Algoritmos de Ordenação

Tópicos

- Introdução
- Bubble Sort
- Selection Sort
- Quick Sort
- Insertion Sort
- Heap Sort
- Gnome Sort
- Merge Sort
- Shell Sort

BUBBLE SORT

- O Bubble Sort é um algoritmo de ordenação que percorre repetidamente uma lista de elementos, comparando pares adjacentes e trocando-os se estiverem na ordem errada. Ele continua esse processo até que toda a lista esteja ordenada. Os elementos "borbulham" gradualmente para suas posições corretas ao longo das iterações. Embora seja fácil de entender e implementar, o Bubble Sort pode ser ineficiente para listas grandes, pois requer muitas comparações e trocas. Sua complexidade de tempo é O(n^2), onde 'n' é o número de elementos na lista.

Tempo:

Pior caso: O(n^2) - ocorre quando a lista está em ordem inversa e requer várias passagens para ordenar todos os elementos.
Melhor caso: O(n) - ocorre quando a lista já está completamente ordenada e nenhuma troca é necessária.
Caso médio: O(n^2) - o desempenho geralmente é quadrático, tornando-o ineficiente para grandes conjuntos de dados.

Comparações:

O(n^2) - o Bubble Sort compara cada par de elementos adjacentes para determinar se eles estão fora de ordem e precisam ser trocados.

Trocas:

O(n^2) - o Bubble Sort realiza trocas sempre que encontra um par de elementos fora de ordem e precisa reposicioná-los.

VANTAGENS

- Simplicidade de entendimento e implementação.
- Desempenho razoável para conjuntos de dados pequenos ou quase ordenados.

INSERTION SORT

- O Insertion Sort é um algoritmo de ordenação que percorre a lista de elementos, construindo uma sublista ordenada à medida que avança. Ele funciona da mesma maneira que classificaríamos um conjunto de cartas em nossas mãos.

MERGE SORT

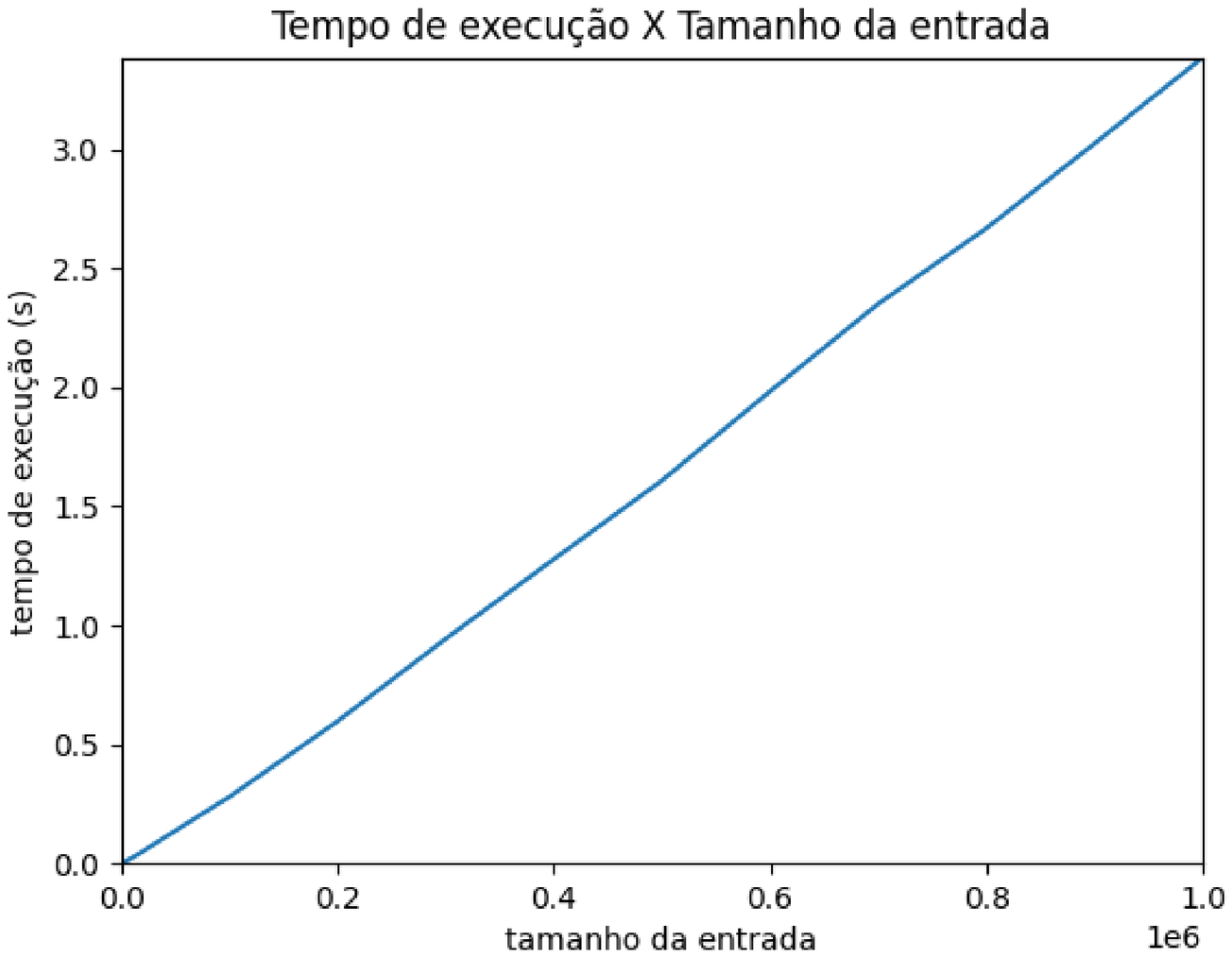
- O Merge Sort é um algoritmo de ordenação eficiente e baseado no conceito de divisão e conquista. Ele divide a lista não ordenada em sublistas menores, ordena essas sublistas de forma recursiva e, em seguida, combina-as para obter a lista ordenada final.

