CSE101 HW7

Yuhang Jiang

June 6, 2023

1.

1. Define the subproblems:
   $dp[i][j][k]$: The maximum value of only consider the first i groups, and there are j people in the upper and k people in the lower.

2. Define and evaluate the base cases:
   $dp[0][0][0] = 0$. Only consider the first 0 groups, the value is 0, and there are 0 people in the upper and 0 people in the lower.

3. Establish the recurrence for the tabulation.

   - Case 1: Put the current group on the upper if there is space on the upper.
     $dp[i][j][k] = dp[i-1][j-p[i]][k] + v[i]$
   - Case 2: Put the current group on the lower if there is space on the lower.
     $dp[i][j][k] = dp[i-1][j][k-p[i]] + v[i]$
   - Case 3: Ignore current group. $dp[i][j][k] = dp[i-1][j][k]$

4. Determine the order of subproblems:
   Order the subproblems from 0 to n.

5. Final form of output:
   $\min\{dp[n]\}$, the maximum value of considering all groups, regardless of how many people in the upper and lower.

6. Put it all together as pseudocode:

   $dp[0][0][0] = 0$, otherwise $dp[i][j][k] = -\infty$
   for $i = 1...n$ **do**
   for $j = 0...U$ **do**
   for $k = 0...L$ **do**
   if $j \geq p[i]$ **then**
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j-p[i]][k] + v[i])$
   if $k \geq p[i]$ **then**
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j][k-p[i]] + v[i])$
   $dp[i][j][k] = \max(dp[i][j][k], dp[i-1][j][k])$
   return $\min dp[n][0...U][0...L]$

7. Runtime analysis:
   The code contains a triple loop, the internal complexity of the triple loop is $O(1)$. The first loop is executed n times, the second loop is executed $U+1$ times, and the third loop is executed $L+1$ times.
   Total time: $O(nUL)$

2.

1. Define the subproblems:
   $dp[i][j][k]$ is the minimum height of a bookshelf: contains first i books, and the last layer of shelf has a height of j and a width of k.

2. Define and evaluate the base cases:
   $dp[0][0][0] = 0$, otherwise $dp[i][j][k] = \infty$. 0 book has only one placement, height is 0 and width is 0.

3. Establish the recurrence for the tabulation.

   - Case 1: Put the next book $(i+1)$ on the current shelf. Restriction: the sum width of the current shelf and next book $\leq W$
     $dp[i+1][\max(j, h[i+1])][k + w[i+1]] = dp[i][j][k] + \max(h[i+1] - j, 0)$
     $(k + w[i+1] \leq W)$

   - Case 2: Place the next book on a new shelf.
     $dp[i+1][h[i+1]][w[i+1]] = dp[i][j][k] + h[i+1]$

4. Determine the order of subproblems:
   Order the subproblems from 0 to n.

5. Final form of output:
   $\min\{dp[n]\}$, the minimum height of the bookshelf contains all books, regardless of the width and height of the last layer.

6. Put it all together as pseudocode:
   H: maximum height of all the books
   $dp[0][0][0] = 1$, otherwise $dp[i][j][k] = \infty$
   for $i = 0...n-1$ **do**
   for $j = 0...H$ **do**
   for $k = 0...W$ **do**
   if $dp[i][j][k] == \infty$ **then** continue
   if $k + w[i+1] \leq W$ **then**
   $dp[i+1][\max(j, h[i+1])][k + w[i+1]] = \min(dp[i+1][\max(j, h[i+1])][k + w[i+1]], dp[i][j][k] + max(h[i+1] - j, 0))$
   $dp[i+1][h[i+1]][w[i+1]] = \min(dp[i+1][h[i+1]][w[i+1]], dp[i][j][k] + h[i+1])$
   return $\min\{dp[n][0...H][0...W]\}$

7. Runtime analysis:
   The code contains a triple loop, the internal complexity of the triple loop is $O(1)$. The first loop is executed n times, the second loop is executed $H+1$ times, and

the third loop is executed $W + 1$ times.
Total time: $O(nWH)$.
Time complexity: $O(nW \max(h_i))$