Q1

LI Yuntong 20944800

① Find the largest number: From the first element in the array to the last element, using binary search. group them into pairs ($\frac{n}{2}$ pairs), compare them (two elements in each pair), select the large one, and continue comparing the larger one in pairs ($\frac{n}{2^2}$ pairs), and so on, until find the largest number. The total height = $\log_2 n$ from bottom to the top. Total comparasion in here to find the largest one is $\sum_{i=1}^{\log_2 n} \frac{n}{2^i} = n-1$

② Find the second large: During the comparison, all the candidate second largest number must have compared the largest number in the process of finding the largest number. During the path of the largest number from the button to the top. In each height level, it will compare with one number. (except the top). So, the number of candidate's second number is $\log_2 n$, same as the height of the binary tree. For this $\log_2 n$ number, do "find the largest number" again. so there are $(\log_2 n - 1)$ comparisons, then we can get the second largest number.

So, in total, the number of comparisons is $(n-1)+(\log_2 n - 1) = n + \log_2 n - 2$

Q2.

$Pr(\text{RANDOM-SAMPLE}(m,n) = \{a_1, a_2, \dots, a_m\}) = \frac{1}{C_n^m} = \frac{m! \cdot (n-m)!}{n!}$

① When $m=1$. for any $a \in [1,n]$

$Pr(\text{RANDOM\_SAMPLE}(1,n) = \{a\}) = \frac{1}{n} = \frac{(n-1)!}{n!}$

② when $1 < m \le n$ suppose $Pr(\text{RANDOM\_SAMPLE}(m-1, n-1) = \{a_1, \dots, a_{m-1}\}) = \frac{(m-1)! \cdot (n-m)!}{(n-1)!}$ is correct.

③ Then. according to the question. for any possible subset $\{a_1, \dots, a_m\}$ of $[1,n]$:

<1> if $n \in \{a_1, \dots, a_m\}$, suppose $a_m = n$.

$Pr(\text{RANDOM\_SAMPLE}(m,n) = \{a_1, \dots, a_m\})$

$= Pr[\text{RANDOM-SAMPLE}(m-1, n-1) = \{a_1, \dots, a_{m-1}\}] \cdot Pr[\text{RANDOM}(1,n) \in \{a_1, \dots, a_{m-1}\}] +$

$Pr[\text{RANDOM-SAMPLE}(m-1, n-1) = \{a_1, \dots, a_{m-1}\}] \cdot Pr[\text{RANDOM}(1,n) = n]$

$= \frac{(m-1)!\,(n-m)!}{(n-1)!} \cdot \left(\frac{m-1}{n} + \frac{1}{n}\right) = \frac{m!\,(n-m)!}{n!}$

<2> if $n \notin \{a_1, \dots, a_m\}$. then.

$Pr(\text{RANDOM\_SAMPLE}(m,n) = \{a_1, \dots, a_m\})$

$= m \cdot Pr[\text{RANDOM\_SAMPLE}(m-1, n-1) = \{a_1, \dots, a_{j-1}, a_{j+1}, \dots, a_m\}] \cdot Pr[\text{RANDOM}(1,n) = a_j]$

$= m \cdot \frac{(m-1)!\,(n-m)!}{(n-1)!} \cdot \frac{1}{n} = \frac{m! \cdot (n-m)!}{n!}$

So. above all, $Pr(\text{RANDOM\_SAMPLE}(m,n) = \{a_1, \dots, a_m\}) = \frac{m! \cdot (n-m)!}{n!}$.

RANDOM_SAMPLE returns a subset of $[1,n]$ of size m drawn uniformly at random.

Q3

```
union (rectangles)
    if length of rectangles is 1 then x_left, x_right, height ← rectangles[0]
        return [(x_left, height), (x_right, 0)]
    mid ← ⌈length of rectangles/2⌉
    L_half ← rectangles[:mid]
    R_half ← rectangles[mid+1:]
    L_p ← union(L_half)
    R_p ← union(R_half)
  return merge(L_p, R_p)


merge(L_p, R_p)
    i ← 0
    j ← 0
    merge_points[]
    current_heights ← [0,0]
    while i < length of L_p and j < length of R_p do
            x_left, h_left ← L_p[i]
            x_right, h_right ← R_p[j]
            x_min ← min of (x_left, x_right)
            if x_left ≤ x_right then
                    current_height[0] ← h_left
                    i ← i+1
            else
                    current_height[1] ← h_right
                    j ← j+1
            max_height ← max of (current_heights[0], current_heights[1])
            if not merged_points or max_height != merged_points[-1][1] then
```

append (x_min , max_height) to the last of merged_points.

extend the remaining points from the left set L_p[i:] to the merged_points

extend the remaining points from the right set R_p[j:] to the merged_points

return merged_points

Q4.

