Bounds on Sorting

Eklavya Sharma

Abstract

This document analyzes lower and upper bounds on the worst-case number of comparisons required for sorting an array of n elements. This is done for both sorting in general and for specific algorithms.

Contents

1 General lower bound

By the decision tree model of computing, we get a lower bound of $\lceil \lg(n!) \rceil$ on the number of comparisons in the worst case.

By Stirling's approximation, we get

$$\lg(n!) = n \lg n - (\lg e)n + \frac{1}{2} \lg n + \lg \sqrt{2\pi} + (\lg e) \left[\frac{1}{12n+1}, \frac{1}{12n} \right]$$

($\lg e \approx 1.4427$ and $\lg \sqrt{2\pi} \approx 1.3257$)

2 Specific algorithms

2.1 Insertion sort

In the worst case, insertion sort performs $\frac{n(n-1)}{2}$ comparisons.

2.2 Insertion sort with binary search

Binary searching an array of size n takes $\lfloor \lg n \rfloor + 1$ comparisons. (Solve the recurrence $f(1) = 1 \wedge f(n) = f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1$)

Therefore, number of comparisons is

$$(n-1) + \sum_{i=1}^{n-1} \lfloor \lg i \rfloor \leq (n-1) + \lfloor \lg((n-1)!) \rfloor$$

2.3 Merge sort

Merging 2 sorted arrays of size m and n can be done in at most $m + n - 1$ comparisons.

In the worst case, merge sort performs $f(n)$ comparisons, where $f(0) = f(1) = 0$ and $f(n) = f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + f\left(\left\lceil \frac{n}{2} \right\rceil\right) + (n-1)$.

The solution to this recurrence is [2]

$$f(n) = n(\lfloor \lg n \rfloor + 1) - 2^{\lfloor \lg n \rfloor + 1} + 1 \in n \lfloor \lg n \rfloor - [0, n-1]$$

This is $O(n)$ higher than the decision-tree lower bound.

2.4 Heapsort

With a binary heap, total number of comparisons for heapsort

$$\leq 2(n - 1 + \lfloor \lg((n-1)!) \rfloor) \leq 2n \lg n - 2(\lg e - 1)n - \lg n + \lg \pi - \frac{5}{6}$$

See [1] for the algorithm and analysis.

2.5 Randomized quicksort

Partitioning an array of size n about a pivot can be done in $n-1$ comparisons.

Let $f(n)$ be the expected number of comparisons required for randomized quicksort. Therefore, $f(0) = f(1) = 0$ and

$$f(n) = (n-1) + \frac{1}{n}\sum_{i=1}^{n}(f(i-1) + f(n-i))$$

$$\begin{aligned}
&f(n) = (n-1) + \frac{1}{n}\sum_{i=1}^{n}(f(i-1) + f(n-i)) \\
&\Rightarrow nf(n) = n(n-1) + 2\sum_{i=0}^{n-1} f(i-1) \\
&\Rightarrow nf(n) - (n-1)f(n-1) = 2(n-1) + 2f(n-1) \\
&\qquad\qquad \text{(subtract equations for } n \text{ and } n-1\text{)} \\
&\Rightarrow \frac{f(n)}{n+1} - \frac{f(n-1)}{n} = \frac{2(n-1)}{n(n+1)} = \frac{4}{n+1} - \frac{2}{n} \\
&\Rightarrow \frac{f(n)}{n+1} - f(0) = \sum_{i=1}^{n}\left(\frac{4}{i+1} - \frac{2}{i}\right) = 2H(n+1) + \frac{2}{n+1} - 4 \quad (H(n) = \textstyle\sum_{i=1}^{n}\frac{1}{i}) \\
&\Rightarrow f(n) = 2((n+1)H(n) - 2n)
\end{aligned}$$

Using the integration bound for the sum of a decreasing function:

$$\sum_{i=a}^{b} f(i) \in \left(\int_a^b f(x)dx\right) + [f(b), f(a)]$$

we get $H(n) \in \ln n + \left[\frac{1}{n}, 1\right]$.

Therefore,

$$\begin{aligned}
&f(n) = 2((n+1)H(n) - 2n) \leq 2n\ln n - 2n + 2H(n) \\
&\leq \left(\frac{2}{\lg e}\right) n\lg n - 2n + 2H(n)
\end{aligned}$$

Since, $\frac{2}{\lg e} \approx 1.3863$, randomized quicksort takes approximately 1.3863 times the number of comparisons by the decision-tree lower bound.

References

[1] Eklavya Sharma. Notes: Heaps. URL: https://sharmaeklavya2.github.io/notes/algorithms/heaps.pdf.

[2] Eklavya Sharma. Notes: Recurrence relations. URL: https://sharmaeklavya2.github.io/notes/math/recurrences.pdf.