cpt\_s 350

Homework 3

11641327 Yu-Chieh Wang

2020/2/20

## 1. Given: An array A of n elements.

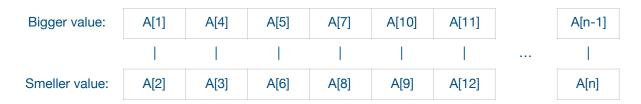
Question: Find an algorithm can select both maximal and minimal elements by running at most 1.5 n comparisons.

| A[1] | A[2] | A[3] | A[4] | A[5] | A[6] | A[7] | A[8] | A[9] | A[10] | A[11] | A[12] | <br>A[n] |  |
|------|------|------|------|------|------|------|------|------|-------|-------|-------|----------|--|
|      |      |      |      |      |      |      |      |      |       |       |       |          |  |

## Solution 1:

Use quick sort on the array A. Then, we can pick the maximal and minimal elements directly.

## Solution 2:



First, We make every two elements a smell group, and do one comparison in each group. we will have 0.5n groups, so it's going to run 0.5n comparisons. Then, we make two large groups: One to combine the bigger value in each smell group. The other one combine the smeller value. Now, each large group has 0.5n elements. Finally, when we do select number, we only compare to a large group (0.5n) comparisons. As a result, we spend 0.5n comparisons on separating an array to two groups, 0.5n comparisons on selecting the largest element, and 0.5n comparisons on

## selecting the minimal element, which is totally 1.5n comparisons.

- 2. Compare the average-case complexities of two algorithm.
- 3. The worst-case complexities for k=3 and k=7.
- 4. Given: an algorithm called ilselect(A, n, i).

Question: Find the worst-case and the average-case complexities.

We learn the time complexities of following algorithms, so we can do it directly.

|              | Best-case | Worst-case | Average-case |
|--------------|-----------|------------|--------------|
| Quickselect  | O(1)      | $O(n^2)$   | O(n)         |
| Linearselect | O(1)      | $O(n^2)$   | O(n)         |

First of all, we separate the process of ilselect to three steps:

- (1) Do partition.
- (2) Run quickselect on the low part.
- (3) Run linearselect on the hight part.

| low part | r | high part |
|----------|---|-----------|
| (r-1)    |   | (n-r)     |

Then, we count the complexities by these steps:

- (a) The worst-case complexities:
  - (1)  $T_{WORST}(n) = O(n)$

(2) 
$$T_{WORST}(n) = \min_{1 \le r \le n} \{ T_{WORST}(r-1) \} = O(r^2)$$

(3) 
$$T_{WORST}(n) = \min_{1 \le r \le n} \{ T_{WORST}(n-r) \} = O((n-r)^2)$$

$$T_{WORST}(n) = O(n) + O(r^2) + O((n-r)^2) = O(n^2)$$

- (b) The average-case complexities:
  - (1)  $T_{AVG}(n) = O(n)$

(2) 
$$T_{AVG}(n) = \frac{1}{n} \sum_{r=1}^{n} \{ T_{AVG}(r-1) \} = O(r)$$

(3) 
$$T_{AVG}(n) = \frac{1}{n} \sum_{r=1}^{n} \{ T_{AVG}(n-r) \} = O(n-r)$$

$$T_{AVG}(n) = O(n) + O(r) + O(n-r)$$
 =  $O(n)$ 

5. 5com