Atividade Prática I - Estrutura de Dados II

Giulia de Araujo Freulon[1], Juliana Gonçalves Camara[1]

[1] Ciência da Computação – Universidade Federal do Maranhão (UFMA)
São Luís – MA – Brasil

{giulia.freulon, camara.juliana}@discente.ufma.br

1. Análise Assintótica

1.1. 1ª questão

O método EncontraMaior no arquivo "PriorityQueueLargest" itera sobre os elementos de V1 n vezes, o que nos dá uma complexidade de O(n). Depois, ele adiciona o elemento de V1 ao max-heap, isso tem complexidade O(log k), pois o max-heap tem tamanho k. Caso o tamanho do max-heap exceda k, ele faz uma operação de remoção do heap, o que leva O(log k). Por fim, ele pega o elemento no topo do max-heap que tem complexidade O(1). Se juntarmos tudo, teremos:

$$n * (\log k + \log k) + 1$$
$$n * (2\log k) + 1$$

Podemos simplificar isso para uma complexidade de:

$$O(n*\log k)$$

1.2. 2ª questão

O método calculaMediana do arquivo "segundaquestao" copia os vetores 1 e 2 para um vetor conjunto, isso necessita de n movimentos para cada vetor, totalizando, assim 2n. Depois, o vetor conjunto é ordenado com MergeSort, o que tem complexidade O(n log n), portanto O(2n log 2n). Os próximos passos para calcular a mediana tem complexidade constante, por isso não os incluiremos. Somando os valores temos:

$$2n + 2n \log 2n$$

Simplificando, temos uma complexidade equivalente a:

$$O(n \log n)$$

1.3. 3ª questão

- Letra A

O método temParesComDistanciaT do arquivo "terceiraquestaoA" precisa de dois loops que vão até n para iterar sobre o vetor e comparar seus elementos, portanto sua complexidade é:

$$O(n^2)$$

- Letra B

O método temParesComDistanciaT do arquivo "terceiraquestaoB" utiliza MergeSort para ordenar o vetor, isso tem complexidade O(n log n) e depois utiliza pesquisa binária para encontrar o valor alvo, tendo complexidade O(log n). Desse modo, temos:

$$n \log n + \log n$$

Isso resultado em uma complexidade de:

$$O(n \log n)$$

1.4. 4ª questão

O método bhsiSort no arquivo "BHSISort" constrói um Max-Heap do vetor de entrada com o método buildHeap. Isso tem complexidade de O(n).

Depois, o método encontrarMenores é chamado pelo vetor começo, este método tem complexidade O(n*x), onde x é o número de elementos correspondente à porcentagem %E de entrada. O vetor fim chama o método encontrarMaiores, que também tem complexidade O(n*x). Em seguida, o vetor meio chama o método removerElementosComuns, que tem complexidade O(n).

Então, ordenamos os vetores começo e fim utilizando selectionSort, que tem complexidade $O(n^2)$, o que nos dá $O(2n^2)$. O próximo vetor a ser ordenado é o meio, que utiliza InsertionSort, por seu tamanho, temos O(n-2x). Por fim, juntamos os vetores com o método combinarArrays, com complexidade de O(n).

Juntando tudo teremos:

$$n + 2(n*x) + n + 2(n^2) + n-2x + n$$

Simplificando essa expressão, ficamos com a complexidade:

$$O(n^2)$$

2. Resultados

Os vetores de inteiros, doubles e strings estão limitados à cinco dígitos ou caracteres.

2.1. 1ª questão

Para vetores P com tamanho máximo de 5 elementos:

- Vetor Pequeno (1.000 elementos)

- De inteiros:

```
Maiores rj-esimos do Vetor = [12138, 69383, 42812, 58619, 56160]

Tempo de execução: 2ms
Movimentações: 7714
Comparações: 2704
```

- De doubles:

```
Maiores rj-esimos do Vetor = [48318.93433111046, 20531.437786302093, 15430.777702071371, 44535.16737146275, 75719.37742188488]

Tempo de execução: 1ms
Movimentações: 7887
Comparações: 2877
```

- De strings:

```
Maiores rj-esimos do Vetor = [xXb8b, aoE7j, G1saS, p2sZ7, dwsfi]

Tempo de execução: 1ms
Movimentações: 6754
Comparações: 1744
```

- Vetor Médio (100.000 elementos)

- De inteiros:

```
Maiores rj-esimos do Vetor = [29617, 87683, 98469, 83421, 43240]

Tempo de execução: 46ms
Movimentações: 657275
Comparações: 157265
```

- De doubles:

```
Maiores rj-esimos do Vetor = [43929.18742303109, 25455.759669191262, 87256.08292641732, 76660.53066559351, 62969.90051335467]

Tempo de execução: 85ms
Movimentações: 704348
Comparações: 204338
```

