Quick Sort: a randomized sorting algorithm

**QSort(L):**

if $|L| \leq 1$ then
  return $L$
else
  choose $p$ from $L$ at random
  /* $p$ is the “pivot” */
  partition $L$ into 3 sublists: $L_{<p}, L_{=p}, L_{>p}$
  return $QSort(L_{<p}) \parallel L_{=p} \parallel QSort(L_{>p})$

Correctness: clear
The “Quick” in “Quick Sort” comes from a very tight inner loop

An idea from Bentley & McIlroy (1993)

Two inner loops:
- moving $b$: scan over $<$, swap $=$, halt on $>$
- moving $c$: scan over $>$, swap $=$, halt on $<$

Swap elements $b$ and $c$, $b++$, $c--$

Repeat until $b$ crosses $c$

When finished, the $=$’s are swapped to the middle
This code uses $n - 1$ comparisons to partition

Using a ternary partitioning scheme is necessary to guarantee good performance when there are a large number of duplicates

**Recursion Analysis**

Consider the recursion tree

If a node has size $m$, the cost of that node is

$\leq m - 1$, and the sizes of its children sum to

$\leq m - 1$

At level $j$, the sizes at level $j$ sum to at most $n - j$

height of recursion tree is $\leq n - 1$

total cost is $\leq n^2$
Define $W := \text{total cost}$

This is a random variable

**Theorem.** $E[W] = O(n \log n)$

Intuition: with every partition, we expect to get an even split, and so it should behave roughly like Merge Sort

Details — more complicated!

Several different proofs . . .

Here is a “slick” one, based on an idea from Alan Siegel
Let $D =$ depth of recursion tree (number of levels)

$D$ is random variable

Then we have $W \leq nD$

So $E[W] \leq nE[D]$

**Goal:** prove $E[D] = O(\log n)$
The recursion tree in more detail...

\[ N_i := \text{size of node } i \]
\[ \mathcal{L}_j := \text{set of indices at level } j \]
\[ T_j := \sum_{i \in \mathcal{L}_j} N_i^2 \]

The \( N_i \)'s and \( T_j \)'s are random variables

**Claim 1:** \( \mathbb{E}[T_j] \leq (2/3)^j n^2 \) for \( j = 0, 1, \ldots \)
Let’s first prove that $E[T_1] \leq (2/3)n^2$

$T_1 = N_2^2 + N_3^2$

Imagine the items are in an array $A[1..n]$ in sorted order

Let $R$ be the index of the pivot in $A$

$R$ is uniformly distributed over $\{1, \ldots, n\}$

$N_2 \leq R - 1$ and $N_3 \leq n - R$

$$E[(R - 1)^2] = \sum_{i=1}^{n} (i - 1)^2/n = \frac{1}{n} \sum_{i=0}^{n-1} i^2$$

$$\leq \frac{1}{n} \int_{0}^{n} x^2 \, dx = \frac{1}{n} \cdot \frac{n^3}{3} = \frac{n^2}{3}$$
The distribution of $n - R$ is the same as that of $R - 1$

Thus, $E[N_2^2] \leq n^2/3$, $E[N_3^2] \leq n^2/3$, and

$E[T_1] = E[N_2^2] + E[N_3^2] \leq (2/3)n^2$

More generally, consider any node $i$ in the tree

“Law of total expectation”:

$$E[N_{2i}^2] = \sum_{m} E[N_{2i}^2 | N_i = m] Pr[N_i = m]$$

$$\leq \sum_{m} (m^2/3) Pr[N_i = m] = (1/3)E[N_i^2]$$

Similarly, $E[N_{2i+1}^2] \leq (1/3)E[N_i^2]$
This shows: $E[T_{j+1}] \leq (2/3)E[T_j]$ for $j = 0, 1, \ldots$

By induction on $j$, we prove Claim 1:

$$E[T_j] \leq (2/3)^j n^2 \quad (j = 0, 1, \ldots)$$

How to use this to prove $E[D] = O(\log n)$?

Recall useful fact:

$$E[D] = \sum_{j \geq 1} \Pr[D \geq j]$$

**Claim 2:** $Pr[D \geq j] \leq E[T_{j-1}]$

Proof:

$$E[T_{j-1}] = E[T_{j-1} \mid D \geq j] \Pr[D \geq j] + E[T_{j-1} \mid D < j] \Pr[D < j]$$

$$\geq 1 \cdot \Pr[D \geq j] + 0$$
\[ E[D] = \sum_{j \geq 1} \Pr[D \geq j] \leq \sum_{j \geq 1} \min\{1, E[T_{j-1}]\} \quad \text{(Claim 2)} \]

\[ \leq \sum_{j \geq 1} \min\{1, (2/3)^{j-1} n^2\} \quad \text{(Claim 1)} \]

Let \( j_0 := \text{smallest integer such that} \)

\[ (2/3)^{j_0-1} n^2 \leq 1 \]

Observe: \( j_0 = O(\log n) \)

First \( j_0 - 1 \) terms contribute \( O(\log n) \) to the sum

The remaining terms are bounded by

\[ 1 + (2/3) + (2/3)^2 + \cdots = 1/(1 - 2/3) = 3 \]
Recapping...

\[ E[D] = O(\log n) \]

\[ W \leq nD \]

\[ E[W] \leq nE[D] = O(n \log n) \]