[NOTE for Algorithms course of ECE@UTH.GR]

Analysis of Quicksort

Dimitrios Katsaros

Electrical & Computer Engineering department, University of Thessaly, Volos, Greece

Abstract

This short note provides an average case analysis of Quicksort with a single *pivot* when all input configurations (permutations) are equiprobable.

Introduction

Quicksort [1] is a simple – yet, very efficient – sorting algorithm. A call to the Quicksort function begins by choosing a *pivot* element from the list of items to be sorted. Let us assume the pivot is x. The algorithm proceeds by comparing every other element to x, dividing the list of elements into two sublists: those that are less than x and those that are greater than x. Quicksort then recursively sorts these sublists.

In the worst case, Quicksort requires $\Omega(n^2)$ comparison operations. For instance, suppose our input has the form $x_1 = n, x_2 = n-1, \ldots, x_{n-1} = 2, x_n = 1$. Suppose also that we adopt the rule that the pivot should be the first element of the list. The first pivot chosen is then n, so Quicksort performs $n-1$ comparisons. The division has yielded one sublist of size 0 (which requires no work to be sorted recursively), and another of size $n-1$. The next pivot chosen is $n-1$, so Quicksort performs $n-2$ comparisons, and so on. Continuing in this fasion, Quicksort performs:

$$(n-1) + (n-2) + \cdots + 2 + 1 = \frac{n(n-1)}{2} \quad \text{comparisons.} \tag{1}$$

This bad result comes because we clearly made a bad choice of pivots for the given input. If our pivot always splits the list into two sublists of size at most $\lceil n/2 \rceil$, then the number of comparisons $T(n)$ would obey the followin recursion:

$$T(n) = 2T(\lceil n/2 \rceil) + \Theta(n), \tag{2}$$

which yields the asymptotic solution of $T(n) = O(n \times logn)$, which is the best possible bound for any comparison-based sorting algorithm.

Average case analysis of Quicksort

We assume that all inputs are distinct, and for simplicity, that we are sorting the numbers 1 to n. We assume that there are $n!$ and equally likely input configurations.

Let us assume the x is the chosen pivot, and denote as q the rank of $x = A[r]$ returned by Quicksort's routine PARTITION, i.e., the number of elements less than x. We deduce that the probability that $rank(x) = q$ equals to $1/n$ for each $1 \leq q \leq n$.

Let $T(n)$ denote the average running time of Quicksort. We can calculate the expected running time $T(n)$ as the weighted combination of n conditional expected running times, where each conditional expectation is conditioned on a different rank q for the partioned element, i.e.,:

$$T(n) = \Theta(n) + \sum_{rank(x)=q} \Bigg(Prob[rank(x) = q] \times \Big(\text{expected runtime given that } rank(x) = q\Big)\Bigg) \tag{3}$$

$$= \Theta(n) + \frac{1}{n} \times \sum_{rank(x)=q} \Big(\text{expected runtime given that } rank(x) = q\Big) \tag{4}$$

$$= \Theta(n) + \frac{1}{n} \times \sum_{q=1}^{n} \Bigg(T(q-1) + T(n-q) \Bigg) \tag{5}$$

$$= \Theta(n) + \frac{2}{n} \times \sum_{q=1}^{n} T(q). \tag{6}$$

We slightly modify this last equation to get:

$$T(n) = n + \frac{2}{n} \times \sum_{q=0}^{n-1} T(q). \tag{7}$$

We multiply both ends by n:

$$n \times T(n) = n^2 + 2 \times \sum_{q=0}^{n-1} T(q). \tag{8}$$

Thus the recurrence involving $T(n-1)$ is the following:

$$(n-1) \times T(n-1) = (n-1)^2 + 2 \times \sum_{q=0}^{n-2} T(q). \tag{9}$$

We subtract the two equations (9) and (8) one from another, and we get:

$$n \times T(n) - (n-1) \times T(n-1) = n^2 - (n-1)^2 + 2T(n-1) \implies \tag{10}$$

$$n \times T(n) - (n+1) \times T(n-1) = 2n - 1 \implies \tag{11}$$

$$T(n) = \frac{n+1}{n} \times T(n-1) + 2 - \frac{1}{n} \implies \tag{12}$$

$$\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{2}{n} + \Theta\Big(\frac{1}{n^2}\Big). \tag{13}$$

By setting $Z(n) = \frac{T(n)}{n+1}$, we get:

$$Z(n) = Z(n-1) + \frac{2}{n} + \Theta\Big(\frac{1}{n^2}\Big). \tag{14}$$

Using the fact that $Z(1) = T(1)/2 = 1/2$, we get:

$$Z(n) = \sum_{1 \le i \le n} \frac{2}{i} + \Theta(1) \tag{15}$$

$$= 2H_n + \Theta(1) \tag{16}$$

$$= 2ln(n) + \Theta(1). \tag{17}$$

Therefore,

$$T(n) = (n+1) \times Z(n) = 2(n+1) \times ln(n) + \Theta(n) = 2n \times ln(n) + \Theta(n) \tag{18}$$

Summary

In this short note we developed an average case analysis of the single-pivot Quicksort. Even though it is believed that the research on Quicksort-like sorting algorithms is not very active, several significant efforts deny this belief. For instance, dual pivot Quicksort [5], multi-pivot[1] Quicksort [4] comprise singificant works keeping this area of investigation still hot. Modern research on sorting is affected even by deep learning [2, 3].

References

[1] C. A. R. Hoare. Quicksort. *The Computer Journal*, 5(1):10–16, 1962.

