UNIVERSIDADE FEDERAL DO CEARÁ – CAMPUS SOBRAL

CURSO DE ENGENHARIA DA COMPUTAÇÃO

DISCIPLINA: TÓPICOS ESPECIAIS EM COMPUTAÇÃO II

PROFESSOR: FISHER JONATAS FERREIRA

ESQUENTA 2

Francisco Henrique Rocha Silva

Matrícula: 413563

Sobral-CE

2023.2

SUMÁRIO

.

1.Introdução

Esse trabalho apresenta a análise de dois algoritmos, que visam encontrar o maior valor em um vetor, a partir da amostragem do tempo de execução e consumo de memória. Com base nos dados coletados foi realizado a construção de uma tabela com as informações solicitadas pelo trabalho, como mostra a tabela 01, partir da qual será construída os gráficos de consumo de memória e tempo de execução.[

Tabela 01

	Algoritmo 01		Algoritmo 02	
Matrizes	Tempo de execução(ms)	Memória gasta(Kb)	Tempo de execução(ms)	Memória gasta(Kb)
100	0,17	16,38	0,19	10,33
200	0,19	20,48	0,28	16,38
1000	0,24	49,15	0,31	45,06
2000	0,31	77,82	0,36	69,63
5000	0,52	163,84	0,44	118,78
10000	4,68	1.908,74	1,53	1.163,26
50000	9,11	2.142,21	3,75	1.576,96
100000	47,50	7.682,94	6,12	3.506,18
500000	60,39	8.186,91	32,46	8.536,06
1000000	70,12	8.246,94	54,80	8.617,98
5000000	309,02	11.321,34	206,49	11.612,16
10000000	1.494,00	15.833,09	768,89	14.700,26
100000000	2.858,00	48.883,71	1.612,00	27.402,24

2.Algoritmo 01: “ Maxval1”

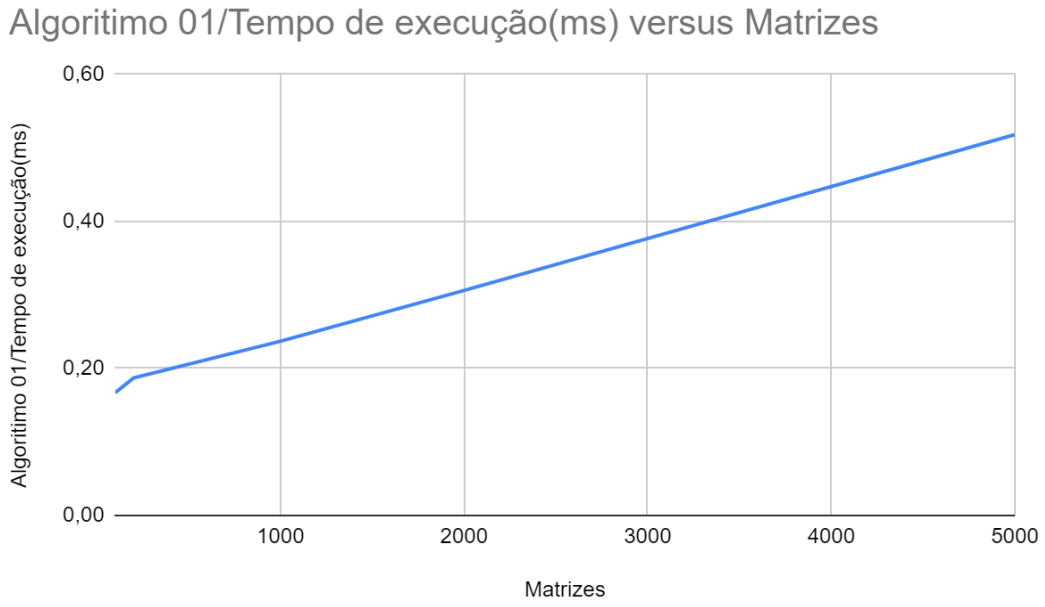
O algoritmo 01, anexo A, possui complexidade de tempo O(n), onde "n" é o número de elementos no vetor "A" que é percorrido uma única vez para encontrar o valor máximo. Logo o tempo de execução depende do tamanho do vetor “A” se comportando como uma função linear, no entanto, utilizando a amostragem de 13 matrizes com número de elementos variando de 100 á 100000000 percebe-se, na prática, que o real comportamento se diverge com base no tamanho da matriz, como mostra o gráfico 01.

