Randomized Quick sort

Assume no two elements are the same in an array $A$ of size $n$.

Algorithm: Sort($A$)

if size of the array $> 1$,

- pick a pivot element $p$ from $A$, uniformly at random, among elements of $A$

- partition the array into three parts: $A_1, p, A_2$

  where $A_1$ contains all elements $< p$,
  $A_2$ contains all elements $> p$.

  (Note that $A_1$ or $A_2$ may be empty)

- Sort $A_1$ and $A_2$, recursively

  and return $A_1, p, A_2$.

Otherwise return the array.

Note that step requires exactly $n-1$ comparisons as every element of $A$ needs to be compared with $p$. 

How many comparisons does \( \text{Sort}(A) \) make on the average? i.e. what is \( T(n) \)?

\[
T(0) = 1
\]
\[
T(1) = 1
\]
\[
E(T(n)) = \frac{1}{n} \sum_{k=0}^{n-1} \left[ T(k) + T(n-1-k) \right] + n - 1
\]

There are \( n \) choices of pivot, each of them is equally likely.

If your choice \( |A_1| = k \) (\( \Rightarrow |A_2| = n-1-k \)) then the \# of comparisons = \( T(k) + T(n-1-k) \).

Since \( \Pr(|A_1| = k) = \frac{1}{n} \), the contribution of this choice to expected value is \( \frac{1}{n} \left[ T(k) + T(n-1-k) \right] \).

Thus, \[
E(T(n)) = \frac{1}{n} \sum_{k=0}^{n-1} \left[ T(k) + T(n-1-k) \right] + n - 1
\]

\[
= n - 1 + \frac{2}{n} \sum_{k=0}^{n-1} T(k)
\]

Let us guess that this recurrence solves to 

\[
T(n) \leq an \log n + b \quad \text{for some constant } a.
\]
Note that $T(1) = 1$ and it holds as long as $b \geq 1$.

Assume it holds for all values $n < n_j$ and we will prove it for $n$.

$$T(n) \leq n - 1 + \frac{2}{n} \sum_{k=0}^{n-1} (ak \log k + b)$$

$$= n - 1 + \frac{2}{n} \sum_{k=0}^{n-1} (ak \log k) + \frac{2}{n} \sum_{k=0}^{n-1} b = 2b$$

As $\frac{2}{n} \cdot b \cdot n = 2b$

$$\leq n - 1 + 2b + \frac{2}{n} \left[ \sum_{k=0}^{n-2} ak \log \frac{n}{2} + \sum_{k=2}^{n-1} ak \log n \right]$$

$$= n - 1 + 2b + \frac{2}{n} \left[ \sum_{k=0}^{n-1} ak \log n - \sum_{k=0}^{n-1} ak \right]$$

$$= n - 1 + 2b + \frac{2}{n} \left[ a \log n \left( \frac{n(n-1)}{2} \right) - a \left( \frac{n}{2} \left( \frac{n+1}{2} \right) \right) \right]$$

$$\leq n - 1 + 2b + a(n-1) \log n - a \left( \frac{n}{4} \right)$$

$$= a \log n + b - \frac{n}{4} a + b + n - 1 - a \log n$$

$$\leq a \log n + b \quad \text{provided } \frac{n}{4} a \geq b + n - 1$$
or if \( a > 4 \cdot b + 4 \) then 

\[
(4b+4)n + \frac{n}{4} (4b+4) \text{ and } \frac{n}{4} = b + n - 4.
\]

Thus expected number of comparisons quicksort makes is \( O(n \log n) \).

An alternate easier way to show the analysis.

This will use linearity of expectation.

We will ask the question what is the chance that the \( i^{th} \) element in the array will be compared against \( j^{th} \) element in \( A \)?

This comparison can only happen if:

- either the \( i^{th} \) element or the \( j^{th} \) element is chosen as pivot

- none of the elements in \( A \) whose values are between \( i^{th} \) smallest and \( j^{th} \) smallest are chosen as pivot so far.

What is \( \Pr \) of this to happen = \[ \frac{2}{j - b + 1} \]
There are \( j-i+1 \) elements in \( A \) which are exactly between the \( i^{th} \)-smallest and \( j^{th} \)-smallest in \( A \). Out of these only two elements (either the \( i^{th} \) or \( j^{th} \)) is a good choice.

Let \( X_{ij} = \begin{cases} 0 & \text{if } i^{th} \text{- and } j^{th} \text{- smallest are not compared by the algorithm} \\ 1 & \text{if } i^{th} \text{- smallest } \neq j^{th} \text{- smallest are compared during the algorithm} \end{cases} \)

Define \( X = \sum_{1 \leq i < j \leq n} X_{ij} \) = Total \# Comparisons made by the algorithm

\[
E[X] = E\left[ \sum_{1 \leq i < j \leq n} X_{ij} \right] = \sum_{1 \leq i < j \leq n} E(X_{ij})
\]

\[
= \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1} \cdot 1 + \frac{j-i-1}{j-i+1} \cdot 0
\]

\[
= \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1}
\]
\[
\sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} = \sum_{i=1}^{n-1} \left( \sum_{k=2}^{n-i+1} \frac{1}{k} \right)
\]

Note that
\[\sum_{k=1}^{n} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} \approx \ln n\]

Thus
\[\sum_{i=1}^{n-1} \left( \sum_{k=2}^{n-i+1} \frac{1}{k} \right) \leq \sum_{i=1}^{n-1} 2 \ln n \approx O(n \ln n)\]

Observe that this sorting algo always sorts (correct output) but run time (\# comparisons) are expected.

These are called Las Vegas algo.