Student Information

Jiarui HE
50013538

Problem 1

Analysis

State Transition Equation

$$dp[i][j] = \begin{cases} dp[i][j-1] & \text{if } j > 0, i = 0 \\ dp[i-1][j] & \text{if } i > 0, j = 0 \\ dp[i-1][j] + dp[i][j-1] & \text{if } i, j > 0 \end{cases}$$

Boundary Conditions

$$dp[0][0] = 1$$
$$0 \leq i < m, 0 \leq j < n$$

Calculation Order

From the Row 0 to Row $m-1$. For each row, from Column 0 to Column $n-1$.

Time Complexity

$\Theta(mn)$, each element of the grid should calculate the transition equation once. The calculation of each equation is $\Theta(1)$, and there are $m \times n$ elements.

Space Complexity

$\Theta(mn)$, to store the 2D array dp.

Problem 2

Analysis

State Definition

Let $dp[i][j]$ be the maximum total value when we consider the first i items and the total weight of used items is j.

Let v_i, w_i be the value and weight of the i-th (1-based) item.

State Transition Equation

$$dp[i][j] = \begin{cases} \max\{dp[i-1][j], dp[i-1][j-w_i] + v_i\} & \text{if } j \geq w_i \\ dp[i-1][j] & \text{otherwise} \end{cases}$$

Boundary Conditions

$$dp[0][0] = 0$$
$$\forall 0 < j \leq W, dp[0][j] = +\infty (\text{not necessary})$$
$$0 \leq i \leq n, 0 \leq j \leq W$$

Calculation Order

From 1-th items to n-th items. There is no requirement on the enumerate order of j . But we used to enumerate j from 0 to W .

Time Complexity

Ignored. Provided.

Space Complexity

$\Theta(nW)$ to store the 2-D array dp, while v_i and w_i are both $\Theta(n)$. Thus, the total space complexity is $\Theta(nW)$.

Problem 2 - Better Space Complexity Version

Analysis

It is difficult for me to describe this optimization with mathematical equations following the required structure. The principle is actually the same as the previous one.

Let $dp[j]$ be the maximum weights costing j capacity.

Then we calculate using the following pseudocode:

$$\begin{aligned} & dp[0] = 0 \\ & \text{for } i \leftarrow 1 \text{ to } n: \\ & \quad \text{for } j \leftarrow W \text{ to } w_i: \\ & \quad \quad dp[j] \leftarrow \max\{dp[j], dp[j-w_i] + v_i\} \end{aligned}$$

In the calculation loop, when $i = i_0, j = j_0$, $dp[0 \ldots j_0]$ represents $dp[i_0 - 1][0 \ldots j_0]$ in the previous version, while the $dp[j_0 + 1 \ldots W]$ represents $dp[i_0][j_0 + 1 \ldots W]$. After the state transition equation, $dp[j_0]$ represents the $dp[i_0][j_0]$. Furthermore, we remove the condition checking if $j \geq w_i$, since that when $j < w_i$, then $dp[i][j] = dp[i-1][j]$, which is $dp[j] = dp[j]$ and unnecessary.

Boundary Conditions

$$\begin{gathered} dp[0] = 0 \\ \forall 0 < j \leq W, dp[j] = +\infty (\text{not necessary}) \\ 1 \leq i \leq n, 0 \leq j \leq W \end{gathered}$$

Calculation Order

From 1-th items to n-th items. There is no requirement on the enumerate order of j . For each item, enumerate j from W to w_i .

Time Complexity

The same as the previous version

Space Complexity

$\Theta(W)$ for dp, while v_i and w_i are both $\Theta(n)$. Then the total complexity is $\Theta(n + W)$.