Quick Sort: a randomized sorting algorithm

**QSort(L):**

choose a “pivot” \( p \) from \( L \) **at random**
partition \( L \) into 3 sublists: \( L_{<p}, L_{=p}, L_{>p} \)

if \( \text{length}(L_{<p}) > 0 \) then \( L_{<p} \leftarrow \text{QSort}(L_{<p}) \)
if \( \text{length}(L_{>p}) > 0 \) then \( L_{>p} \leftarrow \text{QSort}(L_{>p}) \)
return \( \text{concat}(L_{<p}, L_{=p}, L_{>p}) \)
Practical implementation:
• use the fast, in-place 3-way partition algorithm from last time
• this partition algorithm makes at most $n - 1$ comparisons, where $n := \text{length}(L)$

Features:
• Expected # of comparisons = $2n \ln(n) + O(n)$ (for arbitrary inputs)
• With small probability, it can take $\Theta(n^2)$ comparisons
  ◦ if we get unlucky and choose as the pivot the smallest element at every step
• # of swaps $\leq$ # of comparisons $+ O(n)$
• Expected # of swaps = $\frac{1}{3}n \ln(n) + O(n)$ (for random inputs)
  ◦ smaller # of swaps $\implies$ better branch prediction & cache behavior
• Good locality of reference (cache friendly)
• essentially an in place sort
• NOT a stable sort
**Example:**

\[
\begin{array}{cccccccccccc}
2 & 6 & 1 & 6 & 5 & 4 & 1 & 8 & 4 & 5 & 5 & 8 & 3 & 4 & 9 & 8 \\
p=5
\end{array}
\]

\[
\begin{array}{cccccccccccc}
4 & 4 & 1 & 3 & 2 & 4 & 1 & 5 & 5 & 5 & 8 & 8 & 6 & 6 & 8 & 9 \\
p=2
\end{array}
\]

\[
\begin{array}{cccccccccccc}
1 & 1 & 2 & 3 & 4 & 4 & 4 & 5 & 5 & 5 & 6 & 6 & 8 & 8 & 8 & 9 \\
p=8
\end{array}
\]

**Intuition for running time:**

we split the problem into two problems of “roughly equal” size (in linear time) and then solve *both* of them.

reminds us of the recurrence \( T(n) \leq 2T(n/2) + O(n) \)

Master Theorem says: \( T(n) = O(n \log n) \)

BUT ... this is not rigorous: the splitting step is probabilistic, and we may get some “bad” splits
A quick and dirty analysis of Quick Sort

**Idea:** leverage our randomized Quick Select analysis

Imagine that \( L \) is sorted, and for each \( k = 1, \ldots, n \), we can consider the \( k \)th element \( x_k \) in the sorted list

Suppose \( x_k \) appears at levels 0, 1, \ldots, \( D_k \) in the recursion tree

\[ C := \text{number of comparisons} \leq \sum_{k=1}^{n} (D_k + 1) \leq \sum_{k=1}^{n} D_k + n \]

- Each key is involved in at most one comparison with a pivot at each level

**Key observation:** the distribution of \( D_k \) is *precisely the same* as the distribution of the recursion depth of \( QSelect(L, k) \)

**Idea:** *from \( x_k \)'s point of view, we are just running \( QSelect(L, k) \)*

**Therefore:** \( E[D_k] \leq \log_{4/3}(n) + 4 \) for each \( k \), and

\[ E[C] \leq \sum_{k=1}^{n} E[D_k] + n \leq n \log_{4/3}(n) + O(n) \approx 3.48n \ln(n) \]

We’ll give a more careful analysis later: \( E[C] \leq 2n \ln(n) \)
Expected Depth of Quick Sort Recursion

Let $D :=$ depth of the recursion tree for $QSort$ on inputs of length $n$

**Theorem:** $E[D] = O(\log n)$

$E[D]$ can also be viewed as the *average height of a randomly built binary search tree*

- Suppose we have $n$ distinct keys that are randomly permuted
- Now we insert the keys into a binary search tree in the permuted order (starting from the empty tree)
- The size and shape of this tree is exactly the same as that of the recursion tree for Quick Sort

We will prove that $E[D] \leq \alpha \ln(n) + O(1)$, where $\alpha \approx 6.95$

Later, we’ll show $E[D] \leq \beta \ln(n) + O(1)$, where $\beta \approx 4.31$
Proof that $E[D] = O(\log n)$

**Tail sum formula:** $E[D] = \sum_{j \geq 1} \Pr[D \geq j]$

**Union bound:** $\Pr[D \geq j] \leq \sum_{k=1}^{n} \Pr[D_k \geq j]$

Last time, we proved: $\Pr[D_k \geq j] \leq (\frac{3}{4})^j n$

**So we have:** $\Pr[D \geq j] \leq (\frac{3}{4})^j n^2$

A calculation almost identical to that for QSelect

Set $j_0 := \lceil \log_{4/3}(n^2) \rceil$ // = least $j$ such that $(\frac{3}{4})^j n^2 \leq 1$

