Universidade Estadual de Mato Grosso do Sul

Ciência da Computação

Algoritmos e Estruturas de dados II (AED II)

Relatório dos resultados obtidos nos testes de ordenação

João Vitor de Oliveira

RGM: 38342

Dourados-MS

2021

Algoritmos de ordenação

Os algoritmos de ordenação estão presentes no dia a dia da sociedade e, recorremos a eles em muitas situações presentes no cotidiano. Um exemplo disso, é quando queremos encontrar um dado item em uma lista ou tabelas com muitos itens, se esses itens estiverem postos de forma desordenada, teremos muita dificuldade em localizar o item requerido. Porém, se os itens estiverem ordenados (seja por ordem alfabética ou por numérica) com certeza será mais fácil encontrar o item requerido.

Outro exemplo da utilização dos algoritmos de ordenação no dia a dia, é listar quantas pessoas passaram em um vestibular, com a ordenação por pontuação, esse processo de listagem passa a ser mais rápido. Deste modo, é possível concluir que os algoritmos de ordenação facilitam alguns problemas, fazendo com que se economize tempo e muitas vezes dinheiro.

Tendo em base o que foi dito, esse trabalho tem como objetivo mostrar, discutir e analisar os resultados – utilizando gráficos e tabelas para auxiliar - obtidos nos testes com alguns algoritmos de ordenação, com conjuntos pequenos e grandes de números, dispostos em ordem aleatória, crescente e decrescente.

Os algoritmos de ordenação testados foram os seguintes:

- Bubble-sort original (ordenação por bolha);
- Bubble-sort melhorado (ordenação por bolha com critério de parada);
- Insertion-sort (ordenação por inserção);
- Mergesort (ordenação por intercalação);
- Quicksort (ordenação rápida) com pivô sendo o último elemento;
- Quicksort com pivô sendo um elemento aleatório;
- Quicksort com pivô sendo a mediana de três;
- Heapsort (ordenação em Heap).

Para todos os testes feitos, foi usado o sistema operacional Linux Mint 19.1 Tessa, processador Intel Core i7-7500U – 2,70 GHz e 8Gb de memória RAM. Abaixo segue as tabelas e gráficos, mostrando o tempo médio (em segundos) que cada algoritmo gasta para ordenar os diferentes tipos de conjuntos.

Legenda para as tabelas e gráficos:

- O – original;
- M – melhorado;
- PU – Pivô sendo último elemento;
- PA – Pivô sendo um elemento aleatório;
- PM3 – Pivô sendo a mediana de 3.

Ordenação dos Conjuntos em ordem aleatória:

Algoritmos	**Números de elementos**					
	10	**100**	**1000**	**10000**	**100000**	**500000**
Bubble-sort O	0,000	0,000	0,006	0,368	42,499	1038,515
Bubble-sort M	0,000	0,000	0,004	0,288	32,177	813,956
Insertion-sort	0,000	0,000	0,002	0,067	5,982	147,005
Mergesort	0,000	0,000	0,000	0,003	0,020	0,096
Quicksort-PU	0,000	0,000	0,000	0,004	0,017	0,079
Quicksort-PA	0,000	0,000	0,000	0,004	0,017	0,077
Quicksort-PM3	0,000	0,000	0,000	0,004	0,017	0,076
Heapsort	0,000	0,000	0,001	0,005	0,033	0,175

Algoritmos	**Números de elementos**		
	1000000	**10000000**	**100000000**
Mergesort	0,190	2,158	24,359
Quicksort-PA	0,156	1,751	19,647
Quicksort-PM3	0,158	1,773	19,879
Heapsort	0,359	5,031	71,163

Ordenação dos Conjuntos em ordem crescente:

Algoritmos	**Números de elementos**					
	10	**100**	**1000**	**10000**	**100000**	**500000**
Bubble-sort O	0,000	0,000	0,005	0,212	20,558	512,849
Bubble-sort M	0,000	0,000	0,000	0,000	0,001	0,001
Insertion-sort	0,000	0,000	0,000	0,000	0,002	0,002
Mergesort	0,000	0,000	0,000	0,003	0,012	0,049

