UNIVERSIDADE ESTADUAL DE SANTA CRUZ
DEPARTAMENTO DE CIÊNCIAS E TECNOLOGIAS
CET 1216 – PROGRAMAÇÃO PARALELA

COMPARAÇÃO ENTRE A PERFORMANCE DOS ALGORITMOS

Metodologia: Para obter-se uma comparação entre o desempenho os algoritmos desenvolvidos, executou-se cada algoritmo uma vez para matrizes de diferentes ordens. As ordens escolhidas foram sempre potências de 2, iniciando em 2 e terminando em 2048. Com a finalidade de obter-se uma comparação entre a variação entre diferentes linguagens, criou-se códigos equivalentes em Python e em C. Para manter a coerência das medidas, executou-se os códigos em um mesmo computador, e sem outros aplicativos abertos no momento do teste.

Resultados: A tabela e os gráficos comparativos abaixo representam os resultados obtidos a partir da execução dos códigos elaborados em Python e em C. A métrica utilizada para a comparação foi o tempo, medido em segundos utilizando-se tomadas de tempo no início e no final do cálculo, com as funções *datetime.datetime.now()*, em Python, e gettimeofday(), em C.

Código C				Código Python			
Comum		Strassen		Comum		Strassen	
n x n	segundos	n x n	segundos	n x n	segundos	n x n	segundos
2	0.0000	2	0.0000	2	0.0000	2	0.0139
4	0.0000	4	0.0000	4	0.0000	4	0.0109
8	0.0000	8	0.0000	8	0.0000	8	0
16	0.0000	16	0.0000	16	0.0029	16	0.0019
32	0.0000	32	0.0010	32	0.0169	32	0.0140
64	0.0040	64	0.0011	64	0.1309	64	0.1159
128	0.0053	128	0.0090	128	1.0316	128	0.9127
256	0.0450	256	0.0635	256	8.3093	256	7.2047
512	0.4197	512	0.5925	512	69.0847	512	58.0563
1024	7.1720	1024	5.6380	1024	560.2076	1024	480.3543
2048	74.7152	2048	70.5769	2048	4710.4597	2048	3876.8639

Tabela 1: Rendimento dos algoritmos para diferentes tamanhos de matrizes

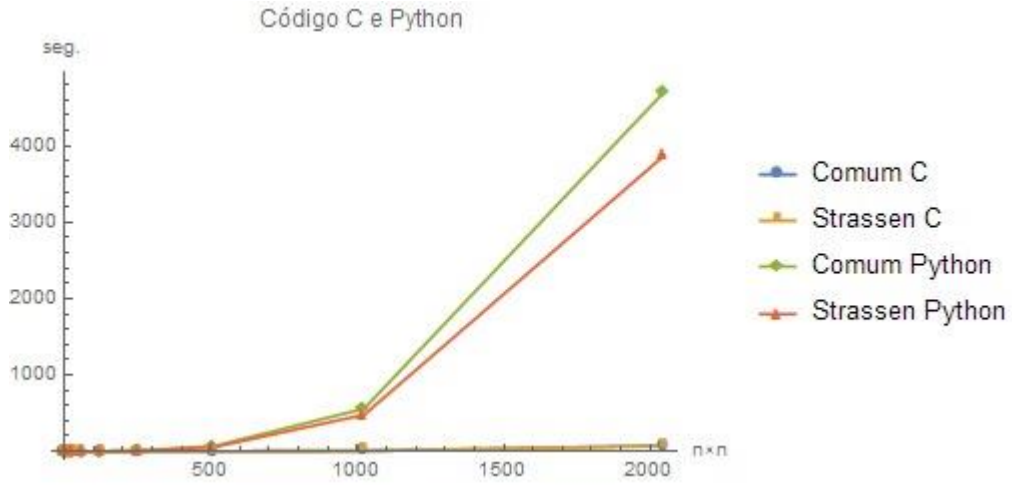

Gráfico 1: Comparação dos resultados obtidos

Relatório desenvuelto por: Frederico, Lucas, Oniram e Pedro
Data: 05/09/2018

Discussão: À primeira vista, é bastante destacada a diferença em rendimento entre os algoritmos equivalentes em Python e em C, o que nos permite concluir que os códigos desenvolvidos em Python foram menos eficientes que os desenvolvidos em C.

