CS 401: Advanced Algorithms 03/12/2021

Lecture #18: Analyzing runtime of Randomized Quick Sort

Lecturer: Nathaniel Kell *Scribe: Khoi Le Viet*

1 Overview

In this lecture, we will analyze the run time of the Randomized Quick Sort. In the next sections, we will be defining the crucial components of the analysis and analyzing the overall runtime.

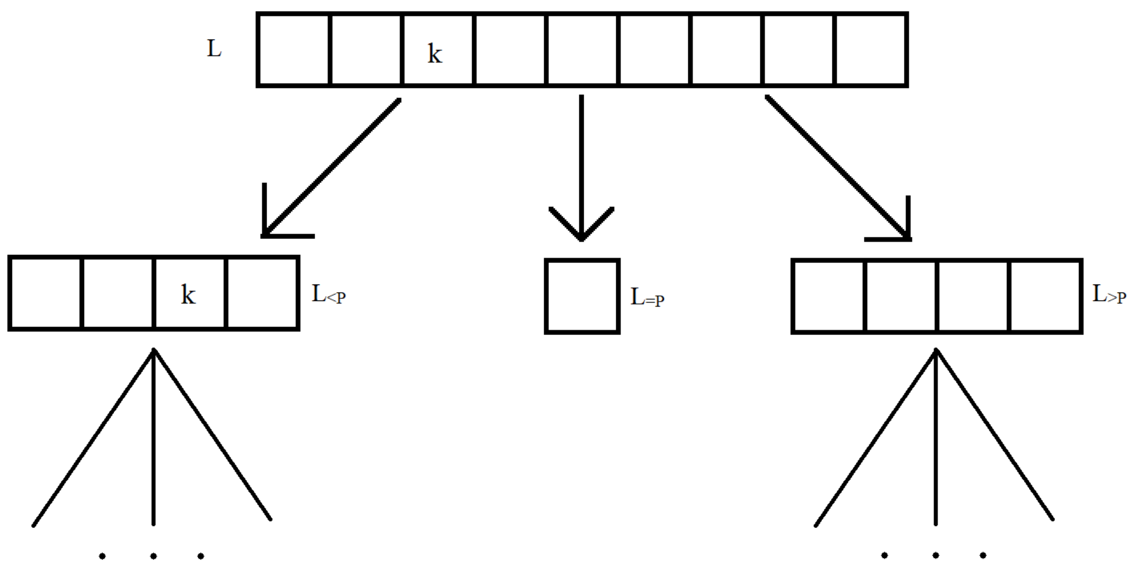
2 Problem Definition:

Before we proceed further to the analysis, we will firstly define the pseudocode for the Randomized Quick Sort:

Algorithm 1 Randomized Quick Sort (L):

1: **if** $|L| \leq 1$: **then**
2:     return L
3: **end if**
4: pivot $\leftarrow$ rand$(1, |L|)$
5: $P \leftarrow L$[pivot]
6: $L_{<P}$, L_P, $L_{>P} \leftarrow$ partition (L, P)
7: $L_{<P} \leftarrow$ Randomized Quick Sort $(L_{<P})$
8: $L_{>P} \leftarrow$ Randomized Quick Sort $(L_{>P})$
9: return $L_{<P} + L_P + L_{>P}$

The line 6 of the pseudocode partitions into three parts as $L_{<P}$ represents the list containing every element in L that is less than P, L_P represents the list containing only P, and $L_{>P}$ represents the list containing every element in L that is bigger than P. Line 7 and 8 performs the algorithm recursively on the two lists $L_{<P}$ and $L_{>P}$, and at the end, the algorithm returns the sorted list.



The above picture illustrates a tree containing the first few steps of the algorithm, where a list L is divided into three sub-lists, and the process continues until the length of every sub-list is one, i.e. the list L is sorted. In the tree above, for an element k in list L, we define P_k to be the path that the element k takes from the root, i.e. list L, to its leaf where a sub-list only contains k. In addition, we also define $|P_k|$ to be the number of edges in P_k.

We also define different types of edges in the tree. Specifically, let $|A|$ be the length of a list A, then for an element k,

- An edge is black if $k \in L_P$
- An edge is called blue if $k \in L'$ and $|L'| \leq \frac{1}{2}|L|$
- An edge is called red if $k \in L'$ and $|L'| > \frac{1}{2}|L|$

For the above defined colored edge, we will define $|P_k(\text{black})|$, $|P_k(\text{blue})|$ and $|P_k(\text{red})|$ as the total number of black edges, blue edges and red edges in P_k, respectively.

