

## ED2 - Aula 6.pdf - Rafael Manfrim

### 3. Qual é o tempo de execução do Quick Sort em um vetor de comprimento $n$ que já está ordenado em ordem crescente? E se o vetor conter todos os elementos iguais?

Se o vetor já estiver ordenado, tanto em ordem crescente, quanto em ordem decrescente, ou todos os elementos forem iguais, ocorre o pior caso de execução do Quick Sort:  $O(n^2)$ , pois não haverão subdivisões no vetor e para cada  $N$ , serão necessárias  $(n-1)$  comparações.

### 4. O que determina a ocorrência do pior caso na execução do Quick Sort? Quais eventos durante sua execução podem causar um desempenho “degradado” do algoritmo?

O algoritmo de Quick Sort se beneficia das ocorrências de subdivisão do vetor em subvetores após o posicionamento correto do pivô, quanto mais próximo do centro de um subvetor o pivô é colocado, mais eficiente a execução do algoritmo. Com isso é evidente, se torna óbvio que quanto mais próximo das "bordas" o pivô é posicionado após a verificação de todos os números, pior se torna o desempenho. Ou seja, nos casos onde todos os números são iguais, ou o vetor já está ordenado, ocorre o pior caso, pois o pivô sempre será colocado em uma "borda" após as verificações.

### 5. Existe uma versão randomizada do Quick Sort. Como ela funciona e qual vantagem ela oferece comparada a versão clássica?

A versão randomizada do Quick Sort possui exatamente a mesma funcionalidade do Quick Sort normal, sua diferença é que o pivô é escolhido aleatoriamente e em seguida trocado com a última posição do subvetor, depois a execução é a mesma do algoritmo original. Sua vantagem é aumentar a probabilidade de ocorrência de  $O(n \log n)$  e diminuir a probabilidade de  $O(n^2)$  em certos padrões de dados.