UFU – Universidade Federal de Uberlândia
Disciplina: Análise de Algoritmos

Aluno: André Luiz Morais Peres de Quinta

1. Descrição do experimento
   1. Qual linguagem de programação foi adotada?
      A linguagem de programação adotada foi: C.

   2. Quantos pares de matrizes foram criados? Quais as dimensões de cada um? (Lembre-se de que vamos considerar apenas matrizes quadradas.)
      Para cada multiplicação matricial foi criada uma nova matriz com valores randômicos no intervalo [0, 9].

      Foram criadas matrizes com as dimensões 2x2, 4x4, 8x8, 16x16, 32x32, 64x64, 256x256, 512x512, 1024x1024, 2048X2048 e 4096x4096, uma vez o algoritmo de Strassen possui melhor performasse para matrizes do tipo $2^n x 2^n$, para outros tipos de matrizes, eventualmente deverá ocorrer um complemento com mais colunas e linha zeradas.

   3. Quantas réplicas de experimentos foram executadas?
      Inicialmente foi executado o algoritmo uma vez para cada dimensões citada acima nos dois algoritmos, para fazer uma observação do tempo estimado do experimento.

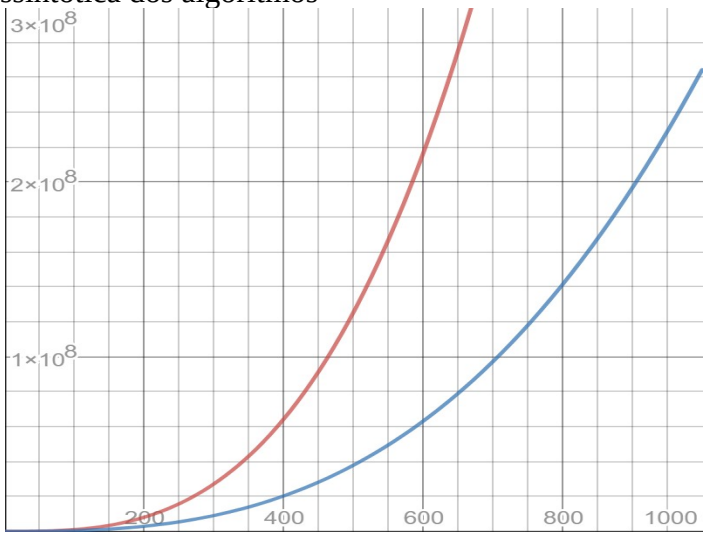
      Para as matrizes com dimensões 2x2 até 1024x1024 foram executadas 30 vezes devido ao seu baixo tempo de processamento.

      Para as matrizes com dimensões 2048X2048 foram executadas apenas 4 vezes (com o intuito de observar o crescimento do tempo de processamento para valores maiores de n), além disso foi testado a multiplicação duas vezes para a matriz de 4096X4096, devido ao alto tempo de processamento,.

2. Gráficos e tabelas sobre o experimento.

Os gráficos e tabela a seguir foram retirados dos resultados obtidos nos de experimentos da implementação dos algoritmos disponível em https://github.com/Adlizm/aa_matrixs.

1. Gráfico análise assintótica dos algoritmos



O gráfico acima mostra as funções da análise assintótica dos dois algoritmos. A curva vermelha representa o algoritmo padrão $O(n^3)$, já a curva azul representa o algoritmo de Strassen $O(n^{\log_2 7})$. Isso nos mostra que a partir de um determinado valor K o algoritmo de Strassen sempre será mais eficiente do que o algoritmo padrão

2. Tabela de Média.

	Média de tempo de processamento em millisegundos	
NxN	Algoritmo Padrão $O(n^3)$	Algoritmo de Strassen $O(n^{Log_2 7})$
´2x2	0	0
4x4	0	0
8x8	0	0
16x16	0	3
32x32	0	19
64x64	4	113
128x128	23	761
256x256	186	5401
512x512	3779	37560
1024x1024	122470	264288
2048x2048	981505	1851860
4096x4096	8102569	12979953

Nessa tabela percebemos uma grande diferença de tempo de processamento entre os algoritmos, o que não estaria de acordo com o gráfico acima, isso pode ser devido

alocação e desalocação das as matrizes auxiliares e as multiplicações pela recurção no Algoritmo de Strassen .

Dito isso, é observado na três últimas dimensões de matrizes testadas que o tempo de processamento do Algoritmo Clássico tem aumentado em um fator de oito vezes da dimensão testada anteriormente, enquanto isso, esse fator no Algoritmo de Strassen é de sete.

Portanto, podemos concluir que embora nos testes realizados o algoritmo de Strassen não foi mais eficiente, esse se tornará mais eficiente para matrizes maiores dos que as testadas nesse trabalho, provavelmente para n > 10.000.

3. Tabela de Desvio Padrão.

	Desvio Padrão de tempo de processamento em milissegundos	
NxN	Algoritmo Padrão $O(n^3)$	Algoritmo de Strassen $O(n^{Log_2 7})$
2x2	0	0
4x4	0	0
8x8	0	0
16x16	0	6
32x32	0	8
64x64	6	10
128x128	6	9
256x256	6	135
512x512	747	131
1024x1024	3932	3530

Obs: Não está mostrando o desvio padrão para N = 2048 e N = 4096, devido ao baixo número de testes que pode obter um resultado tendencioso.

Nessa tabela não percebemos grande diferença entre os desvios padrões dos dois algoritmos. Podemos concluir através dessa tabela a “estabilidade” do algoritmo, isto é, se dado muitas entradas de tamanho NxN se existe muita ou pouca diferença de tempo entre elas.

3. Resultados adicionais.

   Não satisfeito com o resultado obtido pelo Algoritmo de Strassen, propus uma pequena mudança no algoritmo, para matrizes com NxN com N < c (na implementação essa constante é denominada ‘cross’), é realizado o Algoritmo Clássico.

   A imagem abaixo mostra o antes e depois da alteração feita no início da implementação do algoritmo.

```
int multiplyStrassen(Matrix a, Matrix b, Matrix out, int n){
    if(a == NULL || b == NULL || out == NULL) return 0;
    if(n == 1){
        out[0] = a[0]*b[0];
        return 1;
    }
```

```
int multiplyStrassen(Matrix a, Matrix b, Matrix out, int n){
    if(a == NULL || b == NULL || out == NULL) return 0;
    if(n <= cross){
        return multiplyClassic(a,b,out,n);
    }
```

   A tabela abaixo mostra os resultados obtidos pela alteração nas mesmas condições dos testes realizados anteriormente (cross = 64).

	Média de tempo de processamento em millisegundos
NxN	Algoritmo de Strassen $O(n^{Log_2 7})$ com alteração
´2x2	0
4x4	0
8x8	0
16x16	0
32x32	0
64x64	2
128x128	23
256x256	161
512x512	1151
1024x1024	7802
2048x2048	54719
4096x4096	414426

Foi observado uma grande melhore em relação os experimentos realizados anteriormente.