Fall 2020 CS330 Final – A

December 16, 2020

Instructions

- This exam is open book. You may consult your textbook, notes, slides, or other course materials, but no external sources.
- **Absolutely no collaboration is allowed**. You cannot discuss the contents of this exam with anyone except the instructors.
- You may reference and use anything (e.g., algorithms, proofs) discussed in lecture, lab or the textbook. Outside references (i.e., anything other than your notes and the course materials) are not allowed.
- The exam consists of 3 problems, each with several parts.
- You can write your answers by hand on paper, on a tablet, or you can type your solutions. Make sure to clearly mark the problems and also annotate with the problem number when you upload to Gradescope.
- From the time you download this PDF you have **120 minutes** time to upload your final pdf to Gradescope. There is a 10-minute grace period. If you have trouble uploading, please email your PDF immediately to `ads22@bu.edu`.
- If you have questions, you mask them via private post on Piazza or to a course staff member via the Zoom link posted on Piazza.

Problem 1 *Short answers* ***(19 points)***

(a) **(3 points)** Consider an instance of the Stable Matching Problem in which one of the hospitals (call it h^*) and one of the residents (call them r^*) both rank each other first. Prove or disprove: this hospital and resident will be paired together in every stable matching.

(b) **(3 points)** Consider the following recurrence:

$$T(n) = 8 \cdot T\left(\left\lceil \frac{n}{2} \right\rceil\right) + O(n^2) \quad \text{and} \quad T(1) = 1.$$

(a) What is the (asymptotic) depth of the corresponding recurrence tree?

(b) How many leaves does the tree have (asymptotically)?

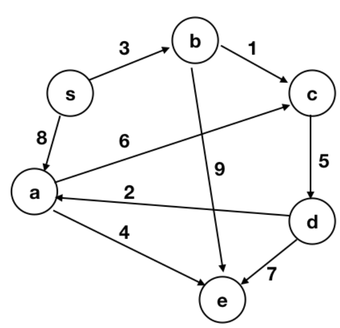
(c) What is the solution to the recurrence (use any method you like). Justify your answer briefly (one sentence).

(c) **(2 points)** Suppose we perform DFS in a **connected, undirected** graph. Which of the following types of edges are possible? Select all that apply:

- Tree edges
- Forward edges (i.e. from a node to one of its descendants)
- Back edges (i.e. from a node to one of its ancestors)
- Cross edges (i.e. from a node to a node that is neither its descendant nor its ancestor)

Whichever subset you selected, draw a connected, undirected graph in which all of them occur. Indicate the order in which DFS explores the vertices, and the type of each edge.

(d) **(2 points)** Consider the following directed graph. Is the tree of shortest paths from s unique? If so, highlight the edges in the tree. If not, give a vertex to which there are two distinct shortest paths from s.



(e) **(2 points)** Prove or disprove: Let G be an undirected graph with unique edge weights. The minimum spanning tree of G includes the minimum-weight edge in every cycle in G.

(f) **(1 points)** Suppose we have a directed graph with edge weights that may be negative or positive. Under what condition will there exist well-defined shortest path distances from vertex s to all other vertices (which you could then find using Bellman-Ford, for example)?

(g) **(2 points)** Suppose you have a flow network $G = (V, E)$, such that every capacity c_e is an *even* number. True or false: there exists a maximum flow f on G that is *even* on every edge. Either prove your answer or give a counter example.

(h) **(4 points)**

Consider the following two problems from class:

Max-Network-Flow (NF): On input of a graph $G = (V, E, c)$ with integer capacities for the edges, nodes s and t, and a number k, determine if there is a valid s-t flow with value at least k.

Halting: On input a program $\langle p \rangle$ and string x, determine if $p(x)$ will eventually halt.

Suppose we have some problem X of unknown complexity.

For each of the following, answer Yes or No, and give a brief (one sentence) justification.

(a) Suppose there is a reduction *from* HALTING *to* X (that is, an algorithm for HALTING that uses a hypothetical algorithm for X as a subroutine). Can we conclude that X is undecidable?

(b) Suppose there is a reduction *from* NF *to* X (that is, a polynomial-time algorithm for NF that uses a hypothetical algorithm for X as a subroutine). Can we conclude that X has a polynomial-time algorithm?