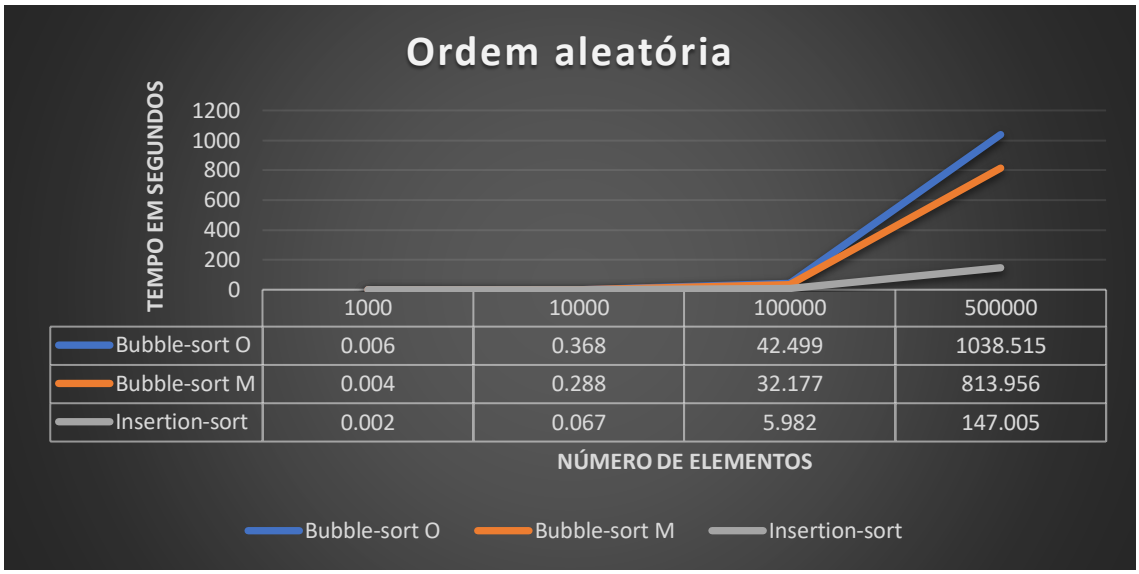
Quicksort-PU	0,000	0,000	0,003	0,103	9,507	240,855
Quicksort-PA	0,000	0,000	0,000	0,003	0,009	0,039
Quicksort-PM3	0,000	0,000	0,000	0,002	0,007	0,028
Heapsort	0,000	0,000	0,001	0,004	0,029	0,120

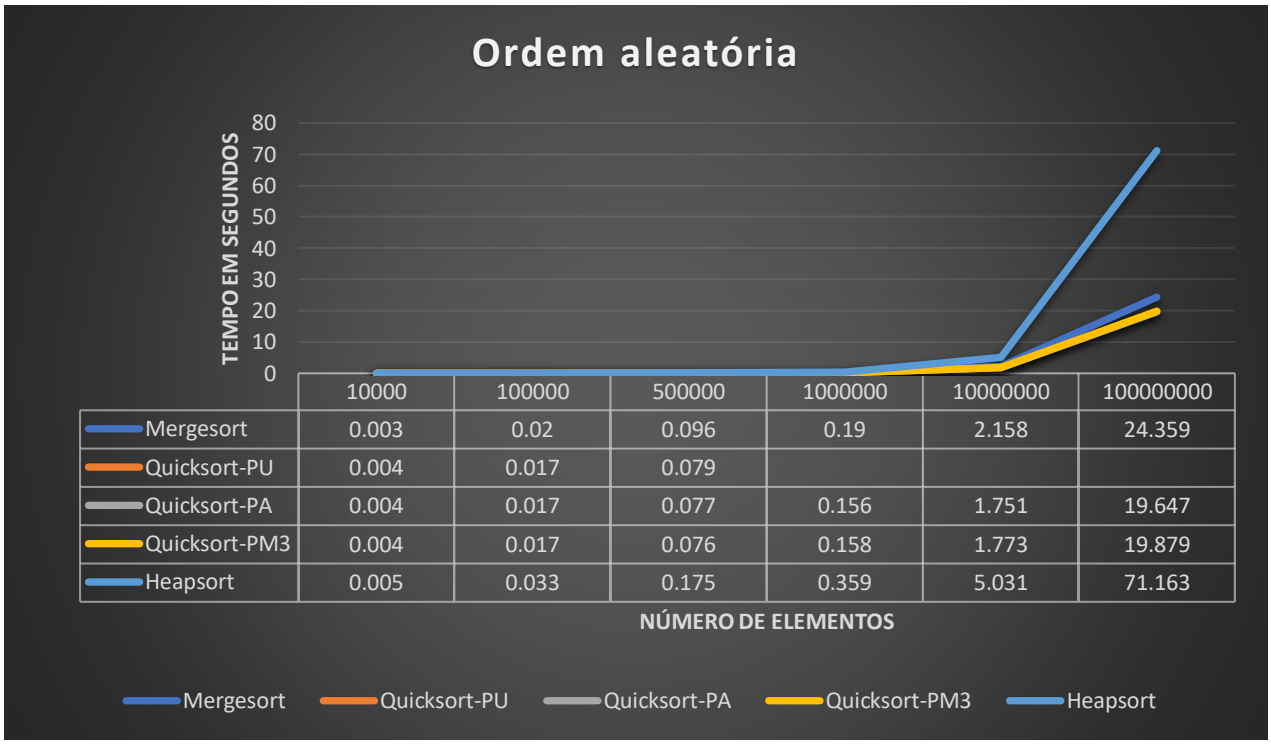
Algoritmos	**Números de elementos**		
	1000000	**10000000**	**100000000**
Mergesort	0,099	1,092	12,248
Quicksort-PA	0,077	0,825	9,141
Quicksort-PM3	0,054	0,574	6,371
Heapsort	0,244	2,771	30,918

