MAC0219 - MiniEP02
Édio Cerati Neto - NUSP 9762678

01.

Executando o programa com o tamanho de matriz $N = 2048$ e com os algoritmos *matrix_dgemm_0* e *matrix_dgemm_1*, repetindo tal experimentação 10 vezes, obtive os seguintