Multiplicação Matrizes - Memória Cache

Felipi Lima Matozinho

Tópicos

- Trechos Código
- Configurações da Máquina
- Metodologia execuções
- Resultados e Comparações
- Considerações Finais

Multiplicação

```cpp
vector<vector<double>>
Matrix::multiply(vector<vector<double>> *matrix1, vector<vector<double>> *matrix2) {
  int r1 = matrix1->size();
  int c1 = (*matrix1)[0].size();
  int c2 = (*matrix2)[0].size();
  int r2 = matrix2->size();

  if (c1 != r2)
    throw std::length_error("Matrixs are not compatible");

  vector<vector<double>> result(r1, vector<double>(c2, 0.0));

  for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c2; j++) {
      result[i][j] = 0;
      for (int k = 0; k < r2; k++) {
        result[i][j] += (*matrix1)[i][k] * (*matrix2)[k][j];
      }
    }
  }

  return result;
}
```

Transposição

```cpp
vector<vector<double>>
OAC::Matrix::transpose(vector<vector<double>> *currentMatrix) {
  int rows = currentMatrix->size();
  int cols = (*currentMatrix)[0].size();

  vector<vector<double>> transpose(cols, vector<double>(rows, 0.0));

  for (int i = 0; i < rows; i++)
    for (int j = 0; j < cols; j++)
      transpose[j][i] = (*currentMatrix)[i][j];

  return transpose;
}
```

Multiplicação Transposta

```
vector<vector<double>>
OAC::Matrix::cacheMultiply(vector<vector<double>> *matrix1,
                           vector<vector<double>> *matrix2,
                           long long *transpositionTime) {
  int r1 = matrix1->size();
  int c1 = (*matrix1)[0].size();
  int c2 = (*matrix2)[0].size();
  int r2 = matrix2->size();
  chrono::system_clock::time_point functionStart;
  common_type_t<chrono::duration<long int, ratio<1, 1000000000>>>
      executionTimePoint;

  if (c1 != r2)
    throw std::length_error("Matrixs are not compatible");
```

Multiplicação Transposta

```
  vector<vector<double>> result(r1, std::vector<double>(c2, 0.0));

  functionStart = chrono::high_resolution_clock::now();
  vector<vector<double>> matrix2Transposed =
      OAC::Matrix::transpose(matrix2);
  executionTimePoint = chrono::high_resolution_clock::now() - functionStart;
  *transpositionTime =
      chrono::duration_cast<chrono::milliseconds>(executionTimePoint).count();

  for (int i = 0; i < r1; i++) {
    for (int j = 0; j < c2; j++) {
      result[i][j] = 0;
      for (int k = 0; k < r2; k++) {
        result[i][j] += (*matrix1)[i][k] * matrix2Transposed[j][k];
      }
    }
  }

  return result;
}
```

Configurações da Máquina

- Processador: Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz
- Caches (soma de todos):
  - L1d: 128 KiB (4 instâncias)
  - L1i: 128 KiB (4 instâncias)
  - L2: 1 MiB (4 instâncias)
  - L3: 6 MiB (1 instância)
- Memória: 8 GB, DDR4, 2400 MT/s
- Disco: SSD 120 GB Kingston

Execuções

- Execuções com tamanhos de 200 até 2000 com passos de tamanho 200
  - Shell script
  - Dump dados em arquivos “.txt”
  - Tempo total para modo “o”
  - Tempo total e tempo da transposição no modo “t”
- Flags de Otimização
- Nenhuma outra atividade executada ao mesmo tempo

