HKOI 2015/16 TFT Q1 Solution

Prepared by: Alex Tung

Comments

- Easy-Medium
- Greedy algorithm; less standard

- More full scorers (5) than expected!

- Lack of partial solution for K <= 10 is puzzling

Observation 1

- For each type X of apple, we only need to consider
  - the first tall tree with type X apples and
  - the first short tree with type X apples

- Plus, we take exactly one apple of each type (thus, in all cases where winning is possible, number of trees climbed = K)

Observation 2

- N is irrelevant
- For each tree, we only need to know the number of the first checkpoint after it

Notations

For 1 <= i <= K,

- Let T[i] be the first checkpoint after the first tall tree with type i apples (-1 if not exist)
- Let S[i] be the first checkpoint after the first short tree with type j apples (-1 if not exist)

Example

- (N = 5, K = 2)
- T 1, T 2, T 1, S 1, T 1
- 2 5

- T[1] = 1; S[1] = 2
- T[2] = 1; S[2] = -1

- Note that the notation used in this presentation is only convenient for demonstrating the idea; it complicates the process of retrieving the answer

Solution 1

- Subtask 1 (all trees are short)

- T[X] = -1 for all X

```
for i from 1 to K
  for j from 1 to K
    if (type j apple has not been taken yet) and (S[X] <= i)
      take type j apple for the i-th checkpoint
      break
```

Solution 1

- Time complexity: $O(K^2)$
- Expected score: 8

Solution 2

- Subtask 2 (N = K+1)
- We need at least K apples; that means we can only give up climbing one tree!

- Exhaustion on which tree to not climb
- For each case, simulate the race

Solution 2

- Time complexity: $O(K^2)$
- Expected score: 15
- Expected score (combined with solution 1): 23

Solution 3

- Solves for K <= 10
- Thanks to observation 1, we can exhaust, for each type of apple, whether to climb **the** tall tree or **the** short tree

- Then, check whether RB can pass all checkpoints in O(K) / $O(K^2)$

Solution 3

- Time complexity: $O(2^K * K)$ / $O(2^K * K^2)$
- Expected score: 27
- Expected score (combined with solutions 1+2): 50

Solution 4 (greedy)

- We consider the checkpoints one by one
- For each checkpoint, we want to choose the “best” type of apple among those available for choice

Solution 4

- Climbing a short tree is always more desirable than climbing a tall tree
- If there are multiple available short trees, any one will do

- If no short trees are available, climb the tall tree T[X] such that S[X] is maximal
- Intuitively, this is because it takes the longest time to ‘wait’ for S[X] to be available

Solution 4

```
for i from 1 to K
  //try short tree
  for j from 1 to K
    if (!used[j]) and (S[j] <= i)
      climb(S[j]) //perform necessary updates
```

Solution 4

```
//try tall tree; only if no short trees are available
  maxi = -1
  choose = -1
  for j from 1 to K
    if(!used[j]) and (T[j] <= i) and S[j] > maxi
      choose = j
      maxi = S[j]
  climb(T[choose])
//if still no trees, ‘Lose’
```

Solution 4

- Time complexity: $O(N + K^2)$
- Expected score: 80

- P.S. this task was originally proposed for HKOI 2015/16, without the last subtask (i.e. solution 4 would suffice to get 100 points)

Solution 5

- We speed up solution 4

Moving from one checkpoint to the next,

- we add the newly available tall trees, T[i], into a heap (e.g. C++ priority_queue), using S[i] for ordering
- we add the newly available short trees, S[i], into a queue (many other data structures will also do)

Solution 5

- Then, do the same as in solution 4
- Short tree is better than tall tree
- Tall tree with larger S[X] is better than tall tree with smaller S[X]

Solution 5

- Time complexity: O(N + K log K)
- Expected score: 100 :)