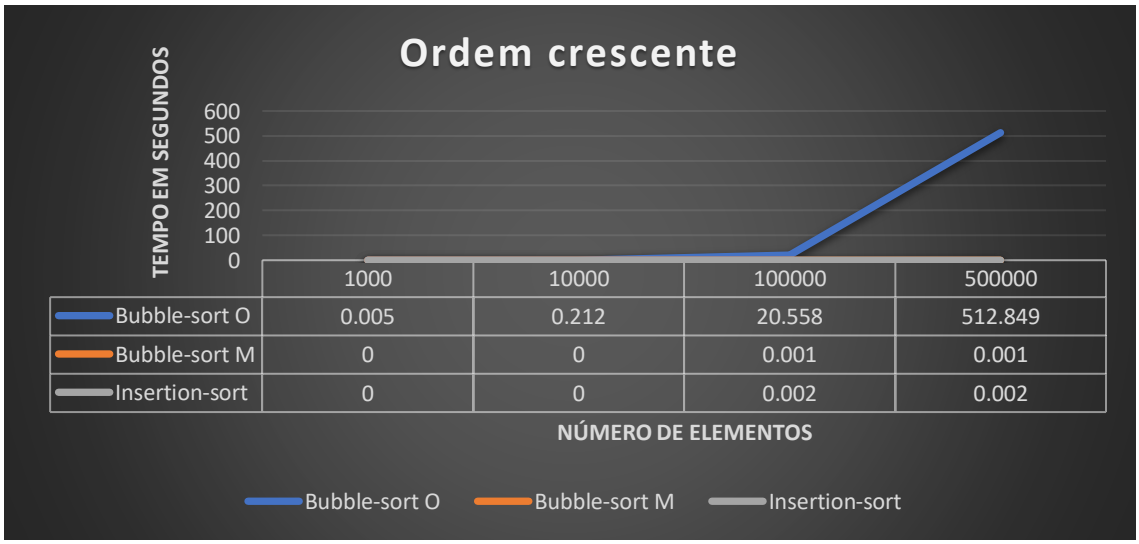
Ordenação dos Conjuntos em ordem decrescente:

