CS-3510-C F23 Exam 3

Vidit Dharmendra Pokharna

TOTAL POINTS

87.5 / 115

QUESTION 1

Question 1 15 pts

1.1 (i) **3 / 3**

✓ **- 0 pts** *Correct*

- 3 pts Incorrect

1.2 (ii) **3 / 3**

✓ **- 0 pts** *Correct*

- 3 pts Incorrect

1.3 (iii) **3 / 3**

✓ **- 0 pts** *Correct*

- 3 pts Incorrect

1.4 (iv) **3 / 3**

✓ **- 0 pts** *Correct*

- 3 pts Incorrect

1.5 (v) **3 / 3**

✓ **- 0 pts** *Correct*

- 3 pts Incorrect

QUESTION 2

2 Question 2 **0 / 5**

- 0 pts Correct

✓ **- 5 pts** *Incorrect*

QUESTION 3

3 Question 3 **5 / 5**

✓ **- 0 pts** *Correct*

- 5 pts Incorrect

QUESTION 4

Question 4 20 pts

4.1 (a.1) **0 / 2.5**

- 0 pts Correct (Infinity)

✓ **- 2.5 pts** *Incorrect*

4.2 (a.2) **2.5 / 2.5**

✓ **- 0 pts** *Correct*

- 2.5 pts Incorrect

4.3 (b) **6 / 10**

- 0 pts Correct

- 2 pts Minor error (e.g. Knapsack with repetition)

✓ **- 4 pts** *Does not add cost $$C_j$$ OR does not include $$i - e_j$$*

- 5 pts Does not take minimum of two entries

- 8 pts Mostly incorrect

- 10 pts Missing

4.4 (c) **1 / 5**

- 0 pts Correct

- 1 pts Returns $$T[\frac{V + 1}{2}, n]$$

- 2 pts Minor error

✓ **- 4 pts** *Major error*

- 5 pts Missing

QUESTION 5

5 Question 5 28 / 30

- 0 pts Correct

- 0 pts Reverses table direction

Table Definition (5 pts)

- 0 pts Correct

- 1 pts Minor mistake

- 3 pts Mostly incorrect table definition.

- 5 pts Table Definition Missing/Completely incorrect

Base Cases and Recurrence (20 pts)

- 3 pts Does not include base case that accounts for going out of array bounds

- 2 pts Minor mistake in base case/ Missing one base case

- 4 pts Base Case Missing/Completely Incorrect

✓ **- 2 pts** *Minor mistake in recurrence*

- 5 pts Recurrence does not align with the table defined.

- 4 pts Explanation of correctness Missing

- 10 pts Recurrence Missing/Completely incorrect

- 8 pts Major mistake in recurrence relation

- 14 pts Non-DP Approach

- 20 pts Missing/Incorrect

Return Value (5 pts)

- 0 pts Return value is consistent with table definition and recurrence (even if incorrect)

- 2 pts Minor Error

- 5 pts Incorrect/Missing

- 8 pts Inefficient solution (higher than $$O(n^2)$$)

- 30 pts Missing Answer

1 The A's would be T. Moreover, T[i+1][j], and T[i+1][j+1]

QUESTION 6

6 Question 6 23 / 25

- 0 pts Correct

Table Definition (5 pts)

- 0 pts Correct

- 1 pts Minor mistake

- 3 pts Mostly incorrect table definition.

- 5 pts Table Definition Missing/Completely incorrect

Base Cases and Recurrence (20 pts)

- 2 pts Minor mistake in base case/ Missing one base case

- 4 pts Base Case Missing/Completely Incorrect

✓ **- 2 pts** *Minor mistake in recurrence*

- 5 pts Recurrence does not align with the table defined.

- 4 pts Explanation of correctness missing/unsatisfactory

- 10 pts Recurrence Missing/Completely incorrect

- 8 pts Major mistake in recurrence relation

- 14 pts Non-DP Approach

- 20 pts Missing

- 6 pts Inefficient solution (higher than $$O(n^2m^2)$$)

- 25 pts Missing Answer

Base cases should include:

T[0][j] = T[i][0] = 0 if i or j = 1 then this will cause issues (T[2, 1]), especially because it is unclear if i or j is odd if you take the floor or ceiling

Recurrence includes P[i][j] as the third option to find the max from, this is when we choose not to make a cut

QUESTION 7

7 Question 7 7 / 15

- 0 pts Correct

Table Definition (5 pts)

- 0 pts Correct

- 1 pts Minor mistake

- 3 pts Mostly incorrect table definition.