3 Analysis of the runtime of the Randomized Quick Sort:

Lemma 1. *For every element k in a list, $|P_k| = O(\log n)$ in expectation.*

Proof. Let n be the length of a list. Based on the definition of colors of edges we defined above, we have that for any path P_k, the total number of edges in P_k is equal to the sum of all black edges, blue edges and red edges in that path. In other words,

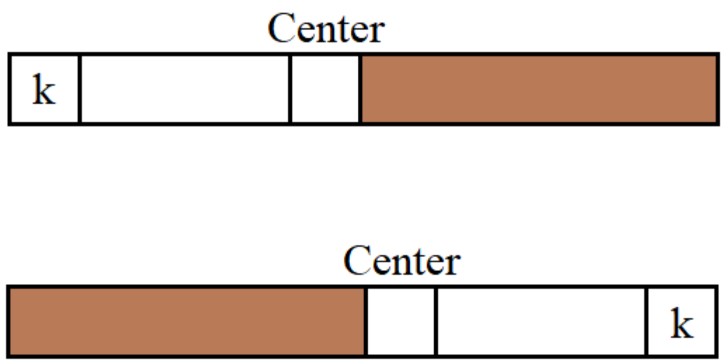
$$|P_k| = |P_k(\text{black})| + |P_k(\text{blue})| + |P_k(\text{red})|$$

Now, we only have to consider every type of edges in that path:

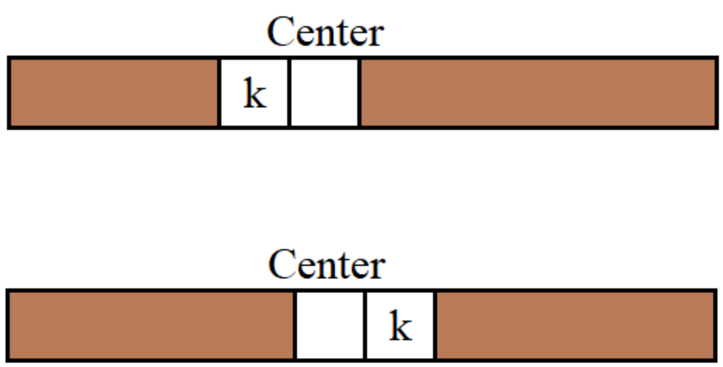
- Consider the number of black edges in P_k, it is exactly one because only the last edge connected to the list containing only k can be black.

- Since the length of the list connected to a blue edge is at most a half of the original list, then there are at most $\log n$ number of blue edges.
- For the number of red edges, based on the definition, it is hard to define the exact number.

Considering the red edges, we will be using probability to find the number of red edges in a path P_k. To calculate the probability of having a red edge, we will consider the following scenario on a sorted list:



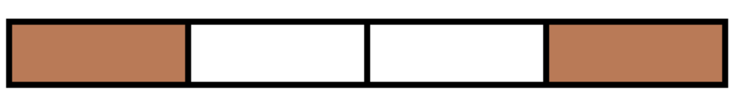
If we let k to be the smallest element or largest element in the list, then we can choose the pivot to be any element in the shaded region to get a red edge on k. Therefore, the probability is roughly a half. However, if we move k nearer to the center:



Then, for this case, the probability to have a red edge on k is roughly one because we can choose the pivot to be any element in the shaded region and k will be in the bigger partition. Therefore, the probability of having a red edge is not constant, and we have to find a different way to make it consistent. Therefore, we will redefine the blue edges and red edges as:

- An edge is called blue if $k \in L'$ and $|L'| \leq \frac{3}{4}|L|$
- An edge is called red if $k \in L'$ and $|L'| > \frac{3}{4}|L|$

Based on this definition, using the same reasoning, we can show that the number of blue edges in P_k is at most $\log_{\frac{4}{3}} n$, and the number or black edge is still one. Considering red edge:



We will divide the list into 3 quantiles, and for every element k in that list:

- If we choose the pivot to be in the shaded region, there will be one blue edge and one red edge.
- If we choose the pivot not to be in the shaded region, there will be two blue edge.

Therefore, regardless of the element in the list, the probability that a red edge appear will be at most $\frac{1}{2}$.

Now, we can bound the number of red edges based on the number of blue edges. Let X_i be a random variable equal to the number of red edges between the i^{th} blue edge and the $i+1^{th}$ blue edge. The variable we defined is a geometric random variable. Therefore, we will have

$$\mathbb{E}[X_i] \leq \frac{1}{\frac{1}{2}} = 2.$$

Therefore, in a path P_k, the number of red edges will be at most $2\log_{\frac{4}{3}} n$, and thus,

$$|P_k| \leq 1 + \log_{\frac{4}{3}} n + 2\log_{\frac{4}{3}} n = O(\log n)$$

$\square$

Based on the tree we shown above, the runtime of the Randomized Quick Sort can be also expressed as $O(\sum_{k \in L} |P_k|)$, and we know that $|P_k| = O(\log n)$ in expectation for every element k. Therefore, the runtime of the algorithm is $O(n \log n)$.

4 Summary

Through the lecture, we have successfully analyze the runtime of the Randomized Quick Sort algorithm, and it is $O(n \log n)$.