Análise de Algoritmos
Projeto 1: Análise da Busca Binária

Jacksson Yuri de Amorim Wasterloô[1]

[1]Instituto de Federal de Brasília (IFB)
Taguatinga – DF – Brasil

[2]Bacharel em Ciência da Computação

jackyuri6@gmail.com

Abstract. *This article aims to introduce search algorithms such as binary search, sequential search and optimized sequential search and comparing them in cases that have been implemented and showing how each algorithm performed for each case. After this, an asymptotic analysis will also be carried out in notation of the* Θ*.*

Resumo. *Esse artigo tem como objetivo a introdução de algoritmos de busca como busca binária, busca sequencial e busca sequencial otimizado-os e comparando-os em casos que foram implementados e mostrando como cada algoritmo se saiu para cada caso. Após isso também será feito uma análise assintótica em notação do* Θ*.*

1. Introdução

A eficiência na manipulação de grandes volumes de dados é um dos desafios fundamentais da computação moderna. Um dos problemas mais clássicos nesse contexto é a busca por elementos em conjuntos de dados, onde a rapidez na localização de uma **chave** é essencial para diversos aplicativos. Para abordar esse problema, uma variedade de algoritmos de busca foram desenvolvidos, cada um com suas próprias características e desempenho.

Neste trabalho, exploramos três desses algoritmos: busca sequencial, busca binária e busca sequencial otimizada ambos os 3 algoritmos tentam minimizar o tempo para encontrar algum número aleatório. O objetivo do trabalho é apresentar esses algoritmos, mostrar implementação deles, em quais pontos são bons ou não tão bons e mostrar a eficácia deles para certos casos.

2. Conceitos Preliminares

Para começar a mostrar a implementações dos algoritmos e seu funcionamento é importante entender melhor como funciona cada um.

2.1. Busca Sequencial

A busca sequencial é o método mais simples de busca em um conjunto de dados. A ideia da busca sequencial é percorrer o vetor ordenado da esquerda pra direita, verificando um a um se a chave foi encontrada. A busca sequencial, no pior dos casos finaliza em O(n), ou seja, se o vetor tem n elementos e o elemento não existe no vetor, por conta de como ela é implementada a busca sequencial vari verificar todos os elementos até o último para então mostrar que o elemento não foi encontrado, o mesmo ocorre quando o elemento a ser encontrado é o último da lista.

2.2. Busca Sequencial Otimizada

A busca sequencial otimizada é uma variação do algoritmo de busca sequencial que aproveita a ordenação parcial do conjunto de dados. Antes de iniciar a busca, o conjunto é pré-processado para garantir que os elementos estejam parcialmente ordenados. Isso permite que a busca sequencial otimizada interrompa o processo assim que encontrar um elemento maior que o procurado, reduzindo o número de comparações necessárias. Embora ainda tenha complexidade temporal linear no pior caso, a busca sequencial otimizada pode ser mais eficiente em conjuntos parcialmente ordenados.

2.3. Busca Binária

A busca binária é um algoritmo que implementa o paradigma de divisão e consquista, ela requer que uma lista esteja ordenada. A ideia da busca binária é comparar o valor que quero encontrar com o valor do meio da lista, caso não seja, o algoritmo verifica se o número que eu quero é maior ou menor que o elemento do meio da lista, caso seja menor a busca binária cria uma sub-lista que seriam os números inferiores ao elemento do meio e então a busca é aplicada nesta lista de inferiores, o mesmo é feito com os elementos que são superiores ao elemento do meio, a busca binária então vai criando várias sub-listas tornando o algoritmo mais rápido para encontrar o alvo que eu quero. A busca binária usa o conceito de logaritmo na base 2(log2), a potência encontrada perto do valor que eu quero encontrar seria o total de etapas até chegar nele, por exemplo, se eu quero encontrar o número 1000 dentro de uma lista com 1000 elementos, pra isso precisamos encontrar o log da base 2 que devido a uma potência x o resultado é igual ou superior a 1000, neste caso especificamente é 10, pos 2**10 é 1024, número próximo a 1000, o resultado de etapas para essa busca binária seria 10.

3. Análise Empirica

3.1. Metodologia dos Experimentos

Para conduzir os experimentos, foram implementados os algoritmos de busca sequencial, busca sequencial otimizada e busca binária em Python 3. Foram utilizados conjuntos de dados de volumes variando de 10^4 a 10^7 para o tamanho da lista e de 10^2 a 10^5 para elementos a serem buscados.