MERGE SORT

- Pior caso:
  - O(n log n)
- Caso médio:
  - O(n log n)
- Melhor caso:
  - O(n log n)

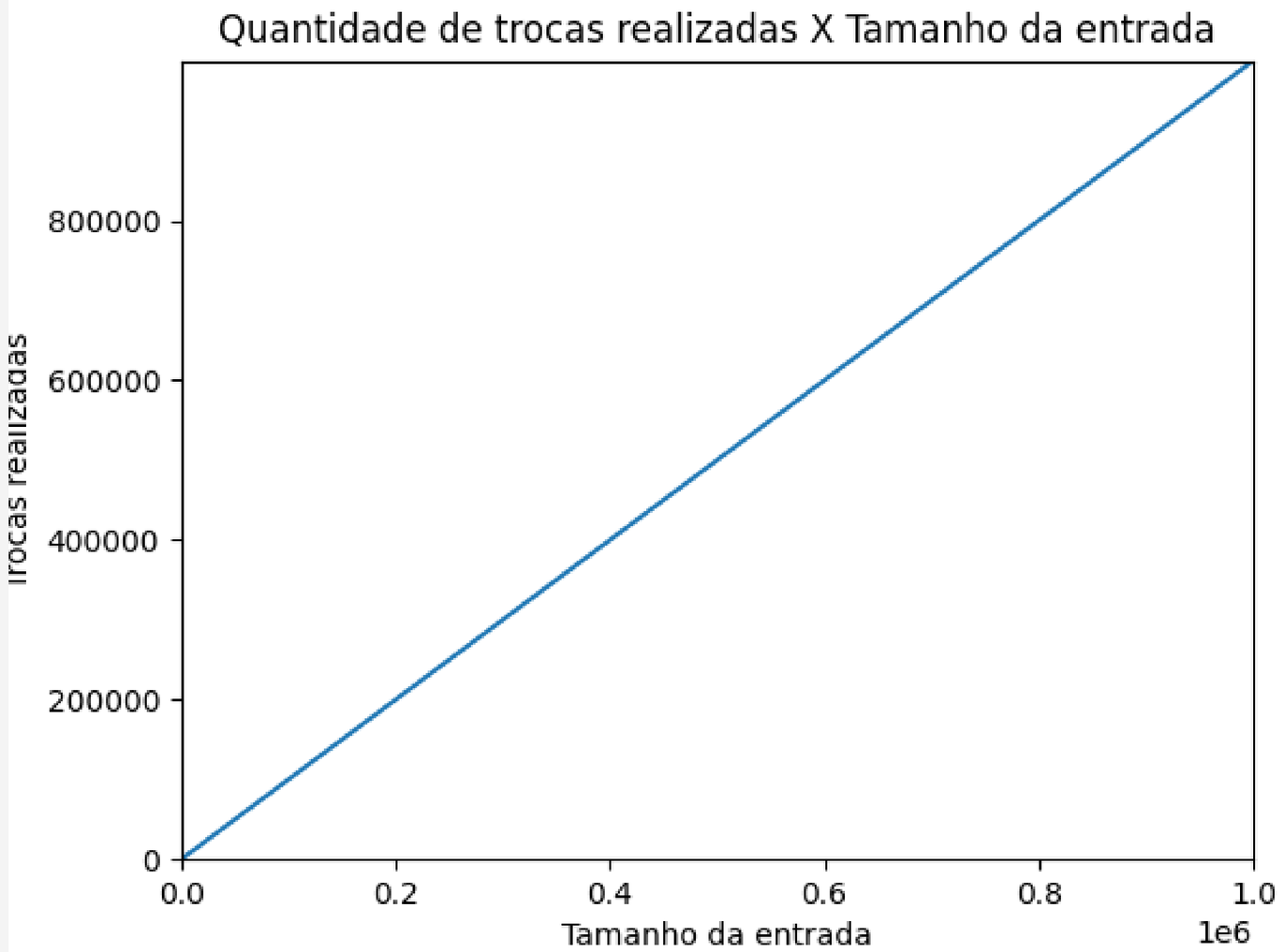
Merge Sort

- Caso médio



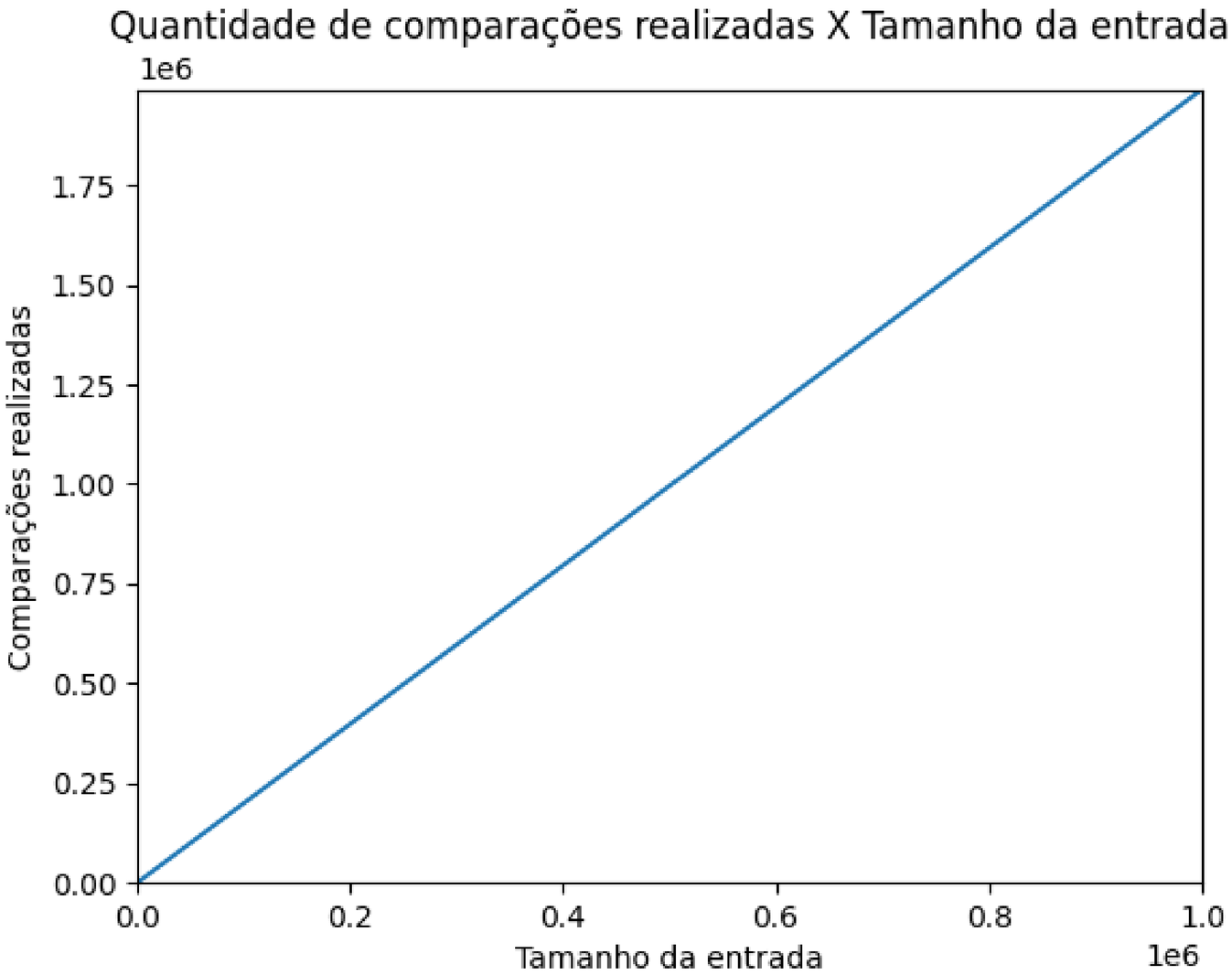
