CSE 102: Spring 2021

Quiz 5: Dynamic Programming (23 points)

45 minutes (Quiz) + 10 minutes (uploading) = 55 minutes
Start Time: 6:00pm; Submission Time: 6:55pm

1. **Canoe Rental Problem** (6 points)

   Consider the canoe rental problem from the homework. Let $c(i, j)$ be the cost of renting a canoe at trading post i and returning it at trading post j, where $i < j$. Assume that you always want to go down the river, so that the costs when $j < i$ are irrelevant. The cost matrix $c[i, j]$ is given below. The goal is to find the cheapest sequence of rentals that allow you to complete the trip.

	1	2	3	4	5
1	0	15	30	40	60
2	–	0	10	30	15
3	–	–	0	12	16
4	–	–	–	0	10
5	–	–	–	–	0

   Show all your work using Dynamic Programming Algorithm

   (a) Let $opt[k]$ be the the cost of cheapest sequence of canoe rentals starting at post 1 and ending at post k. Compute $opt[k]$ for $k = 4$.

   (b) Write down the optimal sequence of canoe rentals going from trading post 1 to trading post 4 (i.e. at what trading post numbers are canoe rentals made?).

   After the computation, fill out the column 4 of the following table (you do not need to compute for $k = 5$).

k	1	2	3	4	5
$opt[k]$	0	15	25		
Optimal Sequence	–	$1, 2$	$1, 2, 3$		

2. **Binomial Coefficient** (3 points)

Consider the following iterative version of computing Binomial Coefficient (discussed in the class).

DynamicBinomialCoefficient(n,k)

```
1. C[0] = 1
2. for i= 1 to k
3.     C[i] = 0
4. for j = 1 to n
5.    for i = k down to 1
6.        C[i] = C[i-1] + C[i]
7. return C[k]
```

A student is planning to use the above algorithm to compute binomial coefficient $\binom{n}{k} = \binom{5}{3}$, where $n = 5$ and $k = 3$.

Initial values of the array C has been provided after the execution of the lines 1-3 of the code. In addition, since the value of $C[0]$ does not change, it has been copied in each subsequent row. [Note that each row overwrites the row above.]

k	$\rightarrow$	0	1	2	3
j	0	1	0	0	0
$\downarrow$	1	1			
	2	1			
	3	1			
	4	1			
	5	1			

(a) There are 15 blank entries above. Indicate the order in which these entries will be filled out, by writing $1, 2, \cdots, 15$ inside the cell. Refer to these as cell number.

(b) In which cell number does the final solution appear?

(c) Which cell numbers contribute to the computation of the value inside the cell number 10?

[You do not need to compute the blank entries in the table.]

3. **0-1 Knapsack Problem** (7 points)

Consider the 0-1 knapsack problem where the thief wants to steal items with values $v = (v_1, v_2, ...v_n)$ with corresponding weights $w = (w_1, w_2, ...w_n)$ with the constraint that the total weight does not exceed the capacity W of the knapsack with the objective of maximizing total value of the goods stolen. For every item, the thief has to make a binary choice: 1 or 0, whether to take the item or leave it. The thief cannot break the item into smaller pieces.

A table is provided with boundary case and some table entries filled out for a specific instance of the problem with 5 objects. Weights and value of the objects are shown on the left hand side columns under w_i and v_i respectively. The maximum capacity is $W = 9$.

			W	$\rightarrow$								
i	w_i	v_i	0	1	2	3	4	5	6	7	8	9
1	3	5	0	0	0	5	5	5	5	5	5	5
2	4	3	0	0	0	5	5	5				$[2, 9]$
3	5	8	0	0	0	5	5					$[3, 9]$
4	6	6	0	0	0	5	5					$[4, 9]$
5	7	12	0	0	0	5	5					$[5, 9]$

A cell in the table is referred by row and column number. For example, [2,9] refers to row 2 and column 9.

(a) Compute the value for the cell [2,9]. Show your work. Which cell numbers will be used to compute this value?

(b) Compute the value for the cell [3,9]. Show your work. Which cell numbers will be used to compute this value?

(c) Compute the value for the cell [4,9]. Show your work. Which cell numbers will be used to compute this value?

(d) Compute the value for the cell [5,9]. Show your work. Which cell numbers will be used to compute this value?

(e) Which objects are stolen?

(f) What is the total weight of the objects stolen?

4. **Matrix Chain Multiplication** (7 points)

Consider the matrix-chain-multiplication problem discussed in the class. Let A_1, A_2, A_3, and A_4 be matrices of dimensions 12×5, 5×25, 25×10, and 10×20 respectively. Let M = $\{m_{ij}\}$ be the matrix, shown below that will store the least number of multiplications required to multiply the matrices using dynamic programming approach. We have filled out some of the entries in the table. There are only two remaining entries in the table.

	j 1	→ 2	3	4
i 1	–	1500	1850	
↓ 2	–	–	1250	
3	–	–	–	5000
4	–	–	–	–

(a) Which entry in the table will you fill out next? Compute this entry and show all your work.

(b) Compute the final entry. Show all your work.

(c) What is the parenthesization for minimizing the computation of matrix multiplication $A_1 A_2 A_3 A_4$?