As consultas foram geradas aleatoriamente dentro do intervalo de valores presente nos conjuntos de dados. Utilizamos a função random.randint() do módulo random do Python para gerar os números aleatórios de consulta.

3.2. Especificações do computador

Processador: Intel Core i5-5200U 2.20GHz

Memória RAM: 10GB

Sistema Operacional: Windows 10

Linguagem de Programação: Python 3.8

4. Análise de Algoritmo

4.1. Análise de Complexidade

4.1.1. Busca Sequencial

Complexidade Temporal: O(1) é o melhor caso para a busca binária, isso é quando é o primeiro elemento da lista. O pior caso da busca sequencial é O(n), ou seja, quandoo elemento está no final da lista ou não está na lista, dessas duas maneiras a busca sequencial vai precisar percorrer a lista toda diminuindo a efiácia dela e aumentando o tempo da busca.

4.1.2. Busca Sequencial Otimizada

A busca sequencial otimizada nesse quesito é semelhante à busca sequencial, ou seja, no pior caso temos O(n), mas pode ser mais eficiente em certos casos devido a otimização dado dentro do algoritmo. E no melhor dos casos temos O(1) como a busca sequencial.

4.1.3. Busca Binária

O(log n) é o pior caso da busca binária pois é preciso vários e vários cortes e criação de sub-listas para se encontrar o elemento buscado. Já o O(1) é o melhor caso para a busca binária pois ela só precisa fazer uma única divisão e o elemento é encontrado, como por exemplo, o número buscado é o elemento do meio da lista.

4.2. Comparações entre as buscas

4.2.1. Busca Sequencial vs. Busca Sequencial Otimizada

A busca sequencial otimizada demonstra ser superior em cenários parcialmente ordenados, onde a otimização na verificação do elemento pode reduzir o tempo de busca. No entanto, em conjuntos de dados totalmente desordenados, ambas as buscas sequenciais têm desempenho semelhante.

4.2.2. Busca Sequencial vs. Busca Binária

A busca binária é melhor, devido a sua complexidade logaritmica, quando se tem grandes volumes de dados, um exemplo comentado anteriormente, é que caso eu queirar achar o número 1000 em uma lista com 1000 elementos, para a busca binária eu levaria apenas 10 passos, já para a busca sequencial(supondo nesse caso que a lista está ordenada) então esse seria o pior caso pra ela pois o elemento se encontra na última posição e ela levaria 1000 passos, resumidamente a busca binária leva a melhor em operações muito grandes. Para o caso de um conjunto menor como uma lista de 10, a depender do caso, a busca sequencial se torna melhor ou equivalente a busca binária.

4.2.3. Busca Sequencial Otimizada vs. Busca Binária

A escolha entre busca sequencial otimizada e busca binária depende do grau de ordenação do conjunto de dados. Em conjuntos de dados parcialmente ordenados, a busca sequencial otimizada pode ter desempenho comparável ou até mesmo superior à busca binária. No entanto, em conjuntos de dados estritamente ordenados, a busca binária é a escolha preferida devido à sua complexidade logarítmica.

5. Discussão

Como resultado foram gerados gráficos apresentando o tempo de cada busca em diversos casos, como comentado anteriormente na seção "Análise Empírica", foram utilizados conjuntos de dados de volumes variando de 10^4 a 10^7 para o tamanho da lista e de 10^2 a 10^5 para elementos a serem buscados. A seguir estão os gráficos(os gráficos ficaram após as referências).

