Analysis of Quicksort

Best-case

In the best case, the partition occurs right down the middle. Let

- \( W(n) \) = the work done by Quicksort on an array of size \( n \).
- \( P(n) \) = the work done in (only) partitioning an array of size \( n \).

Thus, the time taken will be proportional to \( W(n) \).

Observe that \( P(n) = cn \) because partitioning requires only a single scan and does only a constant amount of work in each scan step.

In the best case:

\[
W(n) = P(n) + W\left(\frac{n}{2}\right) + W\left(\frac{n}{2}\right)
\]

\[
= P(n) + W\left(\frac{n}{2}\right)
\]

\[
+ W\left(\frac{n}{2}\right)
\]

\[
= P(n) + P\left(\frac{n}{2}\right) + W\left(\frac{n}{4}\right) + W\left(\frac{n}{4}\right)
\]

\[
+ P\left(\frac{n}{2}\right) + W\left(\frac{n}{4}\right) + W\left(\frac{n}{4}\right)
\]

\[
= P(n) + 2P\left(\frac{n}{2}\right) + 4W\left(\frac{n}{4}\right)
\]

\[
= P(n) + 2P\left(\frac{n}{2}\right) + 4P\left(\frac{n}{4}\right) + 8W\left(\frac{n}{8}\right)
\]

\[
= P(n) + 2P\left(\frac{n}{2}\right) + 4P\left(\frac{n}{4}\right) + \ldots + kP(1) + W(0)
\]

\[
< cn + 2c\frac{n}{2} + 4c\frac{n}{4} + \ldots + kc + W(0)
\]

\[
< kcn
\]

\[
= O(kn)
\]
What is $k$? It is the number of times $n$ can be divided until you reach unit size: $\log n$.

Hence, $W(n) = O(n \log n)$. 
Worst-case

In the worst-case, each partition only peels off one element: the partition for \( n \) is into sizes \((n - 2)\) and 1.

Here,

\[
W(n) = P(n) + W(n - 1) \\
= P(n) + P(n - 1) + W(n - 2) \\
= P(n) + P(N - 1) + \ldots + P(1) \\
= cn + c(n - 1) + \ldots + c \\
= c(n + (n - 1) + \ldots + 1) \\
> c\frac{n(n + 1)}{2} \\
= O(n^2)
\]
Unusual-case

Now, suppose each partition divides the array into two unequal sizes: 80% and 20%.

\[
W(n) = P(n) + W\left(\frac{8n}{10}\right) + W\left(\frac{2n}{10}\right)
\]

\[
= P(n) + W\left(\frac{8n}{10}\right)
+ W\left(\frac{2n}{10}\right)
\]

\[
= P(n) + P\left(\frac{8n}{10}\right) + W\left(\frac{8\cdot8n}{10\cdot10}\right) + W\left(\frac{2\cdot8n}{10\cdot10}\right)
\]

\[
= + P\left(\frac{2n}{10}\right) + W\left(\frac{8\cdot2n}{10\cdot10}\right) + W\left(\frac{2\cdot2n}{10\cdot10}\right)
\]

\[
= cn + c\frac{8n}{10} + W\left(\frac{8\cdot8n}{10\cdot10}\right) + W\left(\frac{2\cdot8n}{10\cdot10}\right)
\]

\[
= + W\left(\frac{8\cdot2n}{10\cdot10}\right) + W\left(\frac{2\cdot2n}{10\cdot10}\right)
\]

\[
= P(n) + 2P\left(\frac{n}{2}\right) + 4P\left(\frac{n}{4}\right) + 8W\left(\frac{n}{8}\right)
\]

\[
= P(n) + 2P\left(\frac{n}{2}\right) + 4P\left(\frac{n}{4}\right) + \ldots + kP(1) + W(0)
\]

\[
< cn + cn + cn + \ldots \text{ k times } \ldots + cn
\]

\[
< kcn
\]

\[
= O(kn)
\]

Again, what is k? It is the number of times n can be divided by successive 80-20 cuts to reach unit size: \(\log_{10/8} n = O(\log_e n)\).

Again, \(W(n) = O(n \log n)\)!
Average-case

It is possible to show: $O(n \log n)$ running time even with random partition sizes, but the proof is more complicated.