a constant number, so running Dijkstra's algorithm c times will result in $O(c\, m \log n)$ running time.

Problem 3 [5 pt.]

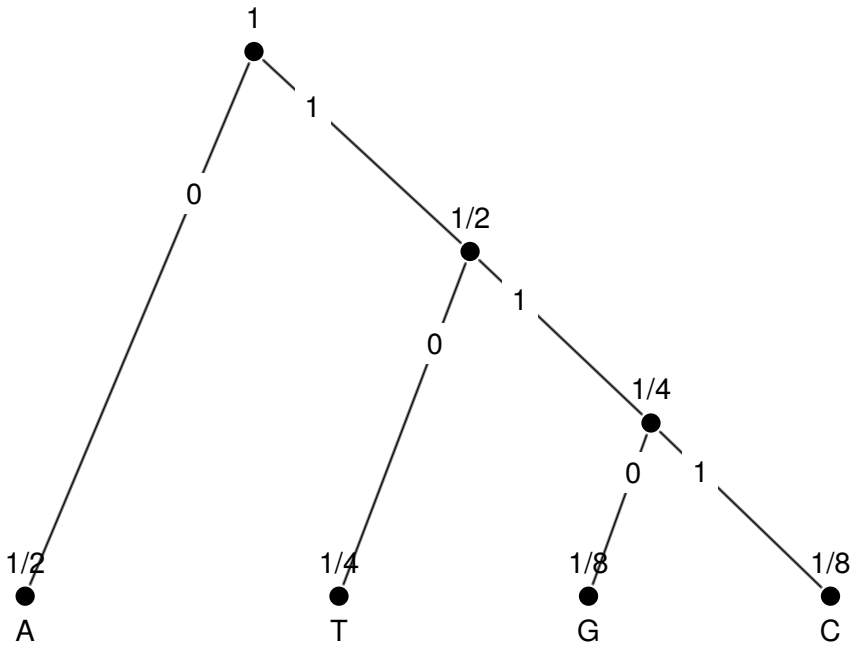
Let $G = (V, E)$ be a connected undirected graph with m edges and n nodes. If we remove a node v from G, we get a subgraph G_v. Given a graph G and a node v, how can you know if subgraph G_v is connected or not? Please describe a $O(m + n)$ algorithm to decide the connectivity of G_v.

Problem 4 [5 pt.]

A long string (10,000 characters) consists of four characters: A, T, G, C; they appear with frequency $1/2$, $1/4$, $1/8$ and $1/8$, respectively.

(a) What is the Huffman encoding for each character in the alphabet? Draw the Huffman encoding tree for each step of the algorithm. The final Huffman encoding tree should look like the following tree. You may draw by hand and upload a picture.

(b) Originally each character is represented in two bits. How many bits are saved if we use the new encoding scheme we obtained in the previous question?



Problem 5 [5 pt.]

Let $G = (V, E)$ be a undirected graph with m edges and n nodes; and let $T = (V, E')$ be its minimum spanning tree. All edge weights are distinct. Assume we remove an edge $(u, v) \in E'$ from graph G. Describe an algorithm to update the minimum spanning tree in $O(m)$ time.

1. **Hint 1:** The removal of edge (u, v) disconnects the spanning tree T in to two set of nodes A and B. Without a loss of generality, we may assume $u \in A$ and $v \in B$.

2. **Hint 2:** We need to find the smallest weight edge whose one end node in A and the other end node in B.

3. **Hint 3:** In order to get full marks, you need to state clearly what algorithm you use to find set A and B; what data structure you use to efficiently look up for target edge; and what are the running time of each portion of your algorithm.

Problem 6 [5 pt.]

We would like to solve word search puzzle problem in which we are given a N by N character board and N words. In the word search puzzle, the N words may be placed horizontally, vertically, or diagonally in the world puzzle board. Your task is to find all the N words.

If you are not familiar with word search puzzle, please refer to `https://thewordsearch.com` for more details.

Please describe a $O(N^3)$ algorithm to solve this problem. You need to specify what data structure you use and what are the time complexity at each step.