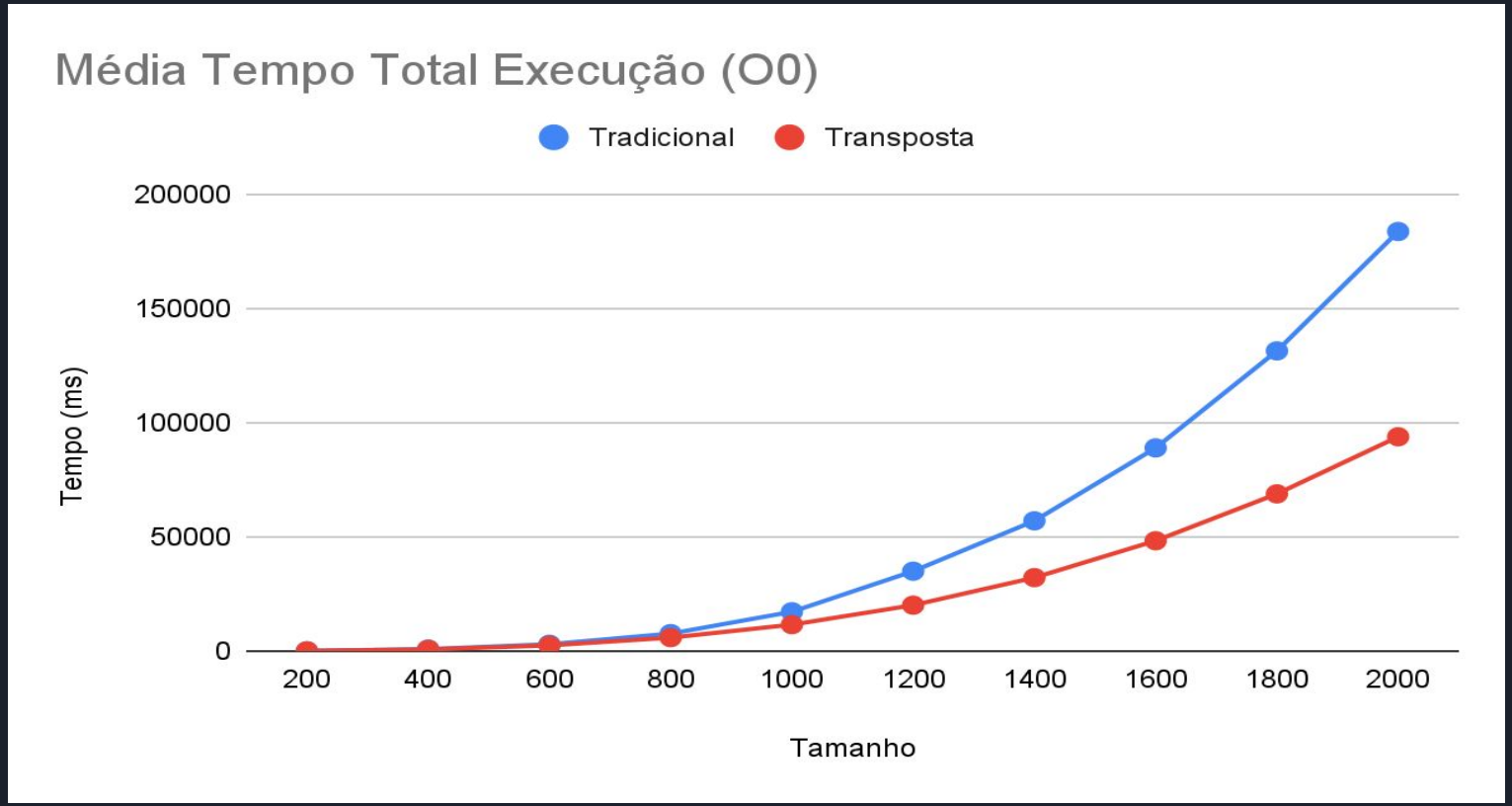
Flags de Otimização

- Otimização O0
  - “Reduce compilation time and make debugging produce the expected results. This is the default.” (man gcc)
- Otimização O3
  - “Optimize yet more. -O3 turns on all optimizations specified by -O2 and also turns on the following optimization flags…” (man gcc)

Análises (O0)