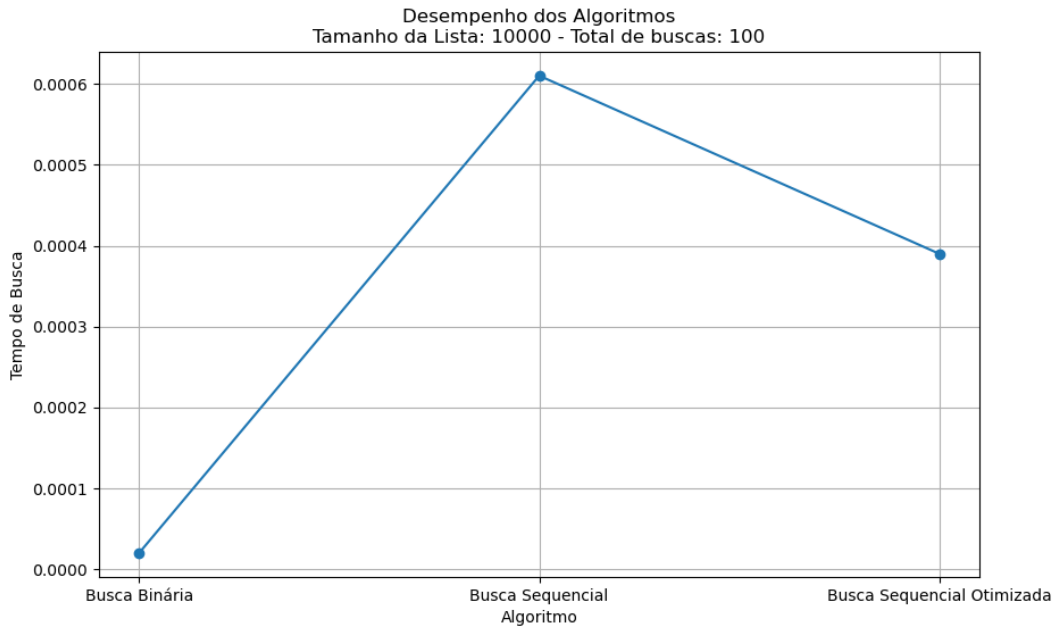

Figura 1. Usando $n = 10^4$ e $q = 10^2$

6. Considerações Finais

Em todos os casos testados a busca binária ficou com melhor desempenho que a busca sequencial e a busca sequencial otimizada(linear), o que confirma que em grandes massas de dados e ordenados a busca binária tende a prevalecer, em outra partida, a busca sequencial otimizada tende a ser igual ou melhor que a busca sequencial comum que não aparentou ser tão boa em relação desempenho-rapidez nestes casos de testes.

7. References

PANTUZA. Busca Binária. [S.l.], 2022. Disponível em: `https://blog.pantuza.com/artigos/busca-binaria`. Acesso em: 21 mar. 2024.

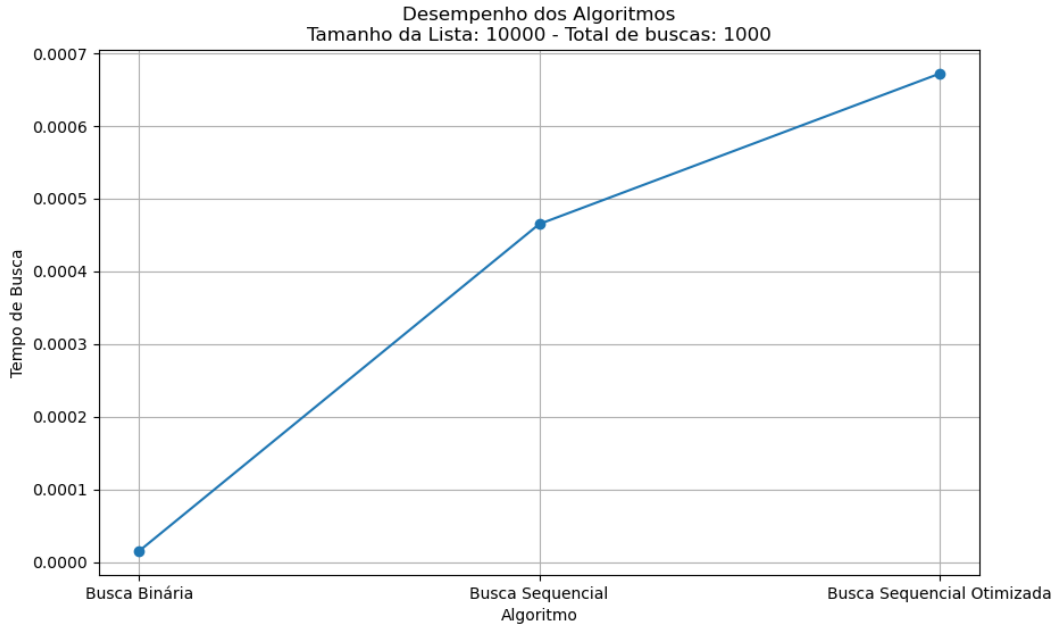

Figura 2. Usando $n = 10^4$ e $q = 10^3$

KHAN ACADEMY. Busca Binária. [S.l.], 2022. Disponível em: https://pt.khanacademy.org/computing/computer-science/algorithms/binary-search/a/binary-search. Acesso em: 21 mar. 2024.