Gráfico 01

Fonte: Autoria própria.

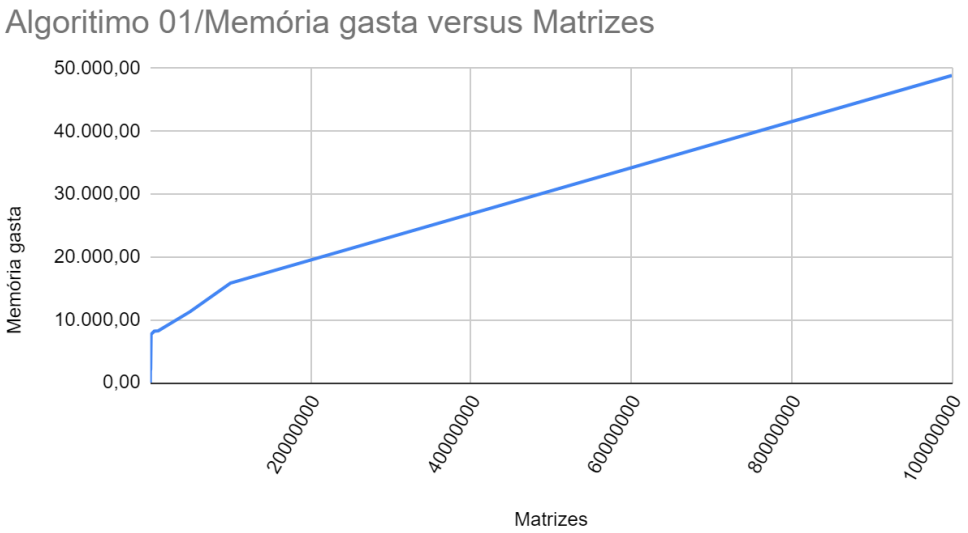
Ao realizar uma observação detalhada percebe-se que a partir da matriz 10.000.000 o gráfico se comporta de forma linear, mas para matrizes menores o mesmo varia muito, assim como é apresentado pelo gráfico 02.

Gráfico 02



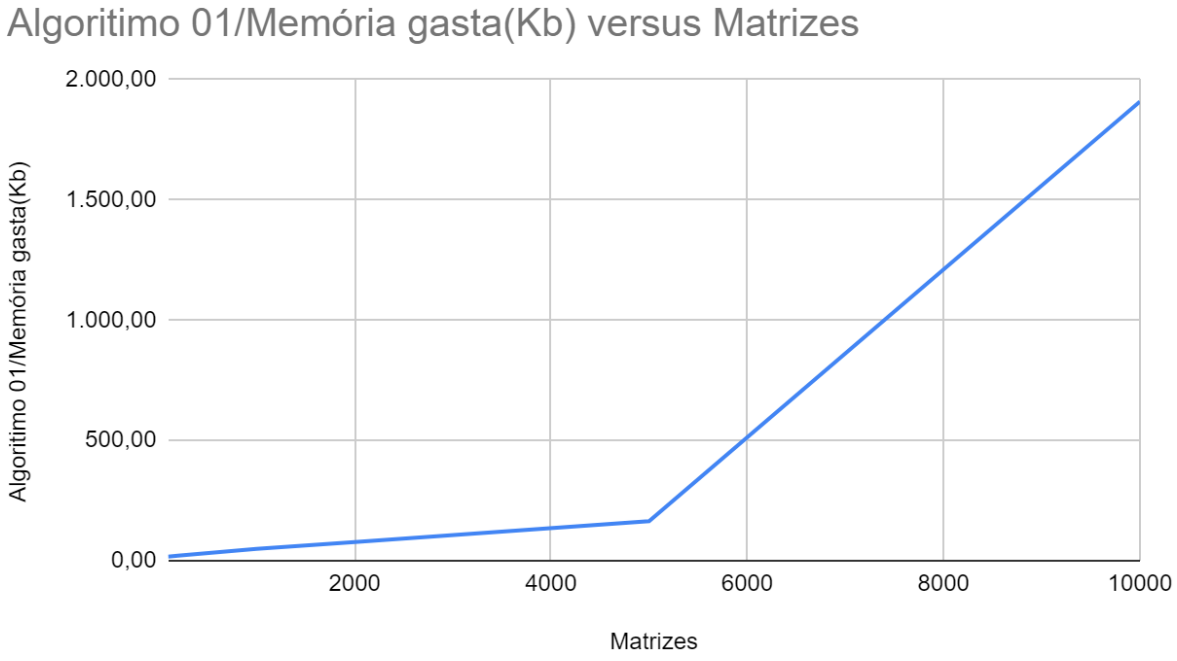
Analisando o consumo de memória percebe-se o mesmo comportamento para matrizes grandes, com tamanho de 10.000.000, como para matrizes pequenas, que é comprovado pelos gráficos 03 e 04.