Média Tempo Total Execução (O0)				
Tamanho	**Tradicional (ms)**	**Transposta (ms)**	**Speed Up**	**Melhoria**
200	98,4	92,9	94,41056911	1,059203445
400	857,4	734,5	85,66596688	1,167324711
600	2987,9	2480,1	83,00478597	1,204749808
800	7641,3	5883,9	77,00129559	1,298679447
1000	17204,2	11559,1	67,18766348	1,488368472
1200	34957	20092,7	57,47833052	1,739786091
1400	57002,5	32146,5	56,39489496	1,773210147
1600	88917,4	48282,6	54,30050811	1,841603393
1800	131383,3	68807,6	52,37164845	1,909430063
2000	183622,7	93791,2	51,07821636	1,957781754

Análises (O0)


Média Tempo Total Execução (O0)
Tradicional
Transposta
200000
150000
100000
50000
0
Tempo (ms)
200
400
600
800
1000
1200
1400
1600
1800
2000
Tamanho

Análises (O0)

Tempo Transposição *vs* Tempo Execução			
Tamanho	Tempo Execução (ms)	Tempo Transposição (ms)	% tempo total
200	92,9	0	0
400	734,5	1	0,1361470388
600	2480,1	3,7	0,1491875328
800	5883,9	7,4	0,1257669233
1000	11559,1	11,6	0,1003538338
1200	20092,7	16,4	0,0816216835
1400	32146,5	23,5	0,07310282612
1600	48282,6	33,7	0,06979740113
1800	68807,6	42,5	0,06176643278
2000	93791,2	56	0,05970709406

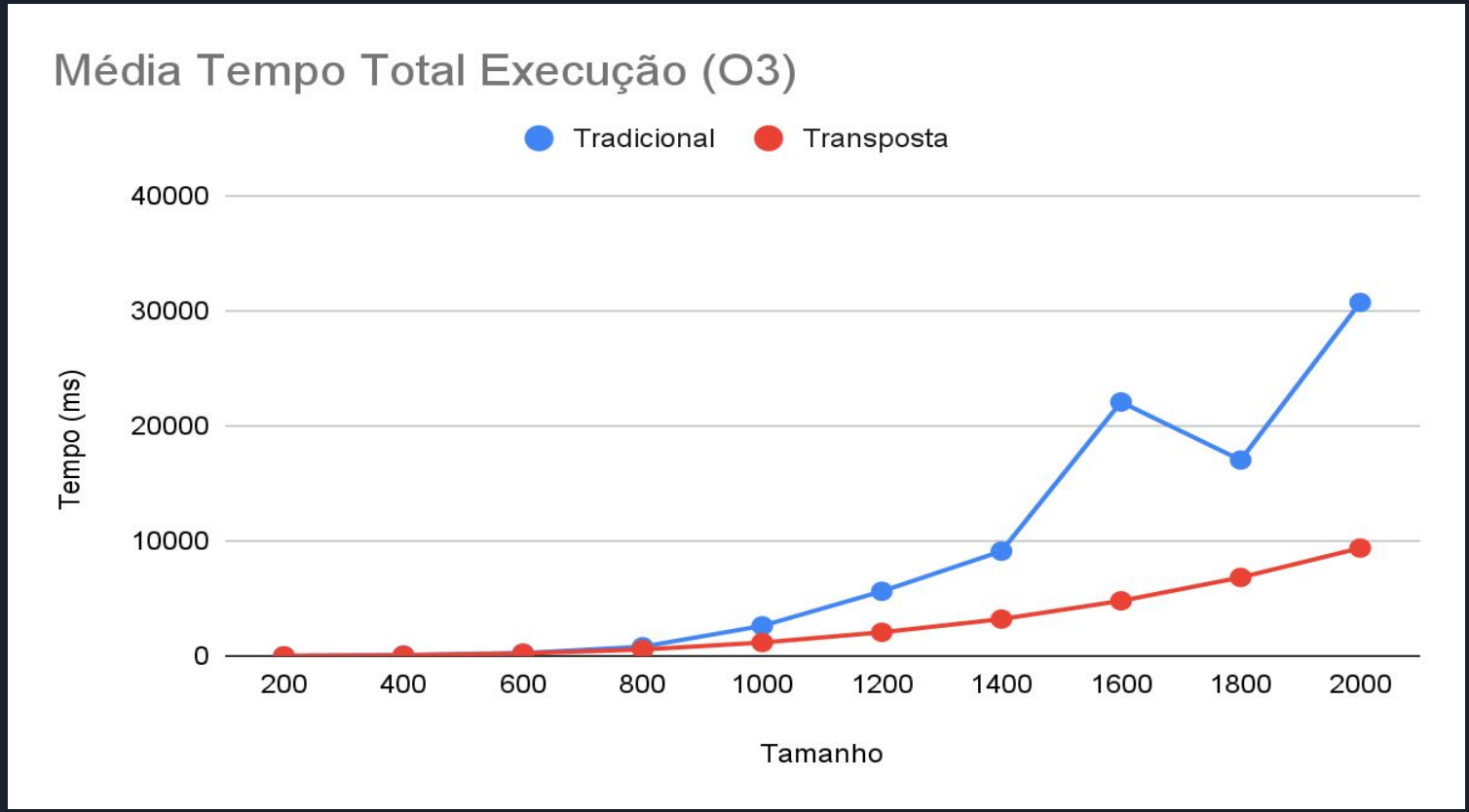
Análises (O0)