(a) $E[X_i] = Pr(X_i = 1) = \frac{1}{n}$

(b) If the i-th smallest number is chosen as the pivot. Then we have two sub-arrays, one is all smaller than this pivot. [(i-1) numbers]. one is larger than this pivot [(n-i) numbers]

The partition cost $\Theta(n)$. Then it continue do the quick sort on the two array

So: we have $E[T(n)] = E[\sum_{i=1}^{n} X_i \cdot (T(i-1) + T(n-i) + \Theta(n))]$

(c) $E[T(n)] = E[\sum_{i=1}^{n} X_i \cdot (T(i-1) + T(n-i) + \Theta(n))]$

$= \sum_{i=1}^{n} Pr(X_i = 1) \cdot E[T(i-1) + T(n-i) + \Theta(n)]$

$= \frac{1}{n} \cdot \sum_{i=1}^{n} E[T(i-1) + T(n-i) + \Theta(n)]$

$= \frac{1}{n} \cdot \sum_{i=1}^{n} [E[T(i-1)] + E[T(n-i)]] + \Theta(n)$

$= \frac{1}{n} \cdot \sum_{i=1}^{n} [2E[T(i-1)]] + \Theta(n)$

$= \frac{2}{n} \cdot \sum_{i=1}^{n} E[T(i-1)] + \Theta(n)$

$= \frac{2}{n} \cdot \sum_{i=0}^{n-1} E[T(i)] + \Theta(n)$

$= \frac{2}{n} \cdot \sum_{i=2}^{n-1} E[T(i)] + \Theta(n)$

(d) Suppose n is even:

$$\sum_{k=2}^{n-1} k\log k = \sum_{k=2}^{\frac{n}{2}-1} k\log k + \sum_{k=\frac{n}{2}}^{n-1} k\log k \le \sum_{k=2}^{\frac{n}{2}-1} k\log\frac{n}{2} + \sum_{k=\frac{n}{2}}^{n-1} k\log n$$

$$= (\log\tfrac{n}{2})\cdot\frac{(\frac{n}{2}+1)(\frac{n}{2}-2)}{2} + (\log n)\cdot\frac{(\frac{3n}{2}-1)\cdot\frac{n}{2}}{2}$$

$$= (\log\tfrac{n}{2})\cdot(\tfrac{n^2}{8}-\tfrac{n}{4}-1) + (\log n)\cdot(\tfrac{3n^2}{8}-\tfrac{n}{4})$$

$$\le \tfrac{n^2}{8}\log\tfrac{n}{2} + \tfrac{3n^2}{8}\log n = \tfrac{n^2}{8}(\log n - \log 2) + \tfrac{3n^2}{8}\log n$$

$$= \tfrac{1}{2}n^2\log n - \tfrac{n^2}{8}\log 2 = \tfrac{n^2}{2}\log n - \tfrac{n^2}{8}$$

Suppose n is odd. same as above.

(e) ① $E[T(n)] \le cn\log n$ for some positive constant c by induction.

when $n=2$. it is true.

Suppose: $E[T(k)] \le ck\log k$. $(k=2,\dots,n-1)$

$$E[T(n)] = \frac{2}{n}\sum_{k=2}^{n-1} E[T(k)] + \Theta(n) \le \frac{2}{n}\sum_{k=2}^{n-1} ck\log k + \Theta(n)$$

$$= \frac{2c}{n}\sum_{k=2}^{n-1} k\log k + \Theta(n) \le \frac{2c}{n}(\tfrac{1}{2}n^2\log n - \tfrac{1}{8}n^2) + \Theta(n)$$

$$= cn\log n - \frac{cn}{4} + \Theta(n) \quad \text{when } n \text{ is suffiency large}$$

$$\le cn\log n$$

② $E[T(n)] \ge mn\log n$ for a positive constant m

when $n=2$. is true

Suppose $E[T(k)] \ge mk\log k$ $(k=2,\dots,n-1)$

$$E[T(n)] = \frac{2}{n}\sum_{k=2}^{n-1} E[T(k)] + \Theta(n) \ge \frac{2}{n}\cdot\sum_{k=2}^{n-1} mk\log k + \Theta(n)$$

$$> \frac{2}{n}\cdot\sum_{k=\lceil\frac{n}{2}\rceil}^{n-1} mk\log k + \Theta(n) \ge \frac{2m}{n}\cdot\frac{(\lceil\frac{n}{2}\rceil+n-1)\cdot\lceil\frac{n}{2}\rceil}{2}\log\lceil\tfrac{n}{2}\rceil + \Theta(n)$$

$$\ge \frac{2m}{n}\cdot(\tfrac{3n^2}{8}-\tfrac{n}{4})\cdot\log\tfrac{n}{2} + \Theta(n) \ge m\cdot(\tfrac{3n}{4}-\tfrac{1}{2})\log\tfrac{n}{2} + \Theta(n).$$

when n is suffiency large. $E[T(n)] \ge mn\log n$.

So. $E[T(n)] = \Theta(n\log n)$

Q5

Compute $k_1, k_2, k_3 \dots k_{q-1}$.

For each i from 1 to $q-1$.

Use black-box algorithm to find the median. $O(n)$

divide the array into two part: all the elements less than the median to L.

all the elements larger than the median to R.

① if k_i corresponds to an index in the lower half. L. find in the lower half

② if k_i corresponds to an index in the upper half. R. find in the upper half.

keep continue until find all $q-1$

For the worst case, we need $(\log q)$ times to do the recursion. and each time we use black-box find the median need $O(n)$ times. and each operation is linear time.

So, the total is $O(n \log q)$