Merge Sort

- Caso médio



Merge Sort

- Caso médio



SHELL SORT

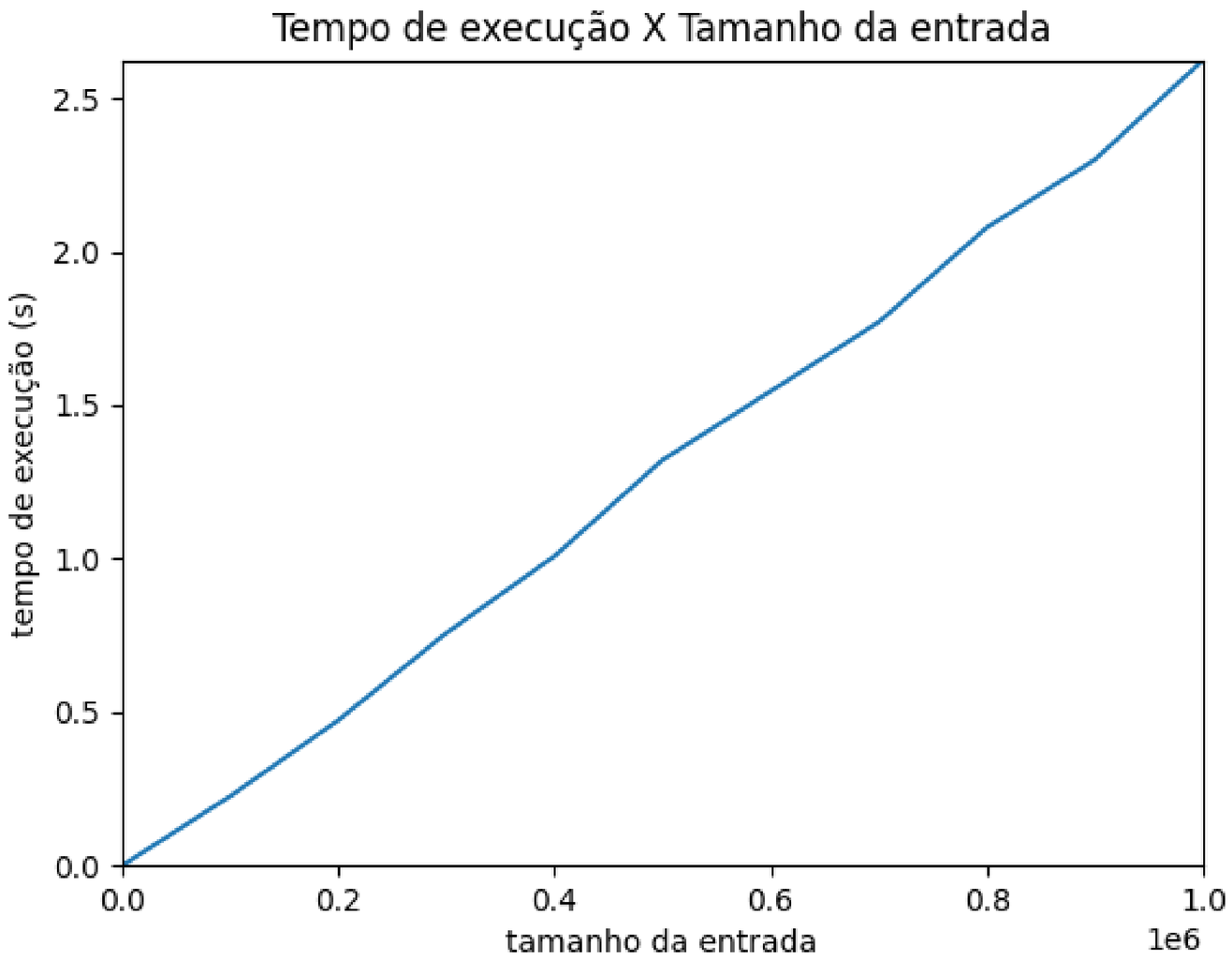
- O algoritmo Shell sort é um método de ordenação que melhora o desempenho do algoritmo de inserção (insertion sort) ao dividir o conjunto de elementos em sublistas menores. Essas sublistas são ordenadas independentemente e, em seguida, combinadas gradualmente até que todo o conjunto esteja ordenado.