- 5 pts Table Definition Missing/Completely incorrect

Base Cases and Recurrence (10 pts)

- 2 pts Minor mistake in base case/ Missing one base case

- 4 pts Base Case Missing/Completely Incorrect

- 2 pts Minor mistake in recurrence

- 5 pts Major mistake in Recurrence/Recurrence does not align with the table defined.

- 4 pts Explanation of correctness Missing

✓ **- 8 pts** *Recurrence Missing/Completely incorrect*

- 8 pts Non-DP Approach

- 10 pts Missing

- 6 pts Inefficient (higher than $$O(n^3)$$)

- 15 pts Missing Answer

Needs a chain matrix approach

GTID: 903772087
NAME: Vidit Pokharna

Georgia Institute of Technology

Fall 2023

CS 3510 C – Design & Analysis of Algorithms
Exam 3 Version A

October 24, 2023 TIME ALLOWED: 75 MINS

Name: Vidit Pokharna

GTID: 903772087

GT Username: vpokharna3

INSTRUCTIONS TO STUDENTS

1. Please write your NAME and GTID clearly on all the pages.
2. This examination paper contains **SEVEN (7)** questions and comprises **TEN (10)** printed pages.
3. ONLY write on the front sheets of paper that are numbered. The backs will not be scanned.
4. Calculators are **NOT** allowed.

I am in aware of and the accordance with Academic Honor Code of Georgia Tech and the Georgia Tech Code of Conduct. I'll use no external help on this test. Also, I have read all the instructions on this page.

Signature: 

GTID: 903772087
NAME: Vidit

Problem 1 *(15 points; 3 points each)*

Indicate whether the following statements are ***true*** or ***false***.

(i) The return value of a Dynamic Programming algorithm using tabulation must be the final entry in the table, such as $T[n]$ or $T[n][m]$.

Answer: False

(ii) The runtime of a Dynamic Programming algorithm using tabulation is always lower bounded by the size of the table. For example, if the table size is $n \times n$, the runtime must be $\Omega(n^2)$.

Answer: True

(iii) In the Longest Palindromic Subsequence problem, $T[i][j]$ represents the longest palindromic subsequence of $X[1 \ldots i]$ and $X[1 \ldots j]$.

Answer: False

(iv) The below pseudocode correctly illustrates how we fill in table entries for the Chain Matrix Multiplication problem (assume 1-indexing and that `range` is inclusive of both bounds):

```
for i in range(1, N):
    for j in range(i, M):
        T[i][j] = ...
```

Answer: False False

(v) 0-1 Knapsack and Knapsack with Infinite Copies share the same runtime and recurrence.

Answer: False

GTID: 903772087
NAME: Vidit

Problem 2 *(5 points)*

You might recall the *minimum interval sum* problem from one of your Divide-and-Conquer homeworks. As it turns out, there's a Dynamic Programming solution that's faster than the divide and conquer approach. Here's the problem again to refresh your memory:

You are given an array of integers A of length n. The *minimum interval sum* is defined as the minimum sum of contiguous elements across all ***nonempty*** intervals $A[i \ldots j]$ for $i \leq j$. For example, given the array $A = [-16, 2, -10, 2, 43, -20]$, the minimum interval sum would be $(-16) + 2 + (-10) = -24$, which would correspond to the interval from $i = 0$ to $j = 2$ (inclusive).

Assume that $T[i]$ represents the minimum interval sum of $A[1 \ldots i]$ ending at $A[i]$. Choose the recurrence relation that yields the correct answer. You do not need to explain your answer.

Write the correct letter option (a, b, c, ...) in the given blank: d

a. $T[i] = T[i-1] + \min(A[i], 0)$

b. $T[i] = \min\{T[i-1] + A[i], 0\}$

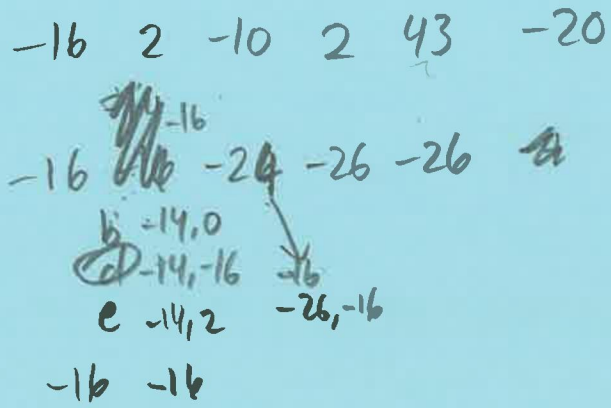
c. $T[i] = \max\{T[i-1] + A[i], 0\}$

d. $T[i] = \min\{T[i-1] + A[i], T[i-1]\}$

e. $T[i] = \min\{T[i-1] + A[i], A[i]\}$

f. $T[i] = \max\{T[i-1], A[i]\}$

g. $T[i] = \min\{T[i-1], A[i], T[i+1]\}$



GTID: 903772087
NAME: Vidit

Problem 3 *(5 points)*

Consider the following problem:

Input : An array $V = [v_1, v_2, ..., v_n]$ of positive integers, and a natural number $K > 0$.
Output : "Yes" if one can make change for K cents using coins of denominations in V, using as many coins as needed from each denomination, and "No" otherwise.

