**QUICKSORT ANALYSIS**

**Worst case timing analysis**: \( T(n) = O(n^2) \)

Quicksort gives a worst case behaviour when the set is already sorted.

Assume the set of elements to be sorted is already sorted and in ascending order.

Consider the first call on partition:

The left_to_right pointer stops at the second element with the cost of one comparison.

The right_to_left pointer stops at the first element after \( n \) comparisons.

The position of the pivot element remains at 1, at the cost of \( n+1 \) comparisons.

Now the set to be sorted contains a left part which is null set and the right part which has \( n-1 \) element.

Hence \( T(n) = n+1 + T(n-1) \) with \( T(0)=T(1)=1 \)

Solving this recurrence by the substitution method:

\[
\begin{align*}
T(n) &= T(n-1) + (n+1) \\
T(n-1) &= T(n-2) + n \\
T(n-2) &= T(n-3) + (n-1) \\
T(n-3) &= T(n-4) + (n-2)
\end{align*}
\]

..........................
\[
\begin{align*}
T(3) &= T(2) + 4 \\
T(2) &= T(1) + 3 \\
T(1) &= T(0) + 2
\end{align*}
\]

Summing all the left hand sides of the above equations:

\[
T(n) + [T(n-1) + T(n-2) + \cdots + T(2) + T(1)]
\]

Summing all the right hand sides of the above equations:

\[
[T(n-1) + T(n-2) + \cdots + T(2) + T(1)] + [T(n)+ (n) + (n-1) + \cdots + 3 + 2] + T(0)
\]

Equating the two sides

\[
T(n) = [T(n)+ (n) + (n-1) + \cdots + 3 + 2] + T(0)
\]

\[
= 1 + 2 + 3 + 4 + \cdots + n + (n+1)
\]

\[
= (n+1)(n+2)/2
\]

\[
= (n^2 + 2n +2)/2 < n^2/2
\]

\[
= O(n^2)
\]

**WORST CASE**

\[
T(n) = O(n^2)
\]
**Average case analysis of quicksort:** \( T(n) = O(n \log n) \)

<table>
<thead>
<tr>
<th>Position of pivot</th>
<th>Left of pivot</th>
<th>Right of pivot</th>
<th>( T(n_{\text{left}}) )</th>
<th>( T(n_{\text{right}}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>n-1</td>
<td>( T(0) )</td>
<td>( T(n-1) )</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>n-2</td>
<td>( T(1) )</td>
<td>( T(n-2) )</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>n-3</td>
<td>( T(2) )</td>
<td>( T(n-3) )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>n-2</td>
<td>n-3</td>
<td>2</td>
<td>( T(n-3) )</td>
<td>( T(2) )</td>
</tr>
<tr>
<td>n-1</td>
<td>n-2</td>
<td>1</td>
<td>( T(n-1) )</td>
<td>( T(1) )</td>
</tr>
<tr>
<td>n</td>
<td>n-1</td>
<td>0</td>
<td>( T(n-1) )</td>
<td>( T(0) )</td>
</tr>
</tbody>
</table>

The number of comparisons for first call on partition:

Assume `left_to_right` moves over \( k \) smaller element and thus \( k \) comparisons.

So when `right_to_left` crosses `left_to_right` it has made \( n-k+1 \) comparisons.

So first call on partition makes \( n+1 \) comparisons.

The average case complexity of quicksort is

\[
T(n) = \text{comparisons for first call on quicksort} + \sum_{1 \leq n_{\text{left}}, n_{\text{right}} \leq n} \left[ T(n_{\text{left}}) + T(n_{\text{right}}) \right] n
\]
\[(n+1) + 2 \left[ T(0) + T(1) + T(2) + \cdots + T(n-1) \right] / n \]

\[nT(n) = n(n+1) + 2 \left[ T(0) + T(1) + T(2) + \cdots + T(n-2) + T(n-1) \right] \]

\[(n-1)T(n-1) = (n-1)n + 2 \left[ T(0) + T(1) + T(2) + \cdots + T(n-2) \right] \]

Subtracting both sides:

\[nT(n) -(n-1)T(n-1) = [n(n+1) - (n-1)n] + 2T(n-1)\]

\[= 2n + 2T(n-1)\]

\[nT(n) = 2n + (n-1)T(n-1) + 2T(n-1)\]

\[= 2n + (n+1)T(n-1)\]

\[T(n) = 2 + (n+1)T(n-1)/n\]

The recurrence relation obtained is:

\[T(n)/(n+1) = 2/(n+1) + T(n-1)/n\]

Using the method of substitution:

\[T(n)/(n+1) = 2/(n+1) + T(n-1)/n\]

\[T(n-1)/n = 2/n + T(n-2)/(n-1)\]

\[T(n-2)/(n-1) = 2/(n-1) + T(n-3)/(n-2)\]

\[T(n-3)/(n-2) = 2/(n-2) + T(n-4)/(n-3)\]

------------------------------------------------------------

------------------------------------------------------------

\[T(3)/4 = 2/4 + T(2)/3\]

\[T(2)/3 = 2/3 + T(1)/2\]
\[ T(1)/2 = 2/2 + T(0) \]

Adding both sides:

\[
T(n)/(n+1) + \left[ T(n-1)/n + T(n-2)/(n-1) + \ldots + T(2)/3 + T(1)/2 \right]
\]

\[= \left[ T(n-1)/n + T(n-2)/(n-1) + \ldots + T(2)/3 + T(1)/2 \right] + T(0) + \]

\[2/(n+1) + 2/n + 2/(n-1) + \ldots + 2/4 + 2/3\]

Cancelling the common terms:

\[
T(n)/(n+1) = 2 \sum_{2<k<(n+1)} 1/k
\]

\[= 2H_{n+1}, \text{ } H_k \text{ is called the kth harmonic number}\]

\[\int_2^{n+1} \frac{dk}{k} \text{ for k varying from 2 to } (n+1)\]

\[<2 \log_e(n+1) - \log_e 2\]

Hence \(T(n) = O((n + 1) 2 \log_e(n+1) - \log_e 2))\)

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

**AVERAGE CASE**

\[ T(n) = O(n \log n) \]