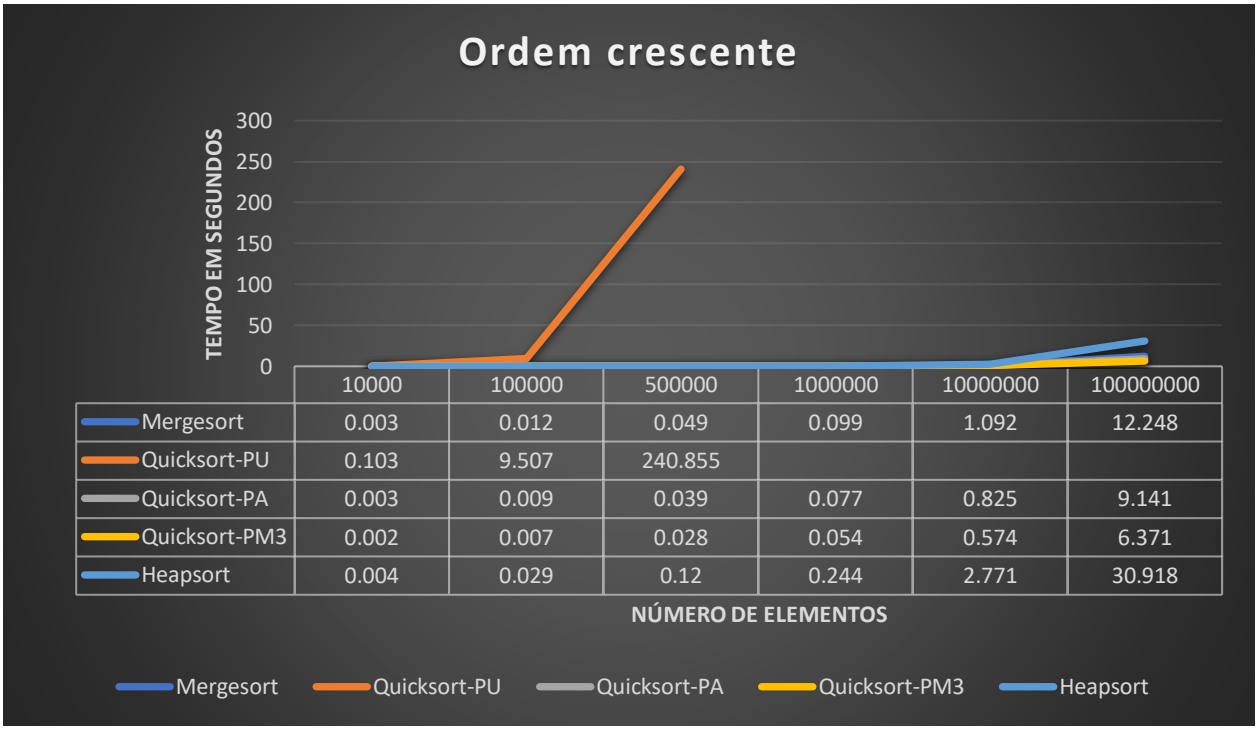
Algoritmos	**Números de elementos**					
	10	**100**	**1000**	**10000**	**100000**	**500000**
Bubble-sort O	0,000	0,000	0,007	0,353	34,726	884,032
Bubble-sort M	0,000	0,000	0,005	0,274	26,908	658,355
Insertion-sort	0,000	0,000	0,004	0,128	11,824	299,523
Mergesort	0,000	0,000	0,000	0,003	0,012	0,051
Quicksort-PU	0,000	0,000	0,003	0,107	9,731	244,736
Quicksort-PA	0,000	0,000	0,000	0,003	0,009	0,039
Quicksort-PM3	0,000	0,000	0,003	0,058	4,851	121,372
Heapsort	0,000	0,000	0,001	0,004	0,025	0,115

Algoritmos	**Números de elementos**		
	1000000	**10000000**	**100000000**
Mergesort	0,099	1,088	12,190
Quicksort-PA	0,077	0,840	9,308
Quicksort-PM3	484,750	X	X
Heapsort	0,245	2,701	30,429