DIO.ME. O poder da busca binária. [S.l.], 2022. Disponível em: https://www.dio.me/articles/o- poder-da-busca-binaria. Acesso em: 21 mar. 2024.

UNICAMP. Aula 11: Algoritmos de Busca. [S.l.], 2022. Disponível em: https://ic.unicamp.br/~mc102/aulas/aula11.pdf. Acesso em: 21 mar. 2024.

FREECODECAMP. Uma introdução à complexidade temporal dos algoritmos. [S.l.], 2022. Disponível em: https://www.freecodecamp.org/portuguese/news/uma-introducao-a-complexidade-temporal-dos-algoritmos/. Acesso em: 21 mar. 2024.

Percival Lucena, "Busca Linear em Listas — Como encontrar Elemento em uma Lista com Python — Estruturas de Dados 2 ", Vídeo, 2018, https://www.youtube.com/watch?v=MLWEFgjqHrg&t=441s. Acesso em: 22 mar. 2024.

Percival Lucena, "COMPLEXIDADE de ALGORITMOS I - Noção INTUITIVA ", Vídeo, Código Simples, 2018, https://www.youtube.com/watch?v=KVlGx-9CuO4&t=16s. Acesso em: 22 mar. 2024.

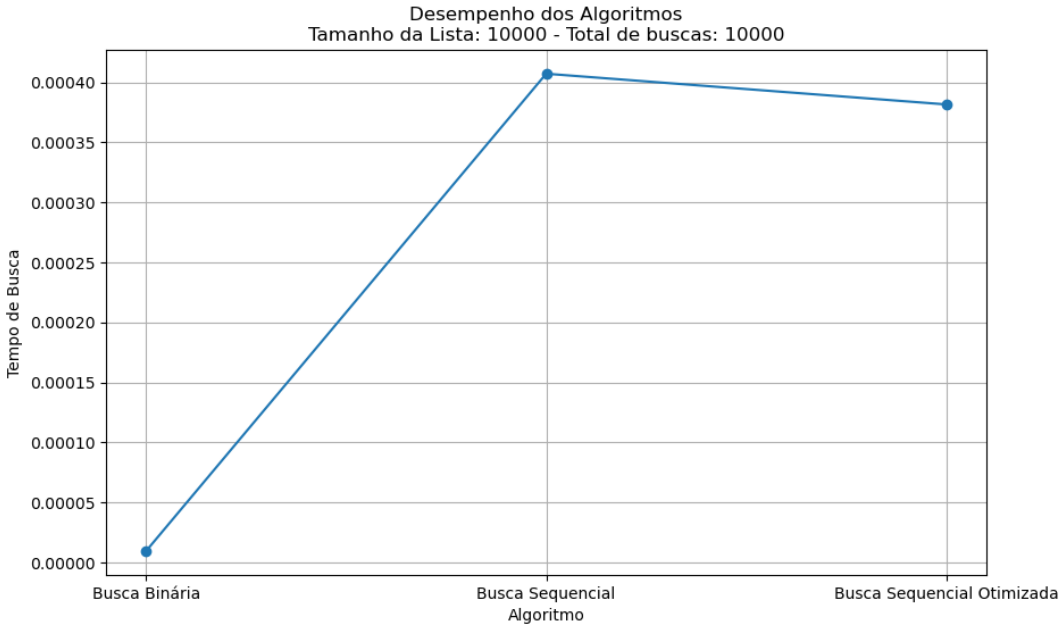

Figura 3. Usando $n = 10^4$ e $q = 10^4$

