Quick Sort

Yufei Tao

Department of Computer Science and Engineering Chinese University of Hong Kong



Quick Sort

э

1/12

イロト イボト イヨト イヨト

Today, we will discuss another sorting algorithm named **quick sort**. It is a randomized algorithm that runs in $O(n^2)$ time in the worst case but $O(n \log n)$ time in expectation.

2/12

< □ > < 同 > < 三</p>

Recall:

The Sorting Problem

Problem Input:

A set S of n integers is given in an array A of length n.

Goal:

Produce an array that stores the elements of S in ascending order.

(日)

э



- Pick an integer p in A uniformly at random, which is called the pivot.
- 2 Re-arrange the integers in an array A' such that
 - all the integers smaller than p are before p in A';
 - all the integers larger than p are after p in A'.
- Sort the part of A' before p recursively (a subproblem).
- Sort the part of A' after p recursively (a subproblem).

Example

After Step 1 (suppose that 26 was randomly picked as the pivot):



After Step 2:

p		
17 12 5 9 26 38 28 88	41 72 83 69 47 68 52 35	

After Steps 3 and 4:

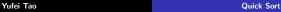


Yufei Tao

Quick sort is not attractive in the worst case: its worst case time is $O(n^2)$ (why?). However, quick sort is fast in expectation: we will prove that its expected time is $O(n \log n)$. Remember: this holds on every input array A.

(4) E (4) E (4)

The rest of the slides will not be tested for CSCI2100.



◆□ ▶ ◆□ ▶ ◆三 ▶ ◆三 ▶ ● □ ● ● ●

First, convince yourself that it suffices to analyze the number X of comparisons. The running time is bounded by O(n + X).

Next, we will prove that $\boldsymbol{E}[X] = O(n \log n)$.



・ロト ・四ト ・ヨト ・ヨト

э

Denote by e_i the *i*-th smallest integer in *S*. Consider e_i, e_j for any i, j such that $i \neq j$.

What is the probability that quick sort compares e_i and e_j ?

This question, which seems to be difficult at first glance, has a surprisingly simple answer. Let us observe:

- Every element will be selected as a pivot exactly once.
- *e_i* and *e_j* are **not** compared, if any element **between** them gets selected as a pivot **before** *e_i* and *e_j*.

For example, suppose that i = 7 and j = 12. If e_9 is the pivot, then e_i and e_j will be separated by e_9 (think: why?) and will not be compared in the rest of the algorithm.

Yufei Tao	Quick Sort

イロト 不得 トイヨト イヨト 三日

Therefore, e_i and e_j are compared if and only if either one is the first among e_i, e_{i+1}, \dots, e_j picked as a pivot.

The probability is 2/(j - i + 1).



< ロ > < 同 > < 回 > < 回 >

-

Define random variable X_{ij} to be 1, if e_i and e_j are compared. Otherwise, $X_{ij} = 0$. We thus have $Pr[X_{ij} = 1] = 2/(j - i + 1)$. That is, $E[X_{ii}] = 2/(i - i + 1).$

Clearly, $X = \sum_{i,i} X_{ij}$. Hence:

Yufei Tao

$$E[X] = \sum_{i,j:i < j} E[X_{ij}] = \sum_{i,j:i < j} \frac{2}{j - i + 1}$$
$$= 2\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{j - i + 1}$$
$$= 2\sum_{i=1}^{n-1} O(\log(n - i + 1))$$
$$= 2\sum_{i=1}^{n-1} O(\log n) = O(n \log n).$$

・ロッ ・雪 ・ ・ ヨ ・ ・ **Quick Sort**

-

The above analysis used the following fact:

$$1 + 1/2 + 1/3 + 1/4 + \dots + 1/n = O(\log n).$$

The left-hand side is called the harmonic series.



メロト メポト メヨト メヨ

э