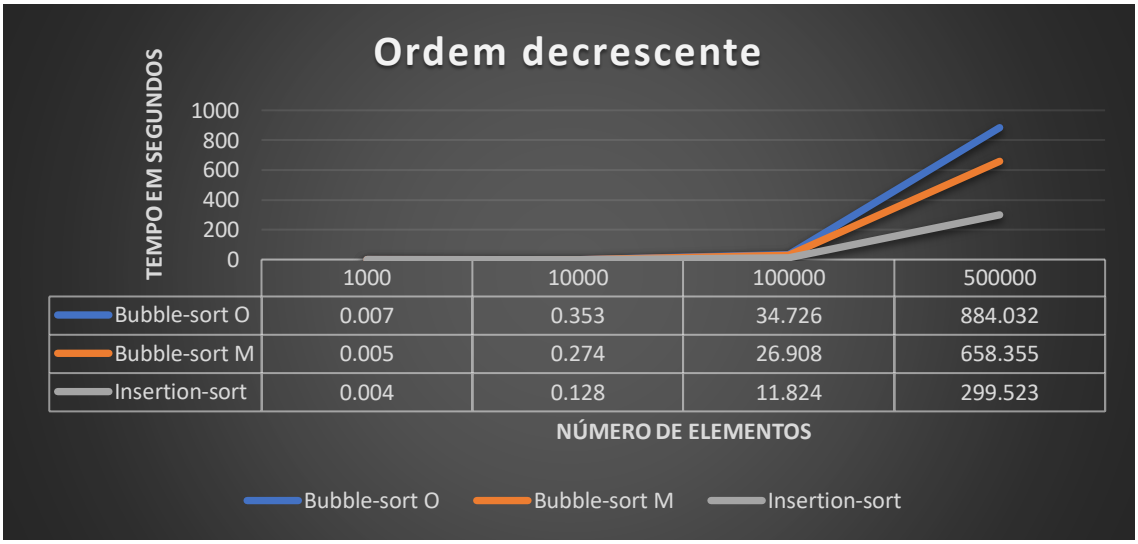

Ordem aleatória
TEMPO EM SEGUNDOS
1200
1000
800
600
400
200
0
1000 10000 100000 500000
Bubble-sort O 0.006 0.368 42.499 1038.515
Bubble-sort M 0.004 0.288 32.177 813.956
Insertion-sort 0.002 0.067 5.982 147.005
NÚMERO DE ELEMENTOS
Bubble-sort O
Bubble-sort M
Insertion-sort


Ordem aleatória
TEMPO EM SEGUNDOS
80
70
60
50
40
30
20
10
0
10000 100000 500000 1000000 10000000 100000000
Mergesort 0.003 0.02 0.096 0.19 2.158 24.359
Quicksort-PU 0.004 0.017 0.079
Quicksort-PA 0.004 0.017 0.077 0.156 1.751 19.647
Quicksort-PM3 0.004 0.017 0.076 0.158 1.773 19.879
Heapsort 0.005 0.033 0.175 0.359 5.031 71.163
NÚMERO DE ELEMENTOS
Mergesort
Quicksort-PU
Quicksort-PA
Quicksort-PM3
Heapsort