Gráfico 03.



Fonte: Autoria própria.

Gráfico 04

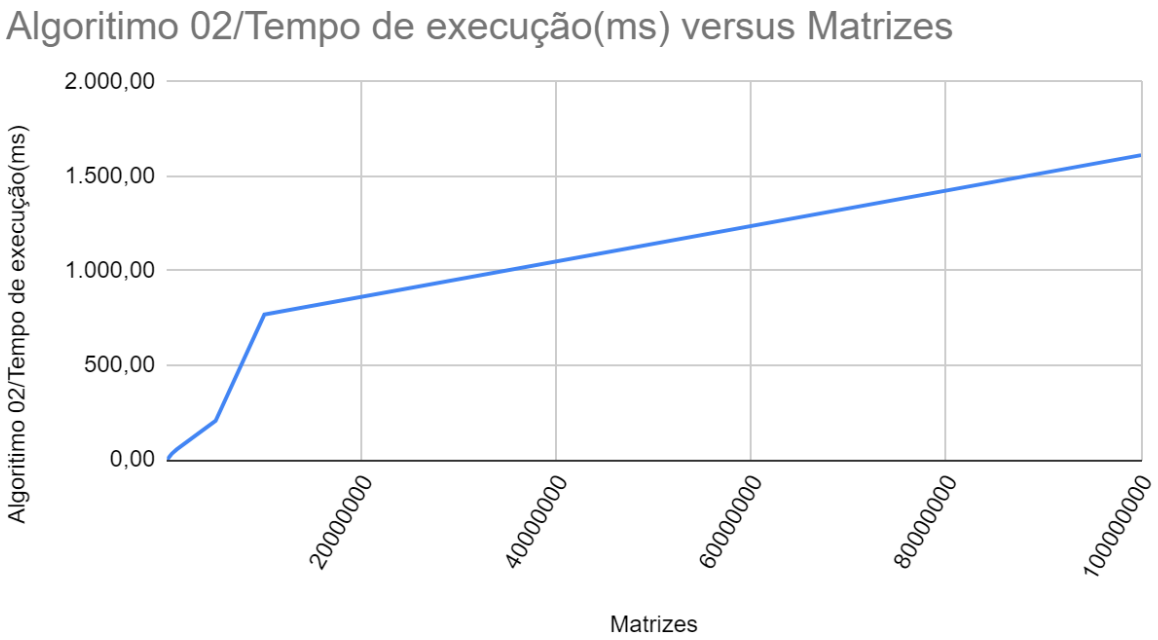


3.Algoritmo 02: “ Maxval2”

É um algoritmo de divisão e conquista com complexidade O(log n). Ele é implementado de forma recursiva, o que poderia levar a um alto consumo de memória devido ao uso da pilha de chamadas. No entanto, como a recursão neste algoritmo é simples e a linguagem JavaScript possui uma otimização eficaz para esse tipo de problema, os resultados de consumo de memória não mostraram grande discrepância em comparação com o tamanho da matriz na qual foi aplicado.

Ao realizar uma análise dos gráficos do tempo de execução e consumo de memória observa-se o mesmo comportamento do algoritmo 01 mas com apenas duas diferenças. A primeira é o intervalo de tempo de execução para matrizes maiores que 5000 bem menor do que comparado ao algoritmo 01,como mostra os gráficos 05.

Gráfico 05



Para matrizes pequenas foi observado que o algoritmo 01 possui um tempo de execução melhor do que o algoritmo 02, como mostra a tabela 02.

Tabela 02

Matrizes	Tempo de execução(ms)	Memória gasta(Kb)	Tempo de execução(ms)	Memória gasta(Kb)
100	0,17	16,38	0,19	10,33
200	0,19	20,48	0,28	16,38
1000	0,24	49,15	0,31	45,06
2000	0,31	77,82	0,36	69,63
5000	0,52	163,84	0,44	118,78