Execução Valgrind		
Parâmetro	**Tradicional**	**Transposta**
I refs	5.312.092.748	5.322.611.299
D refs	3.192.281.880	3.155.678.432
LLd misses	58.403	73.936
LLd miss rate	0,00%	0.0%
D1 misses	48.746.128	5.737.856
D1 miss rate	1,50%	0,20%
LL refs	48.749.860	5.741.613
LL misses	61.381	76.935
LL miss rate	0,00%	0,00%

Análises (O3)

Média Tempo Total Execução (O3)				
Tamanho	**Tradicional (ms)**	**Transposta (ms)**	**Speed Up**	**Melhoria**
200	8,6	8,3	96,51162791	1,036144578
400	71,9	66,2	92,07232267	1,086102719
600	253,3	224	88,43268851	1,130803571
800	804	555,8	69,12935323	1,446563512
1000	2614,2	1164	44,52605003	2,245876289
1200	5617,3	2054,3	36,57095046	2,734410748
1400	9102	3201,8	35,1768842	2,842775939
1600	22059,5	4786,8	21,69949455	4,608402273
1800	17016,3	6819,9	40,07863049	2,495095236
2000	30705,6	9371,4	30,5201657	3,276522185

Análises (O3)


Média Tempo Total Execução (O3)
Tradicional
Transposta
40000
30000
20000
10000
0
Tempo (ms)
200
400
600
800
1000
1200
1400
1600
1800
2000
Tamanho

Análises (O3)

Tempo Transposição *vs* Tempo Execução			
Tamanho	Tempo Execução (ms)	Tempo Transposição (ms)	% tempo total
200	8,3	0	0
400	66,2	0	0
600	224	1	0,4464285714
800	555,8	2	0,3598416697
1000	1164	4	0,3436426117
1200	2054,3	7	0,3407486735
1400	3201,8	9,6	0,2998313449
1600	4786,8	17,1	0,3572323891
1800	6819,9	28	0,4105632047
2000	9371,4	30,7	0,3275924622

Análises (O3)

Execução Valgrind		
Parâmetro	**Tradicional**	**Transposta**
I refs	410.351.086	368.347.053
D refs	182.020.188	139.389.079
LLd misses	58.400	73.933
LLd miss rate	0,00%	0,10%
D1 misses	48.867.335	5.738.470
D1 miss rate	26,80%	4,10%
LL refs	48.870.817	5.741.974
LL misses	61.281	76.831
LL miss rate	0,00%	0,00%

Considerações Finais

- Flag O3 deixa o aplicativo significantemente mais rápido;
- Anomalia com 1600 para tipo ‘o’;
- Redução na quantidade de referências de instrução de O0 para O3;