Suppose you are creating the table T, where

$$T[k] = \begin{cases} 0, \text{ if you } \textbf{cannot} \text{ make change for } k \text{ cents using coins of denominations in } V \\ 1, \text{ if you } \textbf{can} \text{ make change for } k \text{ cents using coins of denominations in } V \end{cases}$$

Select the correct recurrence relation for filling in the entries of T. For simplicity, assume that $T[d]$ returns 0 for any negative d. You do not need to explain your answer.

Write the correct letter option (a, b, c, ...) in the given blank: d

1, [illegible] 5 10 25

a. $T[k] = \max_{1 \le j \le k}\{1 + T[v_j]\}$

b. $T[k] = \max\{T[k-1], v_k + T[k-1]\}$

c. $T[k] = \max_{j<k}\{T[k - v_j]\}$ ✓ $4 - v_1$

d. $T[k] = \max_{1 \le j \le n}\{T[k - v_j]\}$ ✓ $4 - v_2$ $4 - v_3$ $4 - v_4$

e. $T[k] = \min_{1 \le j \le k}\{1 + T[v_j]\}$

f. $T[k] = \min\{T[k-1], v_k + T[k-1]\}$

0	1	2	3	4					
0	0	1	0						

GTID: 903772087
NAME: vidit

Problem 4 *(20 points)*

A certain candidate is running for office, and wants to know how to optimally spend her campaign funds. There are V electoral votes distributed across n districts, according to $E = [e_1, e_2, \ldots e_n]$ so that the ith district has e_i votes and $\sum_i e_i = V$.If our candidate spends money advertising in a district, she will definitely win all of that district's votes. However, it costs a different amount to advertise in each district: $C = [c_1, c_2, \ldots, c_n]$ gives the cost, in dollars, to advertise in each district, so that c_i is the cost to advertise in district i.

Given that V is odd, we want to determine the minimum advertising cost for our candidate to win a strict majority of votes: i.e., at least $(V + 1)/2$ votes.

Example: If you have $V = 99$ and $n = 5$, with $E = [26, 10, 41, 3, 19]$ and $C = [30, 11, 14, 9, 8]$, then your algorithm should output "22", since this is the minimum cost to win a majority of votes, achieved by advertising in districts 3 and 5. This costs $c_3 + c_5 = 22$ and gets $e_3 + e_5 = 60$ votes which is a majority (we would need 50 for a majority since $V = 99$).

Assume that $T[i, j]$ represents the minimum advertising cost of winning at least i votes over districts $1 \ldots j$.

a. (5 points) Define the base cases.

$T[i, 0] =$ 0 $\forall 1 \leq i \leq V$ and $T[0, j] =$ 0 $\forall 0 \leq j \leq n$

b. (10 points) Write the recurrence for filling table entries.

$T[i, j] =$ $\begin{cases} \min(dp[i,j-1],\ dp[i - E[i], j-1]) & \text{if } E[i] \geq i \\ dp[i,j-1] & \text{otherwise} \end{cases}$

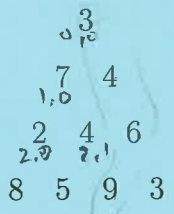
c. (5 points) Explain how you would return the correct value from your filled table.

Return $T[V, n]$

GTID: 903772087
NAME: Vidit

Problem 5 *(30 points)*

Our monkey took a vacation from Homework 6, and he greatly apologizes for missing. However, while on vacation he was on a game show to get some more bananas. He was given a pyramid of banana bunches A, where $A[i][j]$ represents the number of bananas in the banana bunch at position (i, j). The monkey starts at position $(0, 0)$, and at each step, the monkey can only go directly down-right or down-left to get another bunch. Design an algorithm to find the maximal amount of bananas he can pick up from the top of the pyramid to the bottom row. Consider the following example:



(In array form, the input is given as A = [[3], [7, 4], [2, 4, 6], [8, 5, 9, 3]])

The monkey's best path through this pyramid would be $3 + 7 + 4 + 9 = 23$ total bananas. The indices of this path are $(0, 0), (1, 0), (2, 1), (3, 2)$.