[2] A. Kristo, K. E. Vaidya, U. Cetintemel, S. Misra, and T. Kraska. The case for a learned sorting algorithm. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 1001–1016, 2020.

[3] K. E. Vaidya. The case for a learned sorting algorithm. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2021.

[4] S. Wild. *Dual-pivot quicksort and beyond: Analysis of multiway partitioning and its practical potential*. PhD thesis, Department of Computer Science, Technische Universitat Kaiserslautern, 2016.

[5] V. Yaroslavskiy. Dual-pivot quicksort algorithm, 2009.

Approximation of the n-th harmonic number

The summation

$$\sum_{i=1}^{n} \frac{1}{i} \tag{19}$$

is known as the *harmonic number* H_n. We will show that

$$H_n = \sum_{i=1}^{n} \frac{1}{i} = ln(n) + \Theta(1) \tag{20}$$

Since $1/x$ is monotonically decreasing, we can write:

$$ln(n) = \int_{x=1}^{n} \frac{1}{x} dx \leq \sum_{i=1}^{n} \frac{1}{i} \tag{21}$$

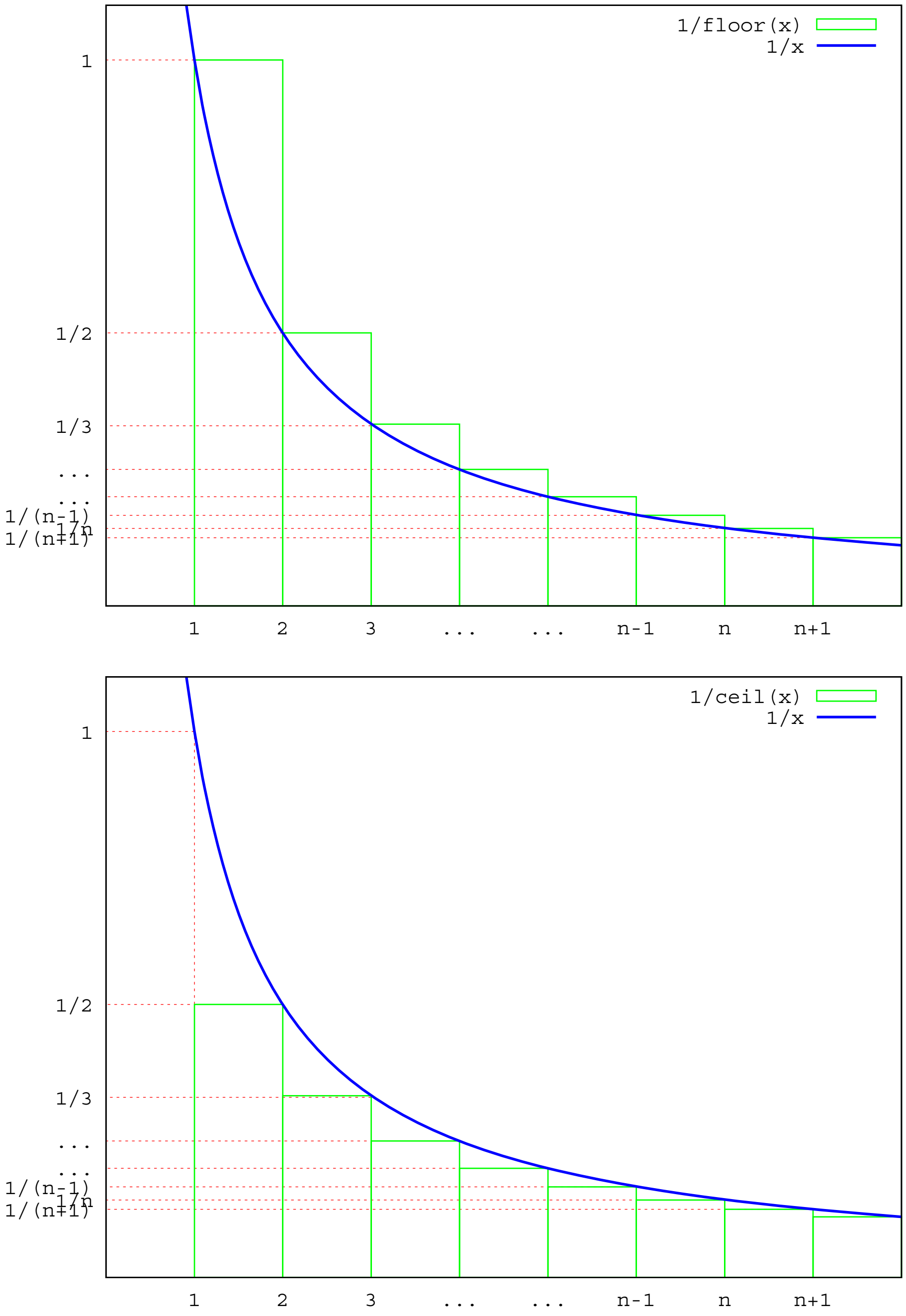
and it holds that

$$\sum_{i=2}^{n} \frac{1}{i} \leq \int_{x=1}^{n} \frac{1}{x} dx = ln(n). \tag{22}$$

The inequality in (22) is clarified in next page's figure. Therefore,

$$ln(n) \leq H_n \leq ln(n) + 1. \tag{23}$$

[1] Analyzed in an amazing PhD thesis.



Approximating the function $1/x$ (blue curve) from above (green boxes in top plot are equal to $\sum_{i=1}^{n} 1/i$) and below (green boxes in bottom plot are equal to $\sum_{i=2}^{n} 1/i$).