INF01124 - Classificação e Pesquisa de Dados - Exercício 2
Entrega no laboratório 29/09/2020
Professor João Comba

1 Quicksort

Em aula vimos o algoritmo de Quicksort. Nesta tarefa é solicitado a implementação de quatro versões deste algoritmo, usando duas estratégias para escolha de particionador, e duas estratégias de particionamento:

Estratégias para escolha do particionador:

1. Implemente o algoritmo de Quicksort, usando como particionador a mediana entre o primeiro, último e elemento na posição média de cada sub-array (mediana de 3);
2. Implemente o algoritmo de Quicksort, usando como particionador um elemento aleatório do array.

Estratégias para particionamento:

1. Particionamento de Lomuto
2. Particionamento de Hoare

Para todas as quatro combinações, teste cada uma das versões do algoritmo com o arquivo *entrada.txt* em anexo. O arquivo de entrada possui um formato simples. Na primeira linha tem um número *n* indicando o número de casos de teste. Nas *n* linhas seguintes, cada linha inicia com un número *m* dizendo quantos elementos inteiros tem o array, seguido de *m* elementos. Um exemplo é dado a seguir:

```
3
4 1 3 2 4
5 5 4 3 2 1
10 1 3 5 7 9 2 4 6 8 10
```

Para cada algoritmo, gere um arquivo na saída (*saida-primeiro.txt* e *saida-aleatorio.txt*) de formato idêntico, mas com os elementos ordenados, como mostrado a seguir.

```
3
4 1 2 3 4
5 1 2 3 4 5
10 1 2 3 4 5 6 7 8 9 10
```

Além disso, gere as estatísticas para cada algoritmo nos arquivos *stats-primeiro.txt* e *stats-aleatorio.txt* contendo as seguintes informações para cada caso de teste:

1. número total de *swaps* (troca entre 2 elementos)
2. número total de chamadas recursivas
3. tempo de execução

Um código exemplo foi distribuído junto com as notas sobre a aula de Quicksort. Um pseudocódigo do algoritmo pode ser encontrado em `https://en.wikipedia.org/wiki/Quicksort#Lomuto_partition_scheme`

Vídeo Quicksort in 4 minutes: `https://www.youtube.com/watch?v=Hoixgm4-P4M`

2 Desafio Bonus - 25% extra

Resolva o problema ”10810 - Ultra-QuickSort”

`https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=20&page=show_problem&problem=1751`

Este é um problema de maratona de programação. O problema será considerado completo se for aceito para no site da https://uva.onlinejudge.org

3 Desafio Bonus - 25% extra

Este problema eu pessoalmente acho muito interessante, embora sua solução esteja mais relacionada com a teoria de combinatória. Portanto, não espero que seja resolvido, mas para quem aprecia desafios e aprender algo mais avançado pode considerá-lo desafiador. Duas dicas. Primeiro, este problema requer achar uma fórmula fechada para contar o número de heaps diferentes em árvores multi-árias. Segundo, o resultado excede o que se pode representar como um inteiro, e portanto requer uma implementação adicional para suportar números grandes.

O problema chama-se ”10247 - Complete Tree Labeling”

`https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=14&page=show_problem&problem=1188`

Este problema será considerado completo se for aceito para no site da https://uva.onlinejudge.org. Além disso, eu quero um texto explicando brevemente como o problema foi resolvido.