Midterm II

Name: Ran Liao

SID: 3034504227

Name and SID of student to your left:

CAROLYN L. WANG 3032952028

Name and SID of student to your right:

Exam Room:

☐ Evans 10 ■ Wheeler 150 ☐ North Gate 105 ☐ Hearst Field Annex A1 ☐ VLSB 2060
☐ Cory 540AB ☐ Other

Please color the checkbox completely. Do not just tick or cross the box.

Rules and Guidelines

- **The exam is out of 110 points and will last 110 minutes.**
- Answer all questions. Read them carefully first. Not all parts of a problem are weighted equally.
- Write your student ID number in the indicated area on each page.
- Be precise and concise. **Write in the solution box provided.** You may use the blank page on the back for scratch work, but it will not be graded. Box numerical final answers.
- The problems may **not** necessarily follow the order of increasing difficulty. ***Avoid getting stuck on a problem.***
- Any algorithm covered in lecture can be used as a blackbox. Algorithms from homework need to be accompanied by a proof or justification as specified in the problem.
- Good luck!

Discussion Section

Which of these do you consider to be your primary discussion section(s)? Feel free to choose multiple, or to select the last option if you do not attend a section. **Please color the checkbox completely. Do not just tick or cross the boxes.**

- ☐ Antares, Tuesday 5 - 6 pm, Mulford 240
- ☐ Kush, Tuesday 5 - 6 pm, Wheeler 224
- ☐ Arpita, Wednesday 9 - 10 am, Evans 3
- ☐ Dee, Wednesday 9 - 10 am, Wheeler 200
- ☐ Gillian, Wednesday 9 - 10 am, Wheeler 220
- ☐ Jiazheng, Wednesday 11 - 12 am, Cory 241
- ☐ Sean, Wednesday 11 - 12 am, Wurster 101
- ☐ Tarun, Wednesday 12 - 1 pm, Soda 310
- ☐ Jerry, Wednesday 1 - 2 pm, Wurster 101
- ☐ Jierui, Wednesday 1 - 2 pm, Etcheverry 3113
- ☐ Max, Wednesday 1 - 2 pm, Etcheverry 3105
- ☐ James, Wednesday 2 - 4 pm, Dwinelle 79
- ☐ David, Wednesday 2 - 3 pm, Barrows 140
- ☐ Vinay, Wednesday 2 - 3 pm, Wheeler 120
- ☐ Julia, Wednesday 3 - 4 pm, Wheeler 24
- ■ Nate , Wednesday 3 - 4 pm, Evans 9
- ☐ Vishnu, Wednesday 3 - 4 pm, Moffitt 106
- ☐ Ajay, Wednesday 4 - 5 pm, Hearst Mining 310
- ☐ Zheng, Wednesday 5 - 6 pm, Wheeler 200
- ☐ Neha, Thursday 11 - 12 am, Barrows 140
- ☐ Fotis, Thursday 12 - 1 pm, Dwinelle 259
- ☐ Yeshwanth, Thursday 1 - 2 pm, Soda 310
- ☐ Matthew, Thursday 2 - 3 pm, Dwinelle 283
- ☐ Don't attend Section.

1 True/False. (22 pts)

(a) In every connected graph in which there is more than one edge of minimum cost, there is more than one minimum spanning tree

○ True ● False

(b) If a connected graph has a cycle in which all the edge costs are the same, then the graph has more than one minimum spanning tree

● True ○ False

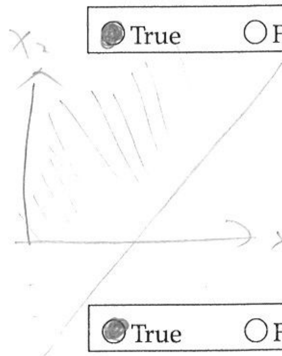
(c) **(8 pts)** Consider the following LP:

$x_2 = x_1 - 3$

$$\max x_1 + x_2$$
$$x_1 - x_2 \leq 3$$
$$x_1, x_2 \geq 0$$

$y_1 \geq 1 \quad -y_1 \geq 1$

$y_1 < -1$



(i) The point $(x_1, x_2) = (0, 5)$ is feasible for the LP

$0 \leq 3+5 = 8$

● True ○ False

(ii) The point $(x_1, x_2) = (3, 0)$ is a vertex for the feasible region of the LP

● True ○ False

(iii) The linear program is bounded

○ True ● False

(iv) The dual of the linear program is feasible

○ True ● False

(d) Given an undirected graph, the shortest path between any two nodes will belong to some minimum spanning tree of the graph.

