Q1 6

550FF8C3-7F13-42BF-9E0B-812573367E00

csci-570-fa-22-exam-2

#342 2 of 12

1) 16 pts

Mark the following statements as **TRUE** or **FALSE** by circling the correct answer. No need to provide any justification.

[TRUE/FALSE] False

In dynamic programming, solving for the value of the optimal solution to subproblems is performed in the bottom up pass.

[TRUE/FALSE]

Given a maximum flow f in a flow network G with m edges, we can determine if G has a unique min-cut in $O(m)$ time.

[TRUE/FALSE]

For a flow network G, if a flow network G' is constructed from G by increasing the edge capacity of each edge in G by 1, then the value of a maximum flow in G' is at most g units more than the value of a maximum flow in G, where g is the number of edges leaving the source in G.

[TRUE/FALSE]

The running time of a pseudo-polynomial time algorithm can be upper bounded by a polynomial function of the size of the problem input (or output).

[TRUE/FALSE]

Every iteration-based dynamic programming algorithm with n^2 unique subproblems has running time $\Omega(n^2)$.

[TRUE/FALSE]

Ford-Fulkerson can be used to find the maximum size matching between two strings in polynomial time.

[TRUE/FALSE]

The problem of checking whether a given flow f of a flow network G is a maximum flow can be solved in linear time.

[TRUE/FALSE]

Let G be a flow network with source s and sink t. Let (A, B) be a maximum cut in G. Let G' be a flow network obtained by adding 1 to the capacity of each edge in G. Then (A, B) must also be a maximum cut in G'.

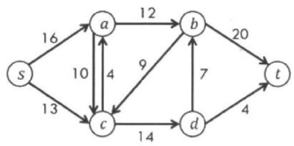
Q2 4

2) 16 pts

I. Consider a bipartite graph G with m edges and $2n$ nodes whose node set is partitioned into two sets X and Y with the property that every edge in G has one end in X and the other in Y.
To find the maximum size matching in G, Ford-Fulkerson is guaranteed to terminate after at most how many iterations? Select the smallest correct upper bound. (4 pts)

a) n
b) $2n$
c) $2n^2$
d) mn

II. Which of the following s-t cuts are min-cuts in the flow network below? Circle all correct answers. (4pts)



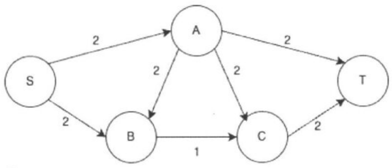
a) $A = \{s, c, d\}$, $B = \{a, b, t\}$
b) $A = \{s, a, c, d\}$, $B = \{b, t\}$
c) $A = \{s, a, c\}$, $B = \{b, d, t\}$
d) None of the above

EBC1A7A4-8822-4FA1-B69D-A15585ED1894

csci-570-fa-22-exam-2

#342 4 of 12

III. Consider the flow network G below with source S and sink T. Suppose we are using the Ford-Fulkerson algorithm to find a maximum flow in G. Suppose that the first augmenting path used by the Ford-Fulkerson algorithm is S-A-B-C-T. How many distinct possible second augmenting paths pass through the vertex A? (4 pts)



a) 1
b) 2
c) 3
d) 4

IV. Suppose a certain problem is described by the following inputs:

- A string of length m.
- An array of integers of size n, where the maximum integer in the array is C.

Which of the statements below are true? Circle all true statements. (4 pts)

a) An algorithm for solving the problem with running time $\Theta(nm)$ is called a linear time algorithm.
b) If for all input instances $C = 10$, then an algorithm with running time $\Theta(nC)$ is strongly polynomial.
c) If for all input instances the average of the integers is 1, then an algorithm with running time $\Theta(n*Sum(Array))$ is strongly polynomial.
d) An algorithm with running time $\Theta(n\log m)$ is called a pseudo-polynomial time

Q3 10

1C7D2F2B-A1AC-4B67-9165-0ADB47F0B01B

csci-570-fa-22-exam-2

#342 5 of 12

3) 16 pts

You have found a hidden treasure that contains *n* diamonds placed in a row. You know each of these diamonds' values [v_0, v_1, ..., v_{n-1}]. Ideally, you would have taken all the diamonds, but this treasure is cursed, and the cave will collapse if you pick up 3 consecutive diamonds. Knowing this, find the maximum total value of the diamonds you can pick up such that no three consecutive diamonds are picked.

a) Define (in plain English) the subproblems to be solved. (4 pts)

Let OPT(i) be the maximum total value of (i...n) diamonds keeping the constraint.

4/4

b) Write a recurrence relation for the subproblems (4 pts)

1/4 incorrect recurrence

$OPT(i) = \max(OPT(i+1), v_i + OPT(i+3))$

$OPT(i) = OPT(i+1)$ if $v_{i+1} > v_i + v_{i+3}$

else

$OPT(i) = OPT(v_i + OPT(i+3))$

7A2679A8-AD7C-4688-8DB9-71F7276C5FCA

csci-570-fa-22-exam-2

#342 6 of 12

c) (6 pts for part c)
Using the recurrence formula in part b, write pseudocode using iteration to find the maximum total value of diamonds you can obtain without having the cave collapse. (4 pts)
Make sure you specify base cases and their values (2 pts)

OPT(0) = v_0

OPT(1) = v_0

OPT(2) = max(v_0, v_1, v_2)

OPT(3) = max(v_0, v_1, v_2)

3/6 no base cases, incorrect loop range

```
for i = 1 to n:
    OPT(i) = max(OPT(i+1), v_i + OPT(i+3))
end for
return OPT(n)
```

d) What is the time complexity of your solution? Is your algorithm efficient? (2 pts)

O(n)

2/2

Yes.