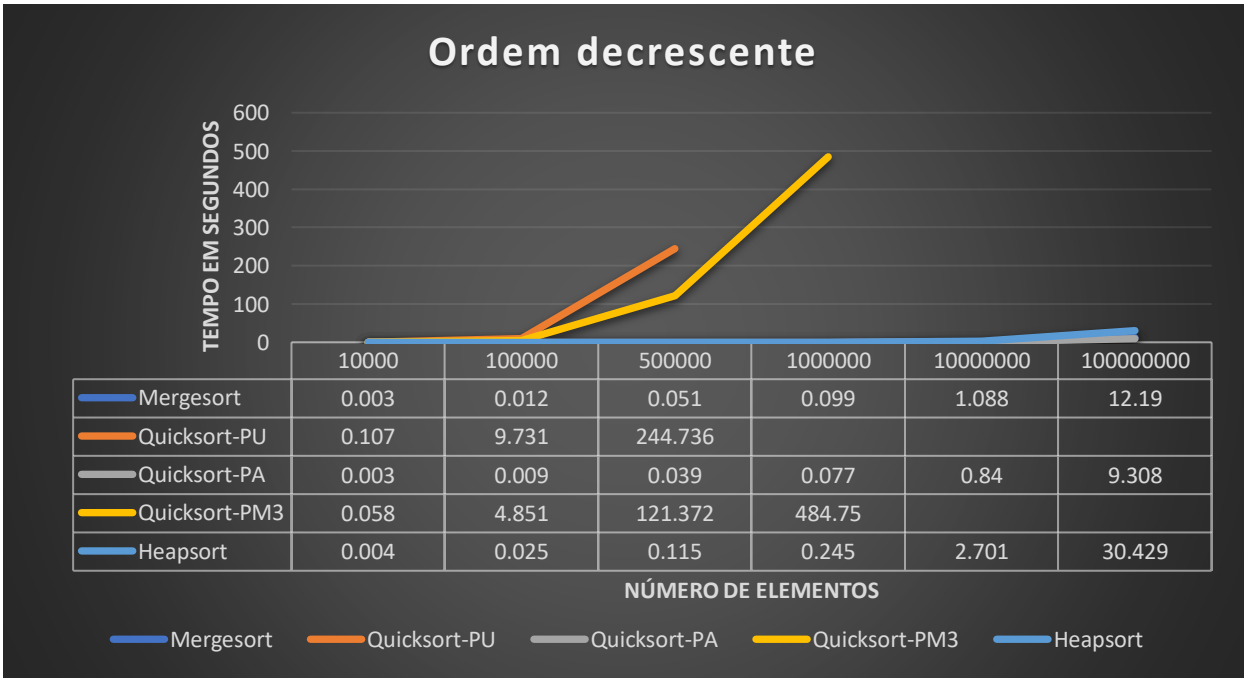

Ordem crescente
TEMPO EM SEGUNDOS
600
500
400
300
200
100
0
1000 10000 100000 500000
Bubble-sort O 0.005 0.212 20.558 512.849
Bubble-sort M 0 0 0.001 0.001
Insertion-sort 0 0 0.002 0.002
NÚMERO DE ELEMENTOS
Bubble-sort O
Bubble-sort M
Insertion-sort



	10000	100000	500000	1000000	10000000	100000000
Mergesort	0.003	0.012	0.049	0.099	1.092	12.248
Quicksort-PU	0.103	9.507	240.855			
Quicksort-PA	0.003	0.009	0.039	0.077	0.825	9.141
Quicksort-PM3	0.002	0.007	0.028	0.054	0.574	6.371
Heapsort	0.004	0.029	0.12	0.244	2.771	30.918



	1000	10000	100000	500000
Bubble-sort O	0.007	0.353	34.726	884.032
Bubble-sort M	0.005	0.274	26.908	658.355
Insertion-sort	0.004	0.128	11.824	299.523



	10000	100000	500000	1000000	10000000	100000000
Mergesort	0.003	0.012	0.051	0.099	1.088	12.19
Quicksort-PU	0.107	9.731	244.736			
Quicksort-PA	0.003	0.009	0.039	0.077	0.84	9.308
Quicksort-PM3	0.058	4.851	121.372	484.75		
Heapsort	0.004	0.025	0.115	0.245	2.701	30.429

Como visto acima, temos 8 algoritmos de ordenação, que se diferem no funcionamento e nos tempos gastos para ordenar diferentes tipos de conjuntos, principalmente por causa de sua complexidade. No Bublle-sort Original, Bublle-sort Melhorado e Insertion-sort sua complexidade é quadrática, logo eles gastam mais tempo para ordenar um conjunto do que os outros, quando o mesmo estão em ordem aleatória e em decrescente, mas quando o conjunto está em ordem crescente, o Bubble-sort Melhorado e Insertion-sort cai no melhor caso deles, O(n), como mostrado no gráfico, quando cai nesse caso eles fazem a ordenação em milissegundos ou menos dos elementos de 100000 e 500000, já o Bubble-sort Original continua em $O(n^2)$.

Os Quicksort, Mergesort e o Heapsort são mais rápidos do que os outros apresentados acima, por causa da complexidade, no caso médio, que é O(n log n). Isso é comprovado com a ordenação de conjuntos com grandes números de elementos, onde a diferença é gritante quando se trata de conjuntos em ordem aleatória, um exemplo disso é quando se ordena 500000 elementos, onde o tempo de ordenação está abaixo de 1 segundo, enquanto os de ordem quadrática demora mais de 100 segundos.

O Quicksort com pivô sendo o último elemento, quando usado em conjuntos de ordem crescente e decrescente, cai no seu pior caso $O(n^2)$, deixando a ordenação mais demorada se comparada com o Mergesort e o Heapsort. O Quicksort com o pivô sendo a mediana de 3, quando testado em conjuntos em ordem decrescente, o tempo de ordenação é lento se comparado com outras ordens, pois cai em seu pior caso, no caso de ordenar acima de 1000000, é muito lento e inviável, por isso na tabela não mostra o tempo que ele gasta para ordenar nesse tipo de caso. Nos conjuntos com poucos elementos (abaixo de 1000), a diferença de tempo que cada algoritmo gasta para ordenar são parecidas, portanto não foi apresentada nos gráficos, apenas nas tabelas para se ter uma noção.