Problem 2 *DP* ***(9 points)***

To celebrate the end of the semester, you are buying presents for your two nieces at a toy store. There are n different toys available. You may decide to buy 0, 1, or 2 copies of each toy.

Each toy has a price p_i. It also comes with two happiness values $h_{i,1}$ and $h_{i,2}$. If you buy them one copy of the toy, their happiness will increase by $h_{i,1}$; if you buy two copies, their happiness will increase by $h_{i,2}$. [1] Each toy also has a price p_i; buying two copies costs $2p_i$.

Your goal is to pick which and how many toys to buy so that you stay within your budget B while making your nieces as happy as possible.

Here is an incomplete description of a dynamic programming algorithm `BuyToys`. Read the description and answer the questions with regard to *this* algorithm.

Input: budget B in dollars (integer), arrays p and h of integers such that $p[i] = p_i$ and $h[i,j] = h_{i,j}$ (where i goes from 1 to n, and j is either 1 or 2).

Output: H, the total happiness achieved by your purchase.

Algorithm Description: For each i between 0 and n, and b between 0 and B, let $OPT(i,b)$ denote the maximum happiness you can create by purchasing a subset of the toys $\{1, ..., i\}$ with a budget of b dollars.

The idea is to consider toys in order 1 through n and consider budgets in \$1 increments from $b = 0$ to B. For each pair of values i, b, the algorithm computes $OPT(i,b)$ (using the values computed previously).

(a) **(1 points)** For which values of i and b does $OPT(i,b)$ denote the final output H? (No explanation needed.)

(b) **(5 points)** Complete the recursive formula below for $OPT(i,b)$. In the formula, a three-way decision is made about each toy i: "don't buy", "buy one", or "buy two". (No explanation, just the formulas or pseudocode.)

$$OPT(i,b) = \begin{cases} 0 & \text{if } i = 0 \text{ or } b = 0 \\ \boxed{} & \text{if } b < p_i \\ \boxed{} & \text{if } p_i \le b < 2p_i \\ \boxed{} & \text{if } 2p_i \le b \end{cases}$$

[1] Warning: Money doesn't actually buy happiness (but toys can be fun).

(c) (3 points) Here is a specific input to your problem. :

$B = \$4$

$p = [\$1, \$3, \$1]$

$h[:, 1] = [1, 4, 2]$ (happiness from one copy)

$h[:, 2] = [2, 7, 5]$ (happiness from two copies)

Compute (and write) all entries of the 4×5 memoization table M , where $M[i, b]$ corresponds to $OPT(i, b)$. Format your answer as follows (we have filled in the base cases):

	0	1	2	3	4
0	0	0	0	0	0
1	0				
2	0				
3	0				

Problem 3 *Distributing vaccines* ***(11 points)***

During the pandemic, there is a global push to vaccinate as many people as quickly as possible. There are K different brands of vaccine available that need to be distributed to L countries. For each vaccine v_i there are q_i doses available. Further, we know the population p_j of each country c_j. However, not every vaccine is approved for use in every country. Currently, for each vaccine v_i there is a list ℓ_i of countries where it is approved.

Find an algorithm to decide whether there is a way to immunize every person in every country with a vaccine approved by their country's health department.

Your algorithm should use a reduction to network flow. Specify your algorithm by answering the questions below. Your answers should be clear enough that any other student could turn them in to working code. (Pictures are helpful.)

(a) **(4 points)** Design a directed graph $G(V, E)$ that is an instance of the max-flow problem. That is, G is directed, it has designated source s and sink t nodes and a capacity $c(e)$ on each directed edge. Describe each part of G:

- Specify the vertices in G. *Hint: there should be nodes for both vaccines and countries.*
- Describe the edges in G. Make clear which direction they are.
- Describe the capacity on each edge.
- Make sure there are vertices named s (source) and t (sink)

(b) **(1 points)** How many vertices does your graph have in terms of K and L (asymptotically)?

(c) **(2 points)** What is the maximum number of edges your graph can have, in terms of K and L (asymptotically)?

(d) **(2 points)** Suppose we found the maximum flow f in G. Describe how to find for each i, j the number of units of the vaccine v_i to be sent to country c_j given the value $f(e)$ along each edge. (Just state how to deduce the value. No need to proof.)

(e) **(2 points)** Suppose that it is indeed possible to get each country its required amount of vaccine. In that case, what is the value of the flow that is returned by the max-flow subroutine?

.