(i) (5 points) Define the entries of your table in words. E.g. $T[i]$ or $T[i][j]$ is ...

T[i][j] = max number of bananas collected at position (i,j)

T is a 2D array of size (len(A)-1) x len(A[len(A)-2])

GTID: 903772087
NAME: Vidit

(ii) (20 points) State the base cases and recurrence for entries of the table in terms of smaller subproblems. Briefly explain in words why it is correct.
Full points will be awarded to the most efficient solutions.

There is no need for base cases since we will start from the second to last row, which will take values from the last row.

len (A) = n

$T[i][j] = A[i][j] + \max(A[i+1][j], A[i-1][j+1])$

recurrence

The recurrence uses the two predecessors of a node by looking at one level below at the j and j+1 value and adding the max of those 2 until we get to the top

(iii) (5 points) Describe how you would compute the output of the algorithm from the filled table.

The answer would be stored in the element T[0][0] as we are working our way up

GTID: 903772087
NAME: Vidit

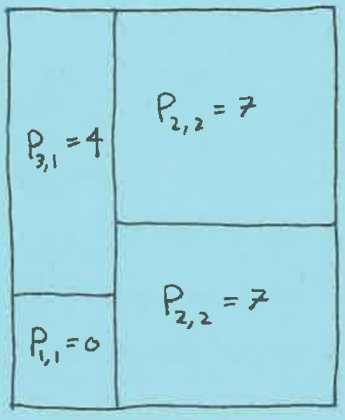
Problem 6 *(25 points)*

Suppose you are given a sheet of metal of dimensions $n \times m$. You are also given a set of prices of rectangles where $P_{i,j}$ is the price you can get for a sheet of metal of size $i \times j$. If $P_{i,j} = 0$ then you will earn no money for that piece. You may only repeatedly make one cut (either vertical or horizontal) to subdivide a sheet into two pieces. Design an algorithm to return the maximum money you can earn by cutting your sheet into rectangles to sell. **Note:** $P_{i,j} = P_{j,i}$, and P is 1-indexed.

Input: $n = 4, m = 3$, $P = \begin{bmatrix} 0 & 2 & 4 \\ 2 & 7 & 5 \\ 4 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Output: 18

Explanation: Make the first cut vertically subdividing the 4×3 sheet into one 4×1 sheet and 4×2. We will then horizontally subdivide the 4×1 sheet into a 3×1 and a 1×1 sheet. We will finally horizontally subdivide the 4×2 sheet into a 2×2 sheet and a 2×2 sheet. All cuts are visualized below. The final sum is $0 + 4 + 7 + 7 = 18$.



(i) (5 points) Define the entries of your table in words. E.g. $T[i]$ or $T[i][j]$ is ...

T[i][j] = maximum money that can be earned by cutting an i x j metal sheet to sell

T is a 2D array of size (n+1) x (m+1)

GTID: 903772087
NAME: vidit

(ii) (20 points) State the base cases and recurrence for entries of the table in terms of smaller subproblems. Briefly explain in words why it is correct.
Full points will be awarded to the most efficient solutions.

1-indexed: $dp[1][1] = P_{1,1}$

recurrence

$$H = \max_{j/2 \le k < j} (dp[i][k] + dp[i][j-k])$$

$$V = \max_{i/2 \le k < i} (dp[k][j] + dp[i-k][j])$$

$$dp[i][j] = \max(H, V)$$

This solution works because we are comparing the pairs that total the current position horizontally and vertically (so we leave one band constant) and find the max of the pairs in both directions, and then max of those values to find the maximum money possible to make an $i \times j$ metal.

GTID: 903772087
NAME: Vidit

Problem 7 *(Extra Credit; 15 points)*

Suppose you are given a string $a_1 ... a_n$, design an algorithm to compute the minimum number of cuts to break the string into pieces such that each piece is a non-empty palindrome.

Input: *aba*
Output: 0

Input: *abaabab*
Output: 2

(i) (5 point) Define the entries of your table in words. Which element(s) of the table do you return?

dp[i][j] = minimum number of cuts to break string into palindromic (non-empty) pieces for $a_i, \ldots, a_j$

(ii) (10 points) State the base cases and recurrence for entries of the table in terms of smaller subproblems. Briefly explain in words why it is correct.
Full points will be awarded to the most efficient solutions.

base

$dp[i][i] = 0 \quad \forall i \leq n$

1 indexed.

END OF EXAM

$$dp[i][j] \overset{10}{=} \begin{cases} \min(dp[i+1][j], dp[i][j-1]) + 1 & a_i = a_j \\ \min(dp[i+1][j], dp[i][j-1]) + 1 & a_i \neq a_j \end{cases}$$

return dp[1][n]