CS/ECE 473 Fall 2020

Homework 1 Problem 1

Ryan Prendergast (ryanp4)
Noah Watson (nwatson3)
Lawson (laws)

1.a

Describe an algorithm to find an optimal solution $k_1 ... k_n$ that runs in time polynomial in k and n.

Solution: We define the function minFuncSum(i,k) which finds the minimum function sum of $T_i ... T_n$ subject to the constraint that the non-negative integers $k_i ... k_n$ sums to k with the following recurrence:

$$\text{minFuncSum(i,k)} = \begin{cases} T_i(k) & \text{if } i = N \\ min_{j=0}^{k}\{T_i(j) + \text{minFuncSum(i+1, k-j)} & \text{otherwise} \end{cases}$$

We find the minimum function sum of the original problem by calling minFuncSum(1, k).

The above recurrence runs in exponential time. However we see that each iteration minFuncSum(i,k) depends on minFuncSum(i + 1, j) for all possible $j = 0$ to n. We memoize in a 2d, $N * k$ array A with the minimum function sum. We also memoize the value of $0 \leq k_i \leq k$ at which the minimum was found.

$A[i][j]$.sum holds the minimum of the functions $T_i ... T_n$ subject to the constraint that $k_i ... k_n$ sums to j; if $i = N$, then $A[n][j].\text{sum} = T_n(j)$. $A[i][j]$.k holds the value of $0 \leq k_i \leq k$ at which the minimum lies; if $i = N$, then $A[n][j].\text{k} = j$

Due to the dependency ordering, we can define an iterative algorithm to fill in this $N * k$ array. We iterate from right to left, from $i = n$ to $i = 1$, filling in the whole column of all possible values of j from $0 ... k$ at each iteration.

To get the sequence $k_1 ... k_n$, we start at $A[1][k]$ after filling in the whole array with the above algo. $A[i][k]$.k is k_1, then we go to $A[2][k - k_1]$. $A[2][k - k_1]$.k is k_2. In general, $k_i = A[i][k - \sum_{j=1}^{i-1} k_j]$.

See Algorithm 1 bellow.

Time Complexity As seen in Algorithm 1 bellow, filling in the array requires a nested loop over n and k, and each iteration requires $O(k)$ steps to find the minimum. So, the total runtime is $O(k^2 * N)$.
Space Complexity The space required is dominated by the array A, which is size $O(N * k)$. ■

Algorithm 1 Iterative Min Function Sum Series

for $i = n$ to $i = 1$ **do**
  for $j = 0$ to $j = k$ **do**
    if $i = n$ **then**
      $A[i][j].\text{sum} = F_i(j)$
      $A[i][j].\text{k} = j$
    else
      $A[i][j].\text{sum} = \min_{h=0}^{h=j}\{F_i(h) + A[i+1][j-h]$
      $A[i][j].\text{k} = \text{argmin}_{h=0}^{h=j}\{F_i(h) + A[i+1][j-h]$
    end if
  end for
end for
$j \leftarrow k$
for $i = 1$ to $i = n$ **do**
  $k_i \leftarrow A[i][j].k$
  $j \leftarrow j - k_i$
end for