Problem 7 [5 pt.]

```
1  def calc(n):
2      k = 0
3      i = n/2
4      while(i<=n):
5          i = i + 1
6          j = 2
7          while(j<=n):
8              j = j * 2
9              k = k + n/2
10     return k
```

What is the time complexity of the function `calc` in big-Oh? What is the return value k in terms of big-Oh notion? For full credit please show your work step by step.

Problem 8 [5 pt.]

What are the advantages of `QuickSort` over `MergeSort`? If we want to sort a linked list of elements, which one between them should we choose? Please explain your answer.

Problem 9 [5 pt.]

Consider a pattern matching problem with string s = “GCATGACTGCGTGACC" and pattern p= “CTGC". Using the `Boyer-Moore` algorithm find the lowest index i within s at which the matched (with p) substring begins. Follow the simplified `Boyer-Moore` algorithm step by step to find the value of i as well as demonstrate how many comparisons are made.

Problem 10 [5 pt.]

Consider an array of n positive integers, $\mathcal{A} = \{a_1, a_2, ..., a_n\}$ and a positive integer m. Using a top down dynamic programming method, design an algorithms that answers us if some subset of $\mathcal{A}$ add up to m where you can use each a_n at most once.
For example, a list of 5 positive integers $\mathcal{A} = \{3, 7, 11, 15, 21\}$ and the given integer $m = 25$. Your designed problem will answer `True` as a subset of $\mathcal{A}$, $\{3, 7, 15\}$, sums to the given value of m.
Prove the correctness of your algorithm and show the computation time with regards to big-Oh notion.

Problem 11 [5 pt.]

Consider the following completed dynamic programming table, L, for a `Longest Common Subsequence problem` for string X with length 10 and string Y with length 12.

X\Y	-1	0	1	2	3	4	5	6	7	8	9	10	11
-1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	1
1	0	0	1	1	2	2	2	2	2	2	2	2	2
2	0	0	1	1	2	2	2	3	3	3	3	3	3
3	0	1	1	1	2	2	2	3	3	3	3	3	3
4	0	1	1	1	2	2	2	3	3	3	3	3	3
5	0	1	1	1	2	2	2	3	4	4	4	4	4
6	0	1	1	2	2	3	3	3	4	4	5	5	5
7	0	1	1	2	2	3	4	4	4	4	4	4	6
8	0	1	1	2	3	3	4	5	5	5	5	5	6
9	0	1	1	2	3	4	4	5	5	5	6	6	6

- What is the length of the longest common subsequence? Explain. Answer this question on the perspective of just by looking at the values of the table L.
- Show the path, in $L(i, j)$, that a tracking algorithm will take to recover the longest common subsequence starting from $L(9, 11)$. Given: the characters of X with indices $\{4, 5, 6, 7, 8, 9\}$ match with the characters of Y with indices $\{0, 3, 4, 5, 7, 11\}$.

Note: Traverse the 2D array starting from $L[9][11]$. Do following for every cell $L[i][j]$:

- If characters (in X and Y) corresponding to $L[i][j]$ are same (Or $X[i-1] == Y[j-1]$), then include this character as part of LCS and move to $L[i-1][j-1]$ cell.
- Else compare value of $L[i][j]$ with $L[i-1][j]$ and $L[i][j-1]$ and go in direction of greater value.

Programming Problem 1 [20 pt.]

This programming question involves three previous lab sessions: Mergesort, Kruskal's Algorithm and Depth-firth Search. At the beginning, you were given a detailed representation of a graph. You need to first find the minimum spanning tree of a given graph. Then we add one edge on the MST which will create a cycle. Your second task is to find the cycle on the modified MST. Please refer the details in file "MST-student.ipynb".

Programming Problem 2 [18 pt.]

A modified knapsack problem is that for a given set of positive integers $\{a_1, a_2, ..., a_n\}$, a knapsack of size s, find a subset A of $\{a_1, a_2, ..., a_n\}$ such that the sum of elements in A is the largest, but at most s. Write a program using `Python` to solve this problem using a top down dynamic programming method.
Test your program for the following modified knapsack problem:
Input list: $\{5, 23, 27, 37, 48, 51, 63, 67, 71, 75, 70, 83, 889, 91, 101, 112, 121, 132, 137, 141, 143, 147, 153, 159, 171, 181, 190, 191\}$ with knapsack size $s = 762$. Print out a subset along with the sum of its elements so that the sum has the closest distance to 762.