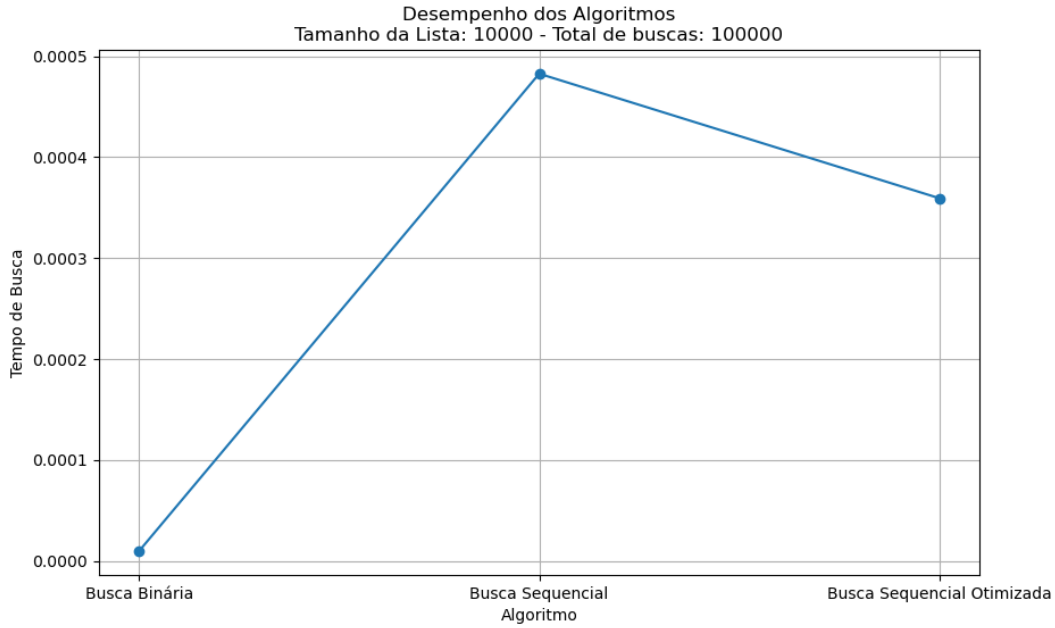

Figura 4. Usando $n = 10^4$ e $q = 10^5$

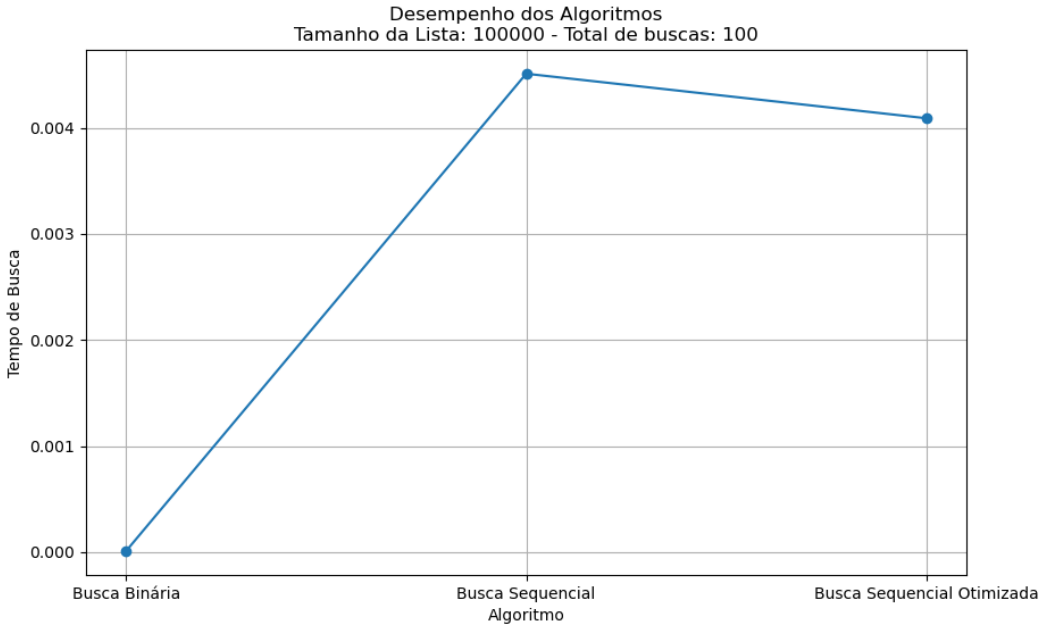

Figura 5. Usando $n = 10^5$ e $q = 10^2$

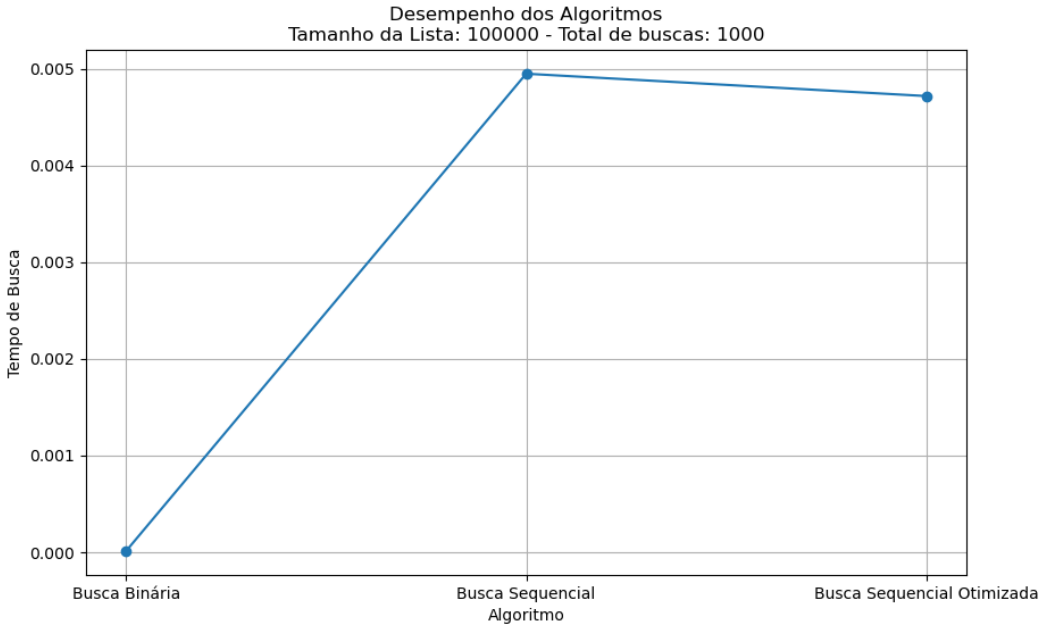

Figura 6. Usando $n = 10^5$ e $q = 10^3$

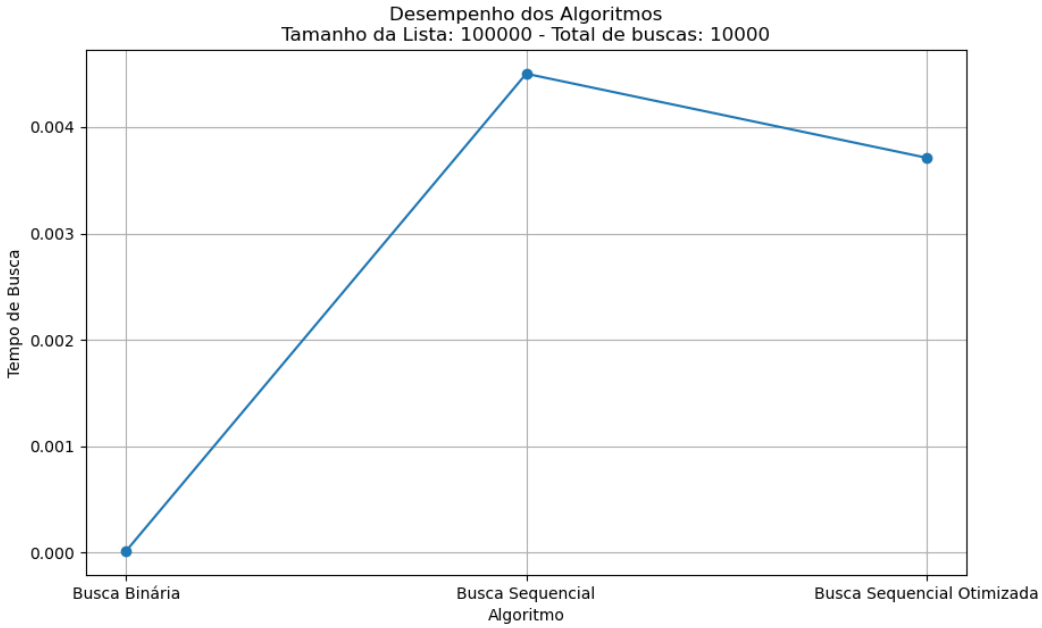

Figura 7. Usando $n = 10^5$ e $q = 10^4$

