Module 5 Dynamic Programming Assignment

Tyler Kim
tkj9ep

November 12, 2022

Response 1

Given: n for the total number for doors and S for number or secured doors.

My algorithm would initialize an array of integers A of size $n+1$ and fill the first $S-1$ indices to 0 and $A[S]$ to 1. The algorithm will iterate through all the indices of A and update each element of the array such that index $A[i] = A[i-1] + 2^{i-S}$. The algorithm will return $A[n]$.

Response 2

Given:
$R = \{r_1, r_2, ..., r_n\} | r_i =$ number of minutes for skiing and $n =$ number of runs
$L =$ minutes available for skiing
$m =$ satisfactory time leftover time dissatification

$$twd(t) = \begin{cases} 0, & \text{if } t = 0 \\ -C, & \text{if } 1 \leq t \leq m \\ (t-m)^2, & \text{otherwise} \end{cases} \tag{1}$$

The algorithm first calculates D_{min} by greedily calculating the minimum number of days needed if each run was done back-to-back. A 2D array of integers of size $n \times D_{min}$, called A, is created and each index is filled with infinity such that $A[i][j] =$ minimum $twd(t)$ given i runs and j days. The algorithm will go through each element in A and fill each value according to the following conditions: if $j == 0 \rightarrow A[i][j] = twd(L - \sum_{k=0}^{i})$, if $i == j \rightarrow A[i][j] = A[i-1][j-1] + twd(L - r_i)$, and if $i > j \rightarrow min(A[i-x][j-y] + twd(L - \sum_{k=i}^{i-x} r_i)) | x =$ looping from i to 1 and $y =$ looping from j to 1. In other words, the last case will calculate minimum value for all possible cases. Finally, the algorithm will return the smallest value in the last column of A.

Response 3

Given: $w =$ max size of the tile cover

The algorithm will first create an array A of integers of size $w+1$ and assign $A[0] = 0, A[1] = 1$, and $A[2] = 2$. The algorithm will iterate through each of the cells after $A[2]$ and update each cell such that $A[i] = A[i-1] + A[i-2]$. Once the algorithm fully fills all the values in A, it will return the output of the value in the very last cell. The runtime of the algorithm is linear or $\Theta(w)$ because the algorithm has to loop through all w values.

Response 4

Let $f(w)$ denote the solution algorithm such that $f(w)$ denotes the number of possible combinations of dices in a $4 \times w$ grid. The algorithm has two base cases: if $w = 1$, return 1, and if $w \leq 0$, return 0. In the recursive call of the algorithm, it will return $f(w-1) + 4f(w-2) + f(w-4)$. The runtime of the algorithm will be $\Theta(n)$ since at each level of recursion, the function does constant work.