Q4 4

16F32EE5-DB17-426D-A1A3-044BC8AF6AAF
csci-570-fa-22-exam-2
#342 7 of 12

4) 18 pts
Let $G = (V, E)$ be a directed graph, and let s and t be two distinct non-adjacent vertices in G.

(a) Design a polynomial-time algorithm for finding a set of nodes C of minimum size such that $s \notin C$ and $t \notin C$ and deleting C from G disconnects s from t so that there is no directed path from s to t. (12 pts)

We will run any maximum flow ~~network~~ algorithm. After finding the maximum flow, we ~~get~~ will get the minimum cut. We will remove all the nodes (except s and t) whose edges are a part of minimum cut. These would be our set of nodes C.

(b) Prove the correctness of your algorithm. (6 pts)

If there ~~are~~ ~~no nodes w~~ is a maximum flow of f and lets say the corresponding min cut is {S,t}. Now since the flow is going through the edges involved in this minimum cut, there exists an s-t path. The moment there is no s-t path present we know that the maximum flow algorithm terminates. So, removing the nodes whose edges are a part of minimum cut, will make sure that there is no path directed from s to t. Conversely, if we are removing the nodes whose edges are a part of the minimum cut, there won't be any ~~no~~ flow and thus no path directed from s to t.

Q5 13

0072B890-13CA-4C23-9CC9-9BBA84F30148

csci-570-fa-22-exam-2

#342 8 of 12

5) 16 pts

A researcher is deciding which programs to run on a supercomputer. The researcher is given a sequence of n programs. Each program i is associated with three positive integers: s_i, w_i, and t_i. If the researcher chooses to run program i, then the supercomputer will be occupied for t_i units of time, during which the supercomputer performs w_i units of work, and the researcher will be unable to run programs $i+1$ through $i+s_i$. The total time the supercomputer is occupied cannot exceed a given positive integer T. Design a dynamic programming algorithm for selecting a valid subsequence of programs for the researcher to run such that the supercomputer performs a maximal amount of work.

a) Define (in plain English) the subproblems to be solved. (4 pts)

Let OPT(i,T) denote the maximal amount of work the supercomputer can perform using (1.... i) programs within time T.

instead of 1..i use i..n -1

b) Write a recurrence relation for the subproblems (4 pts)

$OPT(i,T) = \max(OPT(i+1, T), OPT(i+S_i+1, T-t_i)+\omega_i)$

$OPT(i,T) = OPT(i+1, T)$ if the researcher decides not to run i^{th} program i.e $\omega_{i+1} > \omega_i$

else $OPT(i,T) = OPT(i+S_i+1, T-t_i) + \omega_i$.

B757BAE2-DA58-437B-974D-784CB528D53F
csci-570-fa-22-exam-2
#342 9 of 12

c) (6 pts for part c)
Using the recurrence formula in part b, write pseudocode using iteration to find the maximum amount of work that can be accomplished. (4 pts)
Make sure you specify base cases and their values (2 pts)

$OPT(0,0) = 0$ $\quad OPT(1,T) = w_1$ if $T \geq t_1$ ~~else~~
$OPT(i,0) = 0$
$OPT(0,T) = 0$ $\quad = 0$ else

```
for i = 2 to n:
    for t = t_2 to
        OPT(i, t) =
    endfor
endfor
return OPT(n, T)
```

-1 for incorrect first loop -1 for incorrect return statement **-2**

d) What is the time complexity of your solution? Is your algorithm efficient? (2pts)

$O(nT)$. This is not an efficient algorithm.
It is pseudopolynomial because of T.

Q6 6

5183999C-46EC-442C-AAE1-E947FFC033EF

csci-570-fa-22-exam-2

#342 10 of 12

6) 18 pts

Let G be a flow network. An edge e in G is called *critical* if it crosses every min-cut in G, in other words it goes from set A into set B in every (A,B) min-cut. And it is called *pseudo-critical* if it crosses at least one min-cut in G.

a) Design an efficient algorithm for determining whether a given edge e is critical. (9 pts)

We run an efficient max flow algorithm and then remove that one edge because if there is no flow / no s-t path, that edge would be a critical edge as if it's present in every min cut, removal of it will result in no flow. Even this algorithm will take polynomial time.

Incorrect algorithm 2

b) Design an efficient algorithm for determining whether a given edge e is pseudo-critical. (9 pts)

We run max flow on flow network G and find all the min-cuts in G. We keep on removing edges in the min cuts because of which the max flow reduces. ~~All these~~ After removing one edge if the flow is reduced then that edge is pseudo critical else it's not. This will take polynomial time and we would be using an efficient max flow algorithm. We keeping on repeating the process to find all the pseudo critical edges. We can also just run max flow and then randomly keep on removing the edges. The edges because of which the max is reduced would be the pseudo critical edges.

Finding all min-cuts is not efficient. 4

5D99D781-1DD8-49A3-8B50-4A3EADE822DE

csci-570-fa-22-exam-2

#342 11 of 12

Additional Space

2878C209-2506-4C28-B65F-EC997A119D8A
csci-570-fa-22-exam-2
#342 12 of 12

Additional Space