Dynamic Programming

Axel Ljungström

(Corrected) notes from tutorial 14/9-2023

Problem 1

Given a (non-empty) list of integers $L = [a_0, \dots, a_n]$, find the smallest sum consecutive sublist $L' = [a_j \dots a_{j+k}]$ w.r.t the sum $a_j + \cdots + a_{j+k}$.

Solution

Step 1: Characterise solution

Suppose we are given a solution, $L' = [a_j \dots a_{j+k}]$. We consider two cases

- Case 1 : The length of L' is 1. In this case, $L' = [a_{j+k}]$
- Case 2 : The length of L' is greater than one. In this case, the minimal sum is given by $a_{j+k} + \underbrace{(a_{j+k-1} + \cdots + a_j)}_{\text{Smallest solution ending at } a_{j+k-1}}$

Step 2: Recursively define solution

Let δ_i denote the minimal sum of a sublist with the restriction that it ends with a_i. Using the above, we can recursively define it

$$\delta_i = \begin{cases} a_0 & \text{if } i = 0 \\ \min(a_i, a_i + \delta_{i-1}) & \text{otherwise} \end{cases}$$

The idea is that when all δ_i's are computed, we end up with a list $D = [\delta_0, \dots, \delta_n]$ such that $D[i]$ contains the smallest sum of a sublist ending at a_i. Any smallest sublist has to end somewhere, so by picking $\min_i D[i]$, we have solved our problem.

Step 3: Write program

ALG(L)

$D = [0, \dots, 0]$ #New List of length n

```
D[0] = L[0]          #'Base case'
for i ∈ (1, ..., n) do
|     D[i] = min(L[i], L[i] + D[i − 1])    # This is δ_i from step 2!
end

return D
```

The minimal sum is computed by computing min(ALG(L)).

Step 4: Use the algorithm in step 3 to find the smallest sublist

Exericse

The other problems I mentioned (for next time?):

Problem 2

Given a sequence of integers $a_0, \ldots, a_n$, find the longest increasing subsequence $a_{i_1} < \cdots < a_{i_m}$

Problem 3

Given a set of boxes $B_i = (W_i, L_i, H_i)$ (where W_i = width, L_i = length, H_i = height), find the tallest possible stack of boxes s.t. if B_i is on top of B_j, then $W_i < W_j$ and $L_i < L_j$. Let us also disallow rotatating the boxes, to make things easier.

Problem 4

Same thing again, but with rotations allowed this time.