SHELL SORT

- Pior caso sendo:
  - O(n log n)
- Caso médio:
  - Depende do tamanho do Gap
  - O(n^(3/2)) com gap de Knuth (gap = 3gap + 1)
- Melhor caso:
  - O(n log n)

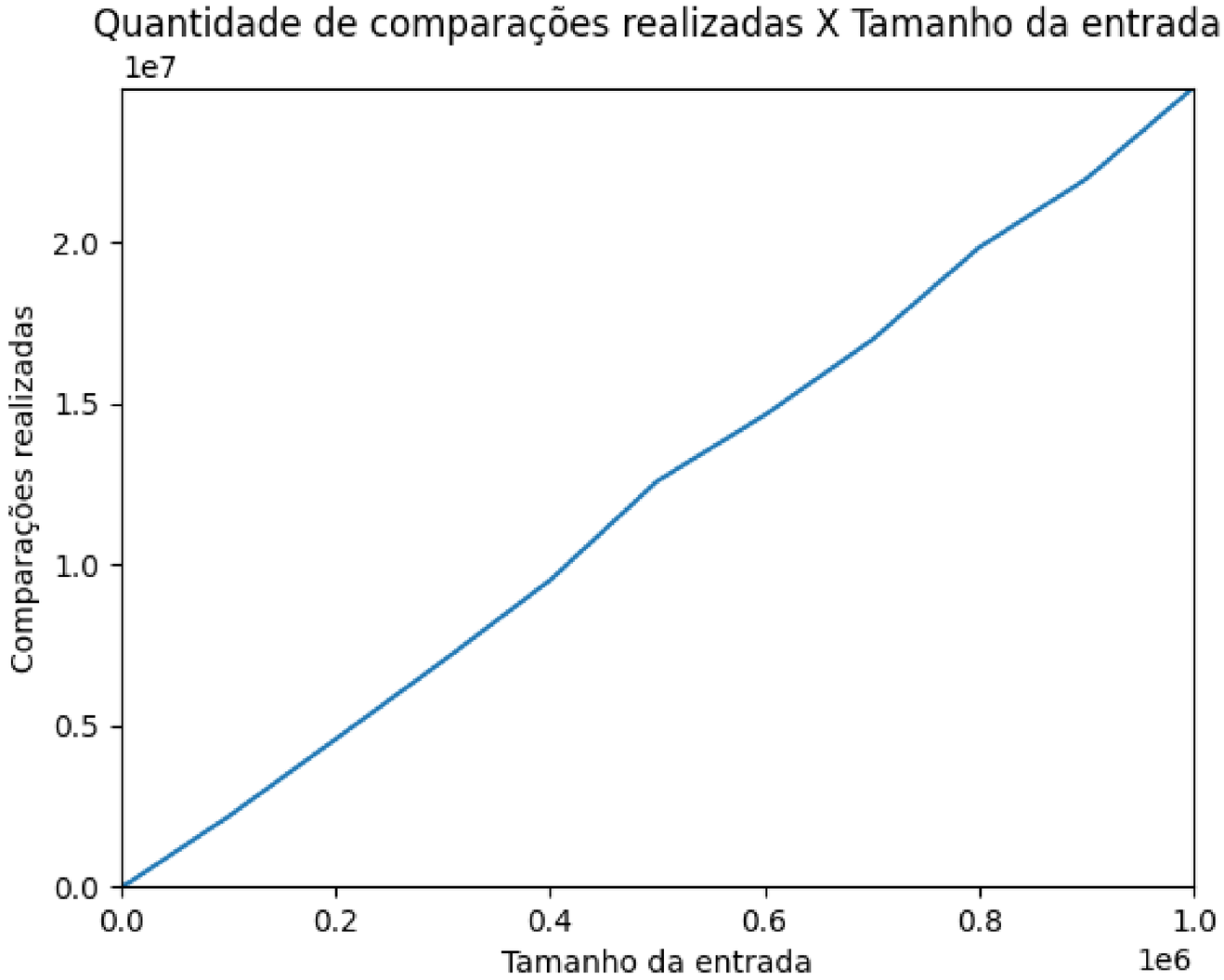
SHELL SORT

- Caso médio



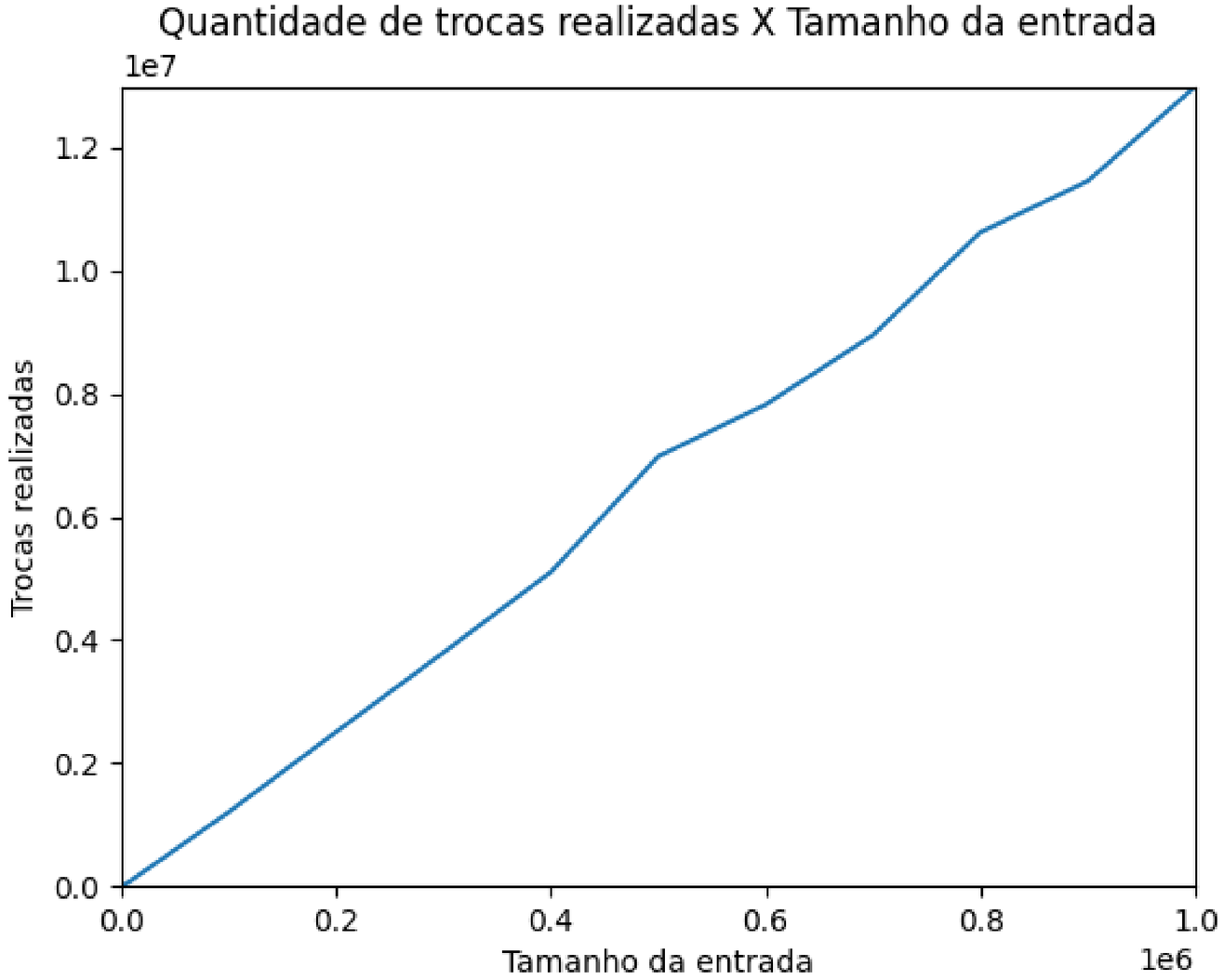
SHELL SORT

- Caso médio



SHELL SORT

- Caso médio



SELECTION SORT

- O Selection Sort é um algoritmo de ordenação que divide a lista em duas partes: uma parte ordenada e uma parte não ordenada. Ele encontra o menor elemento na parte não ordenada e o coloca no final da parte ordenada, repetindo esse processo até que toda a lista esteja ordenada.

QUICK SORT

- O Quick Sort é um algoritmo de ordenação eficiente e amplamente utilizado, baseado no princípio de divisão e conquista. Ele seleciona um elemento da lista, chamado de "pivô", e rearranja os elementos de forma que os elementos menores que o pivô fiquem à sua esquerda, e os elementos maiores à sua direita. Em seguida, o processo é repetido recursivamente para as sublistas à esquerda e à direita do pivô.