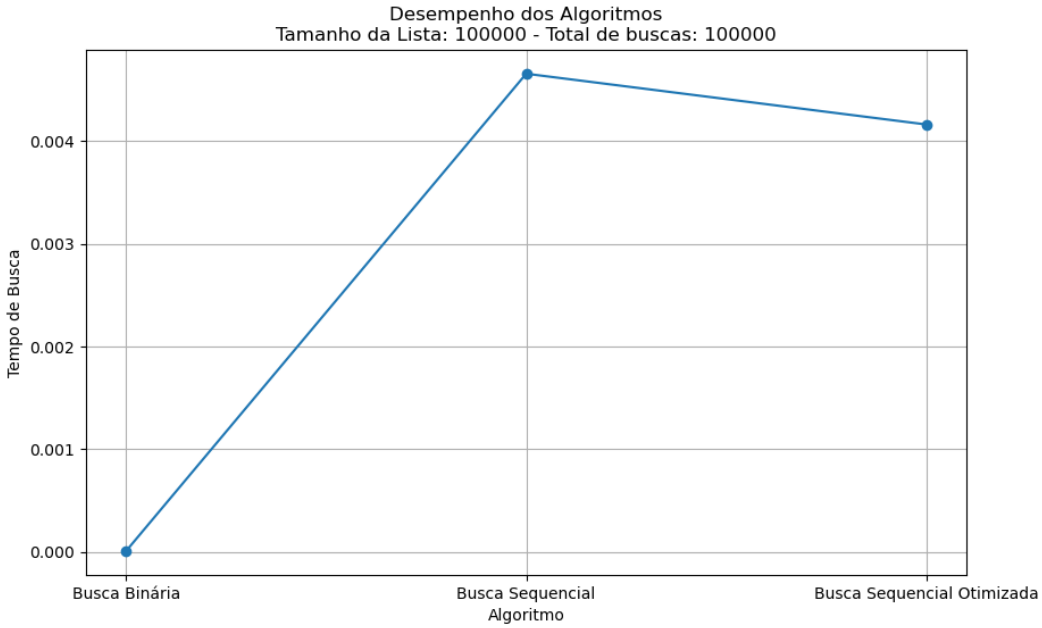

Figura 8. Usando $n = 10^5$ e $q = 10^5$

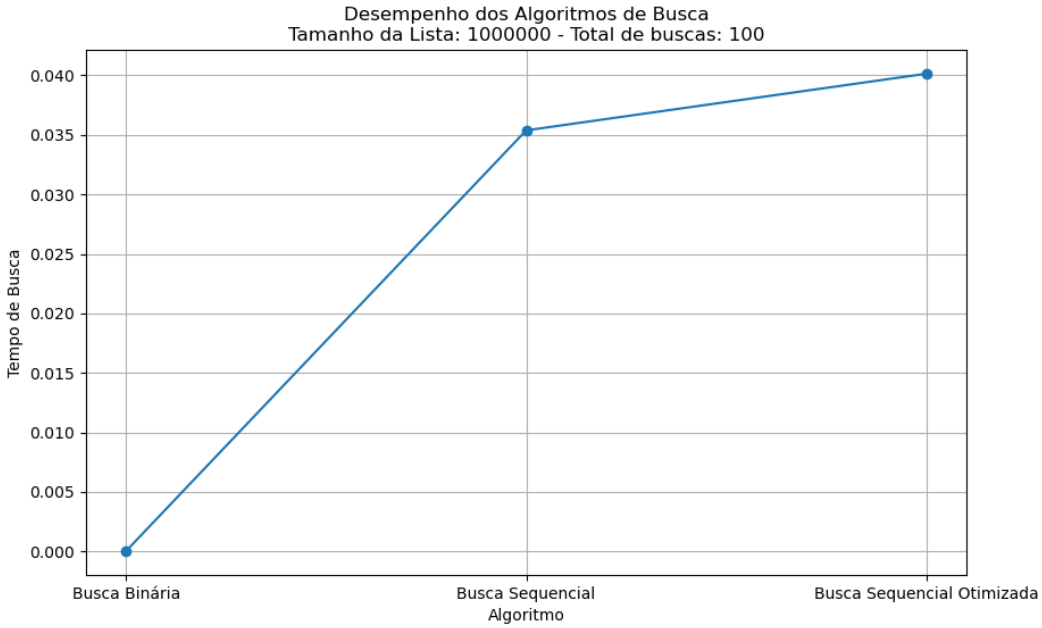

Figura 9. Usando $n = 10^6$ e $q = 10^2$

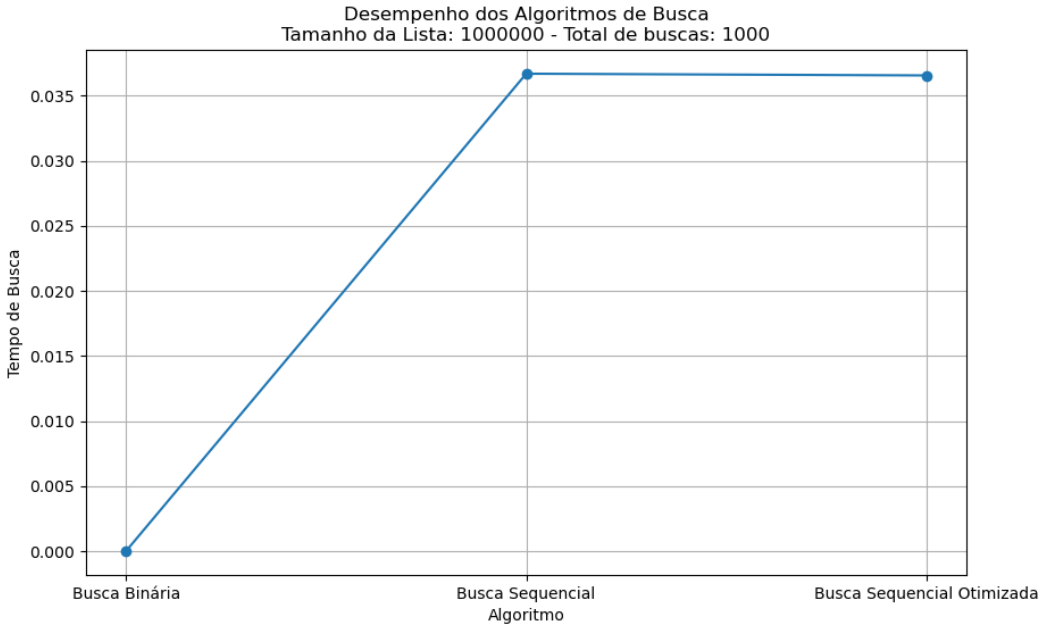

Figura 10. Usando $n = 10^6$ e $q = 10^3$

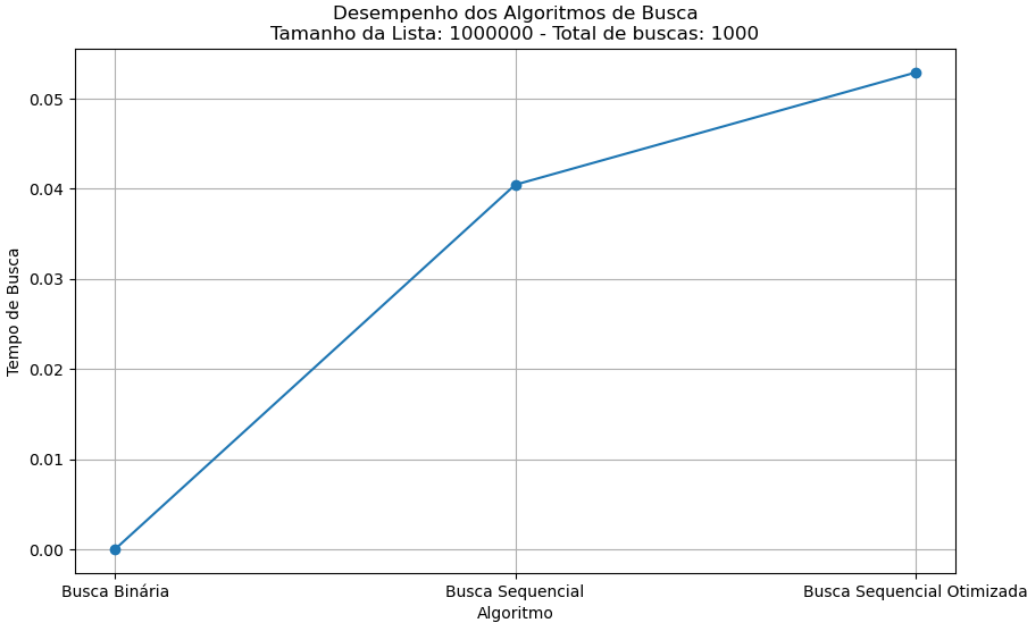

Figura 11. Usando $n = 10^6$ e $q = 10^4$