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

$$= \sum_{j=1}^{j_0-1} \Pr[D \geq j] + \sum_{j=j_0}^{\infty} \Pr[D \geq j]$$

$$\leq \log_{4/3}(n^2) + 4 \approx 6.95 \ln(n)$$
A better bound on running time of Quick Sort

The input is a list $L$ of $n$ items

Let $C_L$ be a random variable representing the number of comparisons made on an input list $L$

Define $\tilde{C}(n)$ to be the maximum value of $E[C_L]$ over all lists $L$ of size at most $n$

**We will show:** $\tilde{C}(n) \leq 2n \ln(n)$

Let $R$ be a random variable representing the relative position (in sorted order) of the randomly chosen pivot

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

If $C_<$ and $C_>$ are random variables representing the number of comparisons made in solving the two subproblems obtained after the partition step, then

$$C_L \leq (n-1) + C_< + C_>$$

# comparisons for partitioning

Total expectation:

$$E[C_L] \leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} \left( E[C_< | R = i] + E[C_> | R = i] \right)$$
We have:

\[
E[C_L] \leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} \left( E[C_\leq | R = i] + E[C_\geq | R = i] \right)
\]

\[
\leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} (\tilde{C}(i - 1) + \tilde{C}(n - i))
\]

\[
= n - 1 + \frac{2}{n} \sum_{i=1}^{n-1} \tilde{C}(i)
\]

We prove by (strong) induction on \(n\) that \(\tilde{C}(n) \leq 2n \ln(n)\) for all \(n \geq 1\)

Assume \(\tilde{C}(i) \leq 2i \ln(i)\) for all \(i < n\)

Let \(L\) be an input of length \(n\)

\[
E[C_L] \leq n - 1 + \frac{2}{n} \sum_{i=1}^{n-1} 2i \ln(i) \leq n - 1 + \frac{4}{n} \int_{1}^{n} x \ln(x)dx
\]

\[
= n - 1 + \frac{4}{n} \left[ \frac{1}{4} x^2 (2 \ln(x) - 1) \right]_1^n = 2n \ln(n) + \frac{1}{n} - 1 \leq 2n \ln(n)
\]

For any \(L'\) of length \(i < n\), we have \(E[C_{L'}] \leq \tilde{C}(i) \leq 2i \ln(i) \leq 2n \ln(n)\)

\[\therefore \tilde{C}(n) \leq 2n \ln(n) \quad \checkmark\]
A better bound on recursion depth of Quick Sort

The recursion tree in more detail . . .

\[ N_1 \]
\[ N_2 \]
\[ N_3 \]
\[ N_4 \]
\[ N_5 \]
\[ N_6 \]
\[ N_7 \]

\( N_i :\) size of node \( i \)

\( \mathcal{L}_j :\) set of indices at level \( j \)

\( T_j :\) \( \sum_{i \in \mathcal{L}_j} N_i^p \)

The \( N_i \)'s and \( T_j \)'s are random variables and \( p > 1 \) is a parameter

**Claim:** \( E[T_j] \leq \gamma^j n^p \) for \( j = 0, 1, 2, \ldots, \) where \( \gamma := \frac{2}{p+1} \)
Proof of claim: \( E[T_j] \leq \gamma^j n^p \) for \( j = 0, 1, 2, \ldots \)
Let’s first prove that \( E[T_1] \leq \gamma n^p \)

\[ T_1 = N_2^p + N_3^p \]

Imagine the items are in \( L \) are sorted
Let \( R \) be the index of the pivot in the sorted list
\( R \) is uniformly distributed over \( \{1, \ldots, n\} \)

\( N_2 \leq R - 1 \) and \( N_3 \leq n - R \)

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

\[
\leq \frac{1}{n} \int_{0}^{n} x^p \, dx = \frac{1}{n} \cdot \frac{n^{p+1}}{p+1} = \frac{n^p}{p+1}
\]
The distribution of $n - R$ is the same as that of $R - 1$
Thus,

$$E[N_2^p] \leq \frac{n^p}{p + 1}, \quad E[N_3^p] \leq \frac{n^p}{p + 1}$$

and

$$E[T_1] = E[N_2^p] + E[N_3^p] \leq \frac{2}{p+1} n^p = \gamma n^p$$

Similar to the calculation we made in QSelect, consider any node $i$ in the recursion tree