- De strings:

```
Maiores rj-esimos do Vetor = [lNsEe, LVF8T, M9ORU, gutqJ, bvM1b]

Tempo de execução: 126ms
Movimentações: 722612
Comparações: 222602
```

- Vetor Grande (1.000.000 elementos)

- De inteiros:

```
Maiores rj-esimos do Vetor = [3284, 41909, 55964, 92723, 81149]

Tempo de execução: 1364ms
Movimentações: 7250119
Comparações: 2250109
```

- De doubles:

```
Maiores rj-esimos do Vetor = [27006.349866367796, 2027.2945688684717, 94788.02481560413, 53558.900283085706, 27902.325940241248]

Tempo de execução: 1673ms
Movimentações: 7949330
Comparações: 2949320
```

- De strings:

```
Maiores rj-esimos do Vetor = [aFffo, tkHvC, 25TwN, ZP02H, lImiN]

Tempo de execução: 1339ms
Movimentações: 7149070
Comparações: 2149060
```

2.2. 2ª questão

- Vetor Pequeno (1.000 elementos)

- De inteiros:

```
Mediana: 50563

Tempo de execução: 3ms
Movimentações: 4005
Comparações: 11960
```

- De doubles:

```
Mediana: 49788.86522436749

Tempo de execução: 6ms
Movimentações: 8010
Comparações: 23924
```

- De strings:

```
Mediana: V4Kk2V6jNc

Tempo de execução: 4ms
Movimentações: 12013
Comparações: 35886
```

- Vetor Médio (100.000 elementos)

- De inteiros:

```
Mediana: 50199

Tempo de execução: 144ms
Movimentações: 400010
Comparações: 1199933
```

- De doubles:

```
Mediana: 49865.86324547614

Tempo de execução: 70ms
Movimentações: 804025
Comparações: 2411852
```

- De strings:

```
Mediana: UtUDEUtVI2

Tempo de execução: 92ms
Movimentações: 1204033
Comparações: 3611813
```

- Vetor Grande (1.000.000 elementos)

- De inteiros:

```
Mediana: 50002

Tempo de execução: 796ms
Movimentações: 4000011
Comparações: 11999918
```

- De doubles:

```
Mediana: 49955.70660169704

Tempo de execução: 1044ms
Movimentações: 8000022
Comparações: 23999858
```

- De strings:

```
Mediana: UzRm4UzS03

Tempo de execução: 1374ms
Movimentações: 12000031
Comparações: 35999798
```

2.3. 3ª questão

- Letra A

- Vetor Pequeno (1.000 elementos)

```
T:1
Encontrou pelo menos um par com diferença T

Tempo de execução: 2ms
Comparações: 19312
```

- Vetor Médio (100.000 elementos)

```
T:5
Encontrou pelo menos um par com diferença T

Tempo de execução: 3ms
Comparações: 36159
```

- Vetor Grande (1.000.000 elementos)

```
T:8
Encontrou pelo menos um par com diferença T

Tempo de execução: 4ms
Comparações: 152774
```

- Letra B

- Vetor Pequeno (1.000 elementos)

```
T:18
Encontrou pelo menos um par com diferença T

Tempo de execução: 2ms
Movimentações: 3550
Comparações: 7373
```

- Vetor Médio (100.000 elementos)

```
T:12
Encontrou pelo menos um par com diferença T

Tempo de execução: 133ms
Movimentações: 200039
Comparações: 599988
```

- Vetor Grande (1.000.000 elementos)

```
Encontrou pelo menos um par com diferença T

Tempo de execução: 374ms
Movimentações: 2000040
Comparações: 5999961
```

2.4. 4ª questão

- Vetor Pequeno (1.000 elementos)

- De inteiros:

```
Tempo de execução: 23ms
Movimentações: 137066
Comparações: 630564
```

- De doubles:

```
Tempo de execução: 19ms
Movimentações: 160802
Comparações: 631912
```

- De strings:

```
Tempo de execução: 27ms
Movimentações: 203519
Comparações: 648771
```

- Vetor Médio (10.000 elementos)

- De inteiros:

```
Tempo de execução: 224ms
Movimentações: 5506827
Comparações: 61505733
```

- De doubles:

```
Tempo de execução: 302ms
Movimentações: 2475203
Comparações: 64723042
```

- De strings:

```
Tempo de execução: 576ms
Movimentações: 2411356
Comparações: 64630430
```

- Vetor Grande (100.000 elementos)

- De inteiros:

```
Tempo de execução: 26407ms
Movimentações: 3074098041
Comparações: 7117750063
```

- De doubles:

```
Tempo de execução: 25048ms
Movimentações: 5709555
Comparações: 7505061423
```

- De strings:

```
Tempo de execução: 64453ms
Movimentações: 58608541
Comparações: 6862225224
```