○ True ● False

(e) **(6 pts)** For the following, consider a network G with an $s-t$ flow f. We say an edge in G is *saturated* when the flow across the edge is equal to the capacity.

(i) If f is a maximum flow, then f saturates all the edges going into t

○ True ● False

(ii) Let f be a maximum flow, and let G' be the result of removing all edges in G that are saturated by f. There is no path from s to t in G'.

● True ○ False

(iii) Let f be any flow (not necessarily a maximum flow), and let G' be the result of removing all edges in G that are saturated by f. If there is no path from s to t in G', then f is a maximum flow in G.

○ True ● False

(f) Let $G = (V, E)$ be an undirected complete graph with edge weights given by w_{uv}. Define the subproblem $d[i, j]$ to be the length of the shortest path from vertex i to vertex j, with the following recurrence relation:

$$d[i,j] = \min\left(w_{ij}, \min_{v\in V}\left(d[i,v] + d[v,j]\right)\right), \quad \text{if } i \neq j$$
$$d[i,i] = 0$$

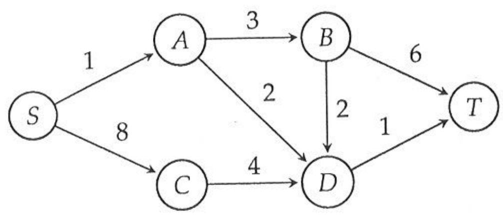
Given only G, there exists a dynamic programming algorithm that uses this recurrence relation to compute $d[i, j]$ for all $i, j \in V$.

○ True ● False

2 Go With the Flow.

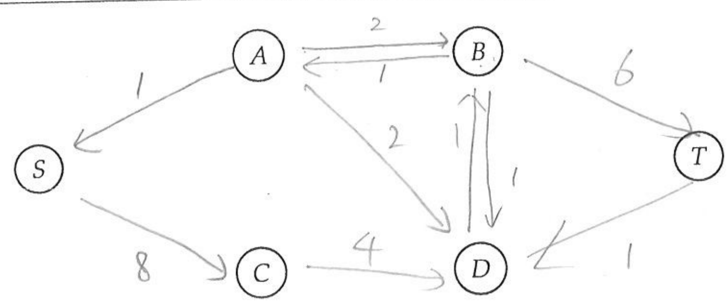
(10pts) Recall that the Ford-Fulkerson algorithm iteratively uses the residual network to compute the max flow.

(a) **(6 pts)** Consider the following network:



Let f be a flow that assigns 1 unit of flow on the path $S \to A \to B \to D \to T$ and 0 flow elsewhere.

(i) Draw the residual graph for f.



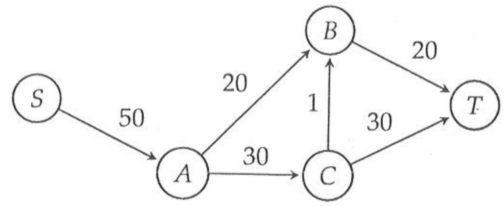
(ii) Give a path in the residual graph from S to T which can accommodate additional flow.

$S \to C \to D \to B \to T$

(iii) What is the value of the maximum flow on this network?

2

(b) **(4 pts)** Consider the following network:



For the next questions, consider all possible sequences of iterations of Ford-Fulkerson.

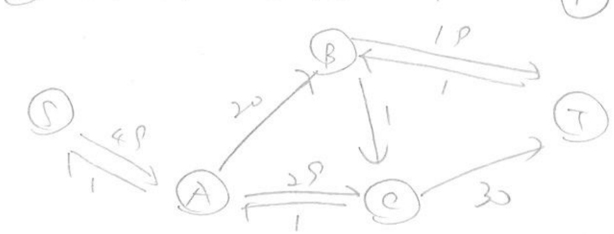
(i) What is the minimum number of iterations until there is no longer any path from S to T in the residual network?

2.

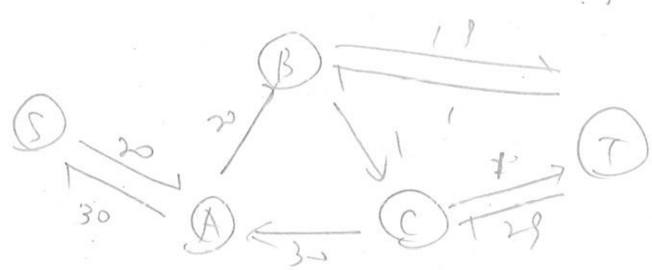
(ii) What is the maximum number of iterations until there is no longer any path from S to T in the residual network?

4

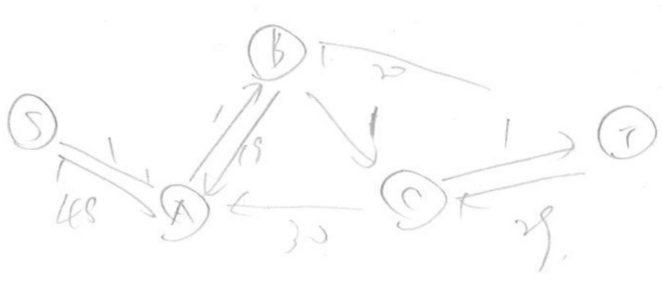
① (S → A → C → B → T ①



② S → A → C → T 29



③ S → A → B → T 19.



3 Grab bag. (14 pts)

(a) **(4 pts)** Formulate the dual of the linear program below in terms of the variables y_1 and y_2.

$$\begin{aligned} &\max x_1 + x_3 \\ y_1 \quad & x_1 - x_2 \le 5 \\ y_2 \quad & x_1 + 3x_2 - x_3 \le 2 \\ & x_1, x_2, x_3 \ge 0 \end{aligned}$$

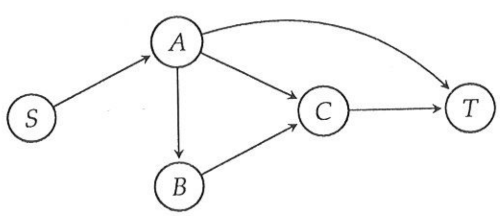
$(y_1+y_2)x_1 + (-y_1+3y_2)x_2 + (-y_2)x_3$
$\le 5y_1 + 2y_2$

$\min \; 5y_1 + 2y_2$
$y_1 + y_2 \ge 1$
$-y_1 + 3y_2 \ge 0$
$-y_2 \ge 1$
$y_1, y_2 \ge 0$

(b) **(4 pts)** Recall the dynamic programming algorithm that finds the shortest path from s to t in an unweighted DAG. Let $P(u)$ denote the length of the shortest path from u to t, and recall that:

$$P(u) = \min_{(u,v)\in E} 1 + P(v)$$
$$P(t) = 0$$

Suppose the algorithm was run on the DAG below. Give **an order** in which the subproblems will be solved from first to last (e.g. "$P(S), P(B), P(C), \ldots$")



$P(T), P(C), P(B), P(A), P(S)$

(c) **(2 pts)** Consider the following set of clauses defining a HORN-SAT formula $\varphi(w, x, y, z, a, b)$:

$$\Rightarrow x, \quad x \Rightarrow y, \quad x \wedge y \Rightarrow w, \quad y \wedge w \Rightarrow z, \quad (x \vee \overline{w} \vee y \vee \overline{z} \vee \overline{a} \vee \overline{b})$$

How many satisfying assignments are there for φ?

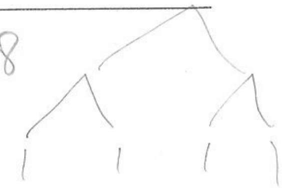
(d) **(4 pts)** We have a primal LP with objective max $x_1 + 4x_2$, and a dual LP with objective min $y_1 - 2y_2 + 0.5y_3$. Suppose that the points $(0, 0.5)$ and $(3, 1, 2)$ are feasible in the primal and dual respectively. Argue in two or fewer sentences that they are also optimal for their respective LPs.

$x_1 + 4x_2 = 0 + 4 \cdot 0.5 = 2$ $y_1 - 2y_2 + 0.5y_3 = 3 - 2 + 1 = 2$

Since they are equal, both of them are optimal

7 1 2 3 5

1 1 2 3 5 8



4 Bounding Codewords.

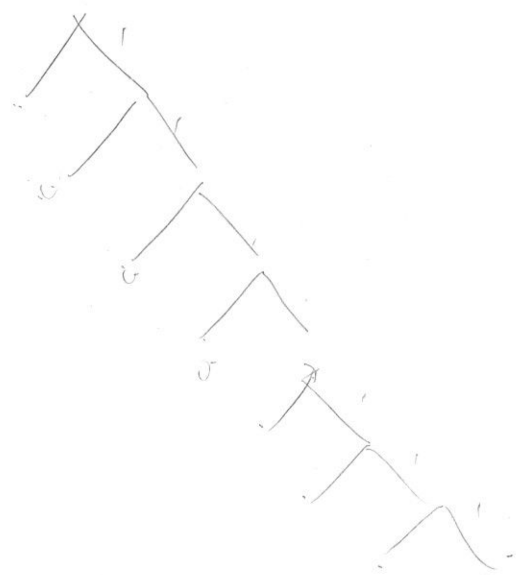
(**6 pts**) Consider the Huffman encoding for an alphabet of characters $c_1, \ldots, c_n$ such that $n = 2^k$ for integer $k \geq 1$, with respective frequencies $f_1, \ldots, f_n$. For each of the following characters, provide as a function of n the minimum and maximum possible codeword length for all sets of f_i such that $0 < f_1 \leq \ldots \leq f_n < 1$.

(a) The most frequent character, c_n

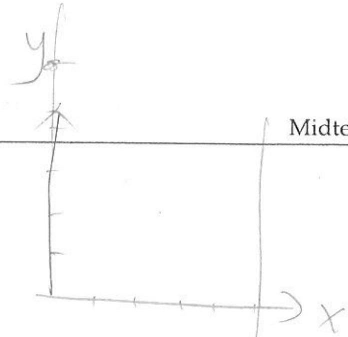
Minimum: 1 Maximum: log(n)

(b) The least frequent character, c_1

Minimum: log(n) Maximum: n − 1



7



5 Tilted.

$2 = x + y$

$-x + 2 = y$

(8 pts) Consider the following family of linear programs parameterized by some real number c:

$$\begin{aligned} \max x + y \\ x \leq 5 \\ y \leq 5 + cx \\ x, y \geq 0 \end{aligned}$$

$0 \leq 5 + 3c$

$-1 \leq c$

For each of the questions below, describe all possible values of c, e.g. "$c \leq -3$ or $5 < c \leq 10$ or $c = 15$". If there are no possible values, write "none". If all real values are possible, write "all".

(a) For what values of c is the point $(x, y) = (5, 0)$ feasible?

$c \geq -1$

(b) For what values of c does the LP have infinitely many optimal points?

$c = -1$

(c) For what values of c is the LP unbounded?

none.

(d) For what values of c is the dual LP bounded? (*Hint:* you do not need to construct the dual to answer this question)

all.

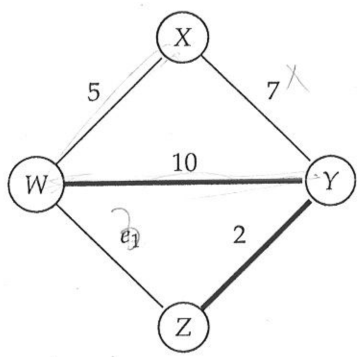
6 Happy Little Trees.

(**10 pts**) For each of the following graphs, all e_i can vary between 1 and 100, inclusive.

For each bolded edge below, mark **exactly one of** ABCD:

- Mark A if for all possible values of the e_i's, every MST contains the bolded edge
- Mark B if for all possible values of the e_i's, only some MSTs (at least one, but not all) contain the bolded edge
- Mark C if for all possible values of the e_i's, no MST contains the bolded edge
- Mark D if none of the above

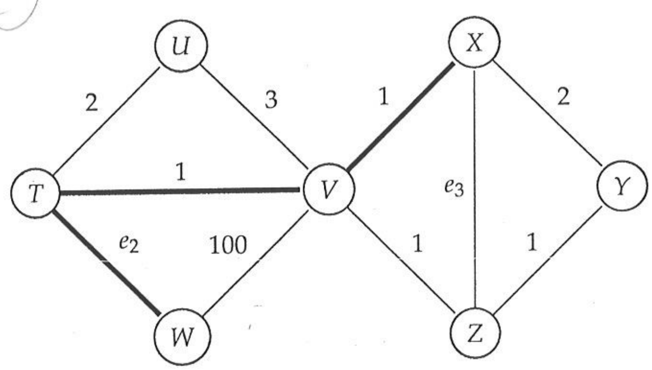
(a)



(i) The edge $\{W, Y\}$ — ○A ○B ●C ○D

(ii) The edge $\{Z, Y\}$ — ○A ○B ○C ●D

(b)



(i) The edge $\{T, V\}$ — ●A ○B ○C ○D

(ii) The edge $\{T, W\}$ — ○A ○B ○C ●D

(iii) The edge $\{V, X\}$ — ○A ○B ○C ●D

7 Sparse Graph MSTs.

(12 pts) You are given a weighted, connected, undirected graph $G = (V, E)$ such that $|E| = |V|$. Give an algorithm that finds a minimum spanning tree in time $O(|V|)$.

(a) Main Idea

Since $|E| = |V|$, there must exist a cycle in the graph. And remove one edge within this cycle will give a spanning tree.
Therefore run DFS or BFS to identify the cycle.
The remove the edge within the cycle with the largest weight.

(b) Runtime analysis

DFS or BFS will cost $O(|V|+|E|) = O(2|V|) = O(|V|)$ time.
And iterating through all edges within this cycle will take at most $O(|E|) = O(|V|)$ time.
Therefore the overall runtime is $O(|V|)$

8 Points on a Line.

(14 pts) You are given a set S of n points on a line. The points are given to you in sorted order. You want to find a set C of points of minimum size such that that every point in S is at distance at most 1 from at least one point in C. (Note that the points in C need not belong to S.) You would like a greedy algorithm that runs in time polynomial in n and finds an optimal solution.

For example, given the points $S = \{2.7, 3.2, 3.6, 4, 4.9, 5.2\}$, a possible solution is $C = \{3.5, 5\}$, and there is no smaller solution.

(a) Describe your greedy algorithm (3 sentences or less)

Scan S from left to right, find the first point that is not satisfied with the requirement. Denote this point as S_i and put S_i+1 into C. Keep doing this until all points are satisfied.

(b) Argue for its correctness using an exchange argument.

Suppose my solution is $C_0 = \{C_1, C_2 \dots C_n\}$
And there's another solution $C_s = \{C_1', C_2' \dots C_n'\}$
Suppose C_i / C_i' are the first point that be different in these two solution. By my greedy approach, $C_i' < C_i$. Otherwise some points will not be satisfied. So change C_i' to C_i will improve C_s and possibly give a better solution. Therefore greedy approach is optimal.

(c) What's the runtime of your algorithm?

$O(n)$

many
?
$P(m_1 \ldots m_n\ B)$

9 I am the Machine(s).

(14 pts) You have a mission-critical production system that consists of n stages that must be completed in sequence. Each stage i can only be completed by a machine of type M_i. Unfortunately, the machines are faulty, and machines of type M_i fail with probability f_i (you may assume all failures are independent of each other).

A system with one copy of each machine type will succeed with probability $(1-f_1)\cdot(1-f_2)\cdots(1-f_n)$. By adding redundant machines of type M_i, stage i is not completed only if all machines of type M_i fail. Therefore, if we have m_i machines of type M_i, the probability that stage i is completed is $1-f_i^{m_i}$ and the probability that the whole system succeeds is now $\prod_{i=1}^{n}(1-f_i^{m_i})$. Notice that if **any** of the $m_i = 0$, the success probability will be zero.

Unfortunately, you only have B dollars to spend on machines, and it costs c_i dollars to purchase each machine of type M_i. You may assume both B and the c_i's are positive integers.

Give a dynamic programming algorithm that finds the maximum achievable success probability while staying under budget. *Your algorithm should compute a probability, not specific values for each m_i.*

More formally:

You are given probabilities $f_1, \cdots, f_n$, costs $c_1, \ldots, c_n$, and nonnegative integer budget B. Give a dynamic programming algorithm to compute the maximum success probability $\prod_{i=1}^{n}(1-f_i^{m_i})$ where each m_i is a nonnegative integer and $\sum_i c_i m_i \le B$.

(a) Define your subproblems.

$P(m_1, m_2 \ldots m_n, B)$ represents the probability that have B money left and m_i copy of machine type i

(b) What are the base cases?

if $B < \min(c_i)$, $P(m_1, m_2 \ldots m_n, B) = \prod_{i=1}^{n}(1-f_i^{m_i})$

(c) Write the recurrence relation for the subproblems.

$$P(m_1, m_2 \ldots m_n, B) = \max \begin{cases} P(m_1+1, m_2 \ldots m_n, B-c_1) & \text{if } B-c_1 \ge 0 \\ P(m_1, m_2+1, \ldots m_n, B-c_2) & \text{if } B-c_2 \ge 0 \\ \vdots \\ P(m_1, m_2 \ldots m_n+1, B-c_n) & \text{if } B-c_n \ge 0 \end{cases}$$