Algorithm Design and Analysis
CSE222 Winter '22

Quiz 2
Time 37 mins.

Please write solutions independent of each other. This is a closed book test. You can not use books or lecture notes.

Problem 1 (10 points)

Mr Monkey is standing in front of a row of banana trees on Skull Island which actually belong to his rival King Kong. Let the trees be labelled as $t_1, t_2, \cdots t_n$. Let $v_i, \forall i = 1, 2, \cdots n$ be the number of bananas in tree t_i. Further, let $\ell_i, i = 1, 2, \cdots n-1$ denote the distance between the trees t_i, t_{i+1}.

The banana trees are unusually high on that island. He wants to steal as many bananas as possible. Here is his plan. He will climb up one of the trees and then keep jumping from one tree to the one with immediate higher index, while collecting all the bananas from each of them. No longer being the agile young monkey he once used to be, he can only jump a total distance of L, after which he will climb down and run away before Kong crushes his head.

Can you write an algorithm to help Mr. Monkey steal as many bananas as possible ? Your algorithm should have a runtime a polynomial in n, L. Please show all the steps of DP as in the sample solution - **subproblem definition, recurrences, pseudocode, runtime** .

(Disclaimer : The author of this problem does not endorse stealing. The story is a work of fiction with no intentional resemblance to any character living or dead)

Solution.

Subproblems. Let $OptBan(i, \ell)$ = the maximum number of bananas Mr. Monkey can collect **if his jumps end at tree** t_i and he has jumped a total length of **at most** ℓ, for all $i = 0, 1, 2, \cdots n$ and $\ell = 0, 1, 2 \cdots L$ where t_0 is a dummy tree which does not exist.

Recurrence. $\forall i = 0, 1, 2, \cdots n, \ell = 1, 2, \cdots L$

$$OptBan(i, \ell) = OptBan(i-1, \ell - \ell_{i-1}) + v_i\}, \text{if } \ell_{i-1} \leq \ell$$
$$OptBan(i, \ell) = v_i, \text{if } \ell_{i-1} > \ell$$

$$OptBan(0, \ell) = 0, \forall \ell = 0, 1, 2, \cdots L$$

$$OptBan(i, 0) = v_i, \forall i = 1, 2, \cdots n$$

Explanation : (Not required for grading) There are two possibilities for $OptBan(i, \ell)$. Since it has to include tree t_i as per the definition of the subproblem, this has to either be the last of the sequence of trees he has jumped or this is the only tree that he climbs and escapes after collecting all bananas from t_i. In case the length of last jump ℓ_{i-1}, that is t_{i-1} to t_i is bigger than ℓ, then surely this jump cannot be part of the sequence and hence the best solution is to just consider t_i. In case this length is less than ℓ, then it actually makes sense to consider the best sequence of jumps that ends at t_{i-1} and is of length at most $\ell - ell_i$ and take the final leap to t_i.

The second base case is interesting. Even if the monkey cannot jump at all ($\ell = 0$), it can still climb up one tree, collect bananas and escape.

Runtime. is clearly $\mathcal{O}(nL)$.

p.s.: This problem is a combination of maximum sub contiguous subarray and knapsack.

Algorithm 1 CollectBanana(Array $T = t_1, t_2, \cdots t_n$)

Let $OptBan$: 2-D Array of size $(n+1) \times (L+1)$
for $\ell = 0$ to L **do**
  $OptBan[0, \ell] = 0$
end for
for $i = 1$ to n **do**
  $OptBan[i, 0] = v_i$
end for
for $i = 1$ to n **do**
  for $\ell = 1$ to L **do**
    if $\ell_{i-1} > \ell$ **then**
      $OptBan[i, \ell] = v_i$
    else
      $OptBan[i, \ell] = OptBan[i, \ell - \ell_{i-1}] + v_i$
    end if
  end for
end for
Return $\max_{1 \le i \le n, 0 \le \ell \le L} OptBan[i, \ell]$

Common mistake. Many of you must have realized that this is an extension of knapsack and have defined the subproblems as $Opt(i, \ell)$ as the maximum number of bananas possible to collect by considering trees $t_1, t_2, \cdots t_i$ and jumping length ℓ. Then, in the case when tree t_i is visited by Mr. Monkey, you have considered the recursion as max of $Opt(i, \ell), Opt(i-1, \ell - \ell_{i-1}) + v_i$. This is wrong since this assumes that the optimal solution $Opt(i-1, \ell - \ell_{i-1})$ ends at t_{i-1}, which is not necessarily true. This would have been correct if Mr. Monkey could jump from any tree to any other tree. But in the problem, it is stated that he can jump to only the immediately next one. This is why it is crucial to define the subproblems as done in the solution (and also solution to Max sum subarray in Tutorial 1)

Naive Algorithm.

Algorithm 2 Naive Algorithm for CollectBanana(Array $T = t_1, t_2, \cdots t_n$)

if $L = 0$ **then**
  Return $\max\{v_i \mid i = 1, \ldots, n\}$
end if
$b = 0$
for $i = 1$ to n **do**
  for $j = i + 1$ to $n - 1$ **do**
    $dist_1 = \sum_{k=i}^{j-1} \ell_k$
    $dist_2 = dist_1 + \ell_j$
    if $dist_1 \le L$ and $dist_2 > L$ **then**
      $b = \max\left\{b, \sum_{k=i}^{j} v_j\right\}$
    end if
  end for
end for
Return b

Rubric: If the answer is a naive algorithm running in $O(n^2)$-time, then the following rubric has to be used.

- (correct pseudocode/description) 10 Marks.
- (Slightly incorrect pseudocode/description) 7 Marks if some cases are missing/incorrect in the naive algorithm.
- (partially correct pseudocode/description) 4 Marks Some part of the algorithm is correct, but some parts are incorrect.
- (incorrect) 0 Marks.

In case the answer is a dynamic programming based algorithm, the following rubric idea has to be used.

- **(correct subproblem)** 2 marks for defining subproblems correctly. (slightly incorrect subproblem) 1 marks for defining subproblems that are correct but cannot be used to get optimal solution. (partially correct subproblem) 0.5 mark for inconsistent definition of subproblems. (incorrect subproblem) 0 mark.
- **(Final answer)** 1 mark for writing correctly, 0 for not writing.
- **(correct recurrence relation)** 2 marks for correct recurrence relation including all the base cases. (slightly incorrect recurrence) 1 marks if the recurrence relation is correct but some base case is missing. (partially correct recurrence) 0.5 mark if there is inconsistency in defining recurrence relation. (incorrect recurrence) 0 mark
- **(correct pseudocode)** 3 Marks for writing correct pseudocode. (partially correct pseudocode) 2 Marks if missing table declaration with dimensions 2 Marks if wrong loop. 1 Mark if both the above issues exist (incorrect pseudocode) 0
- 2 marks for correct run-time.