GNOME SORT

- Gnome Sort, também conhecido como Stupid Sort, é um algoritmo de ordenação simples que funciona de maneira semelhante ao Bubble Sort. Ele percorre a lista comparando elementos adjacentes e os troca se estiverem fora de ordem, mas com uma diferença crucial: em vez de percorrer a lista sequencialmente, ele pode retroceder várias posições se houver uma troca.

HEAP SORT

- Heap Sort é um algoritmo de ordenação que utiliza a estrutura de dados chamada heap (ou árvore binária de heap) para organizar os elementos. Ele constrói uma heap máxima (ou mínima, dependendo da ordem desejada) e, em seguida, extrai o elemento de topo repetidamente para obter a lista ordenada.

Name	Best	Average	Worst	Memory	Stable	Method
Quicksort	$n \log n$	$n \log n$	n^2	$\log n$	No	Partitioning
Merge sort	$n \log n$	$n \log n$	$n \log n$	n	Yes	Merging
In-place merge sort	—	—	$n \log^2 n$	1	Yes	Merging
Introsort	$n \log n$	$n \log n$	$n \log n$	$\log n$	No	Partitioning & Selection
Heapsort	$n \log n$	$n \log n$	$n \log n$	1	No	Selection
Insertion sort	n	n^2	n^2	1	Yes	Insertion
Block sort	n	$n \log n$	$n \log n$	1	Yes	Insertion & Merging
Timsort	n	$n \log n$	$n \log n$	n	Yes	Insertion & Merging
Selection sort	n^2	n^2	n^2	1	No	Selection
Cubesort	n	$n \log n$	$n \log n$	n	Yes	Insertion
Shellsort	$n \log n$	$n^{4/3}$	$n^{3/2}$	1	No	Insertion
Bubble sort	n	n^2	n^2	1	Yes	Exchanging
Exchange sort	n^2	n^2	n^2	1	No	Exchanging
Tree sort	$n \log n$	$n \log n$	$n \log n$ (balanced)	n	Yes	Insertion
Cycle sort	n^2	n^2	n^2	1	No	Selection
Library sort	$n \log n$	$n \log n$	n^2	n	No	Insertion
Patience sorting	n	$n \log n$	$n \log n$	n	No	Insertion & Selection
Smoothsort	n	$n \log n$	$n \log n$	1	No	Selection
Strand sort	n	n^2	n^2	n	Yes	Selection
Tournament sort	$n \log n$	$n \log n$	$n \log n$	n[11]	No	Selection
Cocktail shaker sort	n	n^2	n^2	1	Yes	Exchanging
Comb sort	$n \log n$	n^2	n^2	1	No	Exchanging
Gnome sort	n	n^2	n^2	1	Yes	Exchanging
Odd–even sort	n	n^2	n^2	1	Yes	Exchanging