“Law of total expectation”:

$$E[N_{2i}^p] = \sum_m E[N_{2i}^p \mid N_i = m] \Pr[N_i = m]$$

$$\leq \sum_m \frac{m^p}{p + 1} \Pr[N_i = m] = \frac{1}{p+1} E[N_i^p]$$

Similarly, $E[N_{2i+1}^p] \leq \frac{1}{p+1} E[N_i^p]$

This implies: $E[T_{j+1}] \leq \frac{2}{p+1} E[T_j] = \gamma E[T_j]$ for $j \geq 0$
Implies claim: $E[T_j] \leq \gamma^j n^p$ for $j \geq 0$ (induction)

**Tail sum formula:** $E[D] = \sum_{j \geq 1} \Pr[D \geq j]$

**Observe:** $D \geq j \iff T_j \geq 1$

**Markov:** $\Pr[T_j \geq 1] \leq E[T_j] \leq \gamma^j n^p$

A calculation almost identical to that for $QSelect$

Seting $j_0 := \lceil \log_{1/\gamma}(n^p) \rceil$ \quad // = least $j$ such that $\gamma^j n^p \leq 1$

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

$$= \sum_{j=1}^{j_0-1} \Pr[D \geq j] + \sum_{j=j_0}^{\infty} \Pr[D \geq j]$$

$$\leq \log_{1/\gamma}(n^p) + (p + 1)/(p - 1)$$
We have
\[ E[D] \leq \beta \ln(n) + O(1), \]
where
\[ \beta = \frac{p}{\ln(1/\gamma)} = \frac{p}{\ln((p + 1)/2)} \]

Graph of $\beta$ as a function of $p$

Choosing $p \approx 3.311$ yields $\beta \approx 4.311$ (compare to 6.95)
Tail inequalities for Quick Sort

**Question:** what is the probability that the actual running time exceeds the expected running time by a significant amount

- We have: $E[D] \approx 4.311 \cdot \ln(n)$  
- Markov says: $\Pr[D \geq 2 E[D]] \leq 1/2$
- But we know, for $\gamma = 2/(p+1)$:
  \[
  \Pr[D \geq t \ln(n)] \leq \gamma^{t \ln(n)} \cdot n^p = n^{t \ln(\gamma)} \cdot n^p = n^{t \ln(\gamma) + p} = n^{-k}, \text{ where } k := t \ln((p + 1)/2) - p
  \]

- $k$ maximized at $p = t - 1$ where $k = t \ln(t/2) - t + 1$

Graph of $k$ as a function of $t$

For $t = 8.622$: $\Pr[D \geq t \ln(n)] \leq n^{-4.97}$
and for $n = 10^6$: $\Pr[D \geq t \ln(n)] \leq 10^{-29.8} \approx 2^{-99}$

- Let $C := \# \text{ of comparisons}$  
  - $E[C] \approx 2n \ln(n)$  
  - $C \leq (D + 1)n$
  \[
  C \geq tn \ln(n) + n \implies D \geq t \ln(n)
  \]
  \[
  \Pr[C \geq tn \ln(n) + n] \leq \Pr[D \geq t \ln(n)] \leq n^{-k}
  \]
  \[
  \Pr[C \geq 8n \ln(n) + n] \leq \Pr[D \geq 8 \ln(n)] \leq n^{-4}
  \]
For $n = 10^6$: $\Pr[C \geq 8n \ln(n) + n] \leq 10^{-24} \approx 2^{-80}$
Taking it one step further . . .

The tail inequality for # of comparisons can easily be improved

Exploit the fact that in the worst case, Quick Sort makes at most $n^2$ comparisons

Define $D^*$ to the first level $j$ such that

$$
\sum_{i \in \mathcal{L}_{j+1}} N_i^2 < n
$$

$$
\Pr[D^* \geq j] \leq \Pr\left[ \sum_{i \in \mathcal{L}_j} N_i^2 \geq n \right] \leq \Pr\left[ \sum_{i \in \mathcal{L}_j} N_i^p \geq n \right] \quad \text{(assuming } p \geq 2) 
$$

$$
= \Pr[T_j \geq n] \leq \mathbb{E}[T_j]/n \leq \gamma^j n^{p-1}
$$

We have $C \leq (D^* + 2)n$: \leq n comparisons at each level $0, \ldots, D^*$, and \leq n in total at levels $D^* + 1, D^* + 2, \ldots$

So we have:

$$
\Pr[D^* \geq t \ln(n)] \leq \gamma^{t \ln(p)} n^{p-1} \leq n^{-k-1} \quad \text{(where } k \text{ is as above)}
$$

Therefore:

$$
\Pr[C \geq tn \ln(n) + 2n] \leq \Pr[D^* \geq t \ln(n)] \leq n^{-k-1}
$$

For $n = 10^6$: $\Pr[C \geq 8n \ln(n) + n] \leq 10^{-30} \approx 2^{-100}$