No código em C, observou-se uma melhora de aproximadamente 6% no desempenho para matrizes de ordem 2048, e de aproximadamente 2% pra matrizes de ordem 1024. Por outro lado, observou-se que para matrizes menores, o algoritmo de Strassen apresenta rendimento inferior, chegando a ser quase 70% mais lento no caso de matrizes de ordem 128.

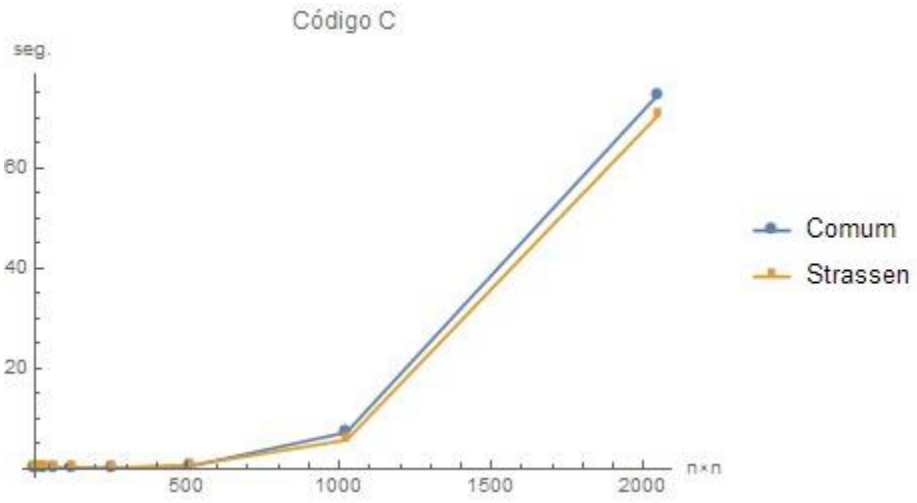

Gráfico 2: Rendimentos dos algoritmos de multiplicação desenvolvidos em C

Já no código em Python, o algoritmo de Strassen apresenta rendimento superior para todas as matrizes com ordem maior ou igual a 16, tendo uma média geral de cerca de 17% de melhora no desempenho em relação ao algoritmo convencional.

Com essas informações em mente, percebe-se que quanto mais computacionalmente custoso o código, maior é a melhora obtida pela substituição do algoritmo tradicional pelo algoritmo de Strassen nas operações de multiplicação de matrizes.

or fim, percebe-se que mesmo alcançando uma melhora considerável utilizando o código em Python, a melhora não alcançou a melhora teórica média, que seria de aproximadamente 41% (utilizando-se n^3 para o algoritmo normal e $n^{2,86}$ para o de Strassen) para o mesmo intervalo de testes. Isso se deve principalmente ao fato de que o cálculo realizado leva em consideração somente as operações de multiplicação, descartando as somas, subtrações, atribuições, etc.

Ordem	2	4	8	16	32	64	128	256	512	1024	2048	Média
Normal	8,0E+00	6,4E+01	5,1E+02	4,1E+03	3,3E+04	2,6E+05	2,1E+06	1,7E+07	1,3E+08	1,1E+09	8,6E+09	-
Strassen	7,3E+00	5,3E+01	3,8E+02	2,8E+03	2,0E+04	1,5E+05	1,1E+06	7,7E+06	5,6E+07	4,1E+08	3,0E+09	-
Melhora	9%	18%	25%	32%	38%	44%	49%	54%	58%	62%	66%	**41%**

Tabela 2: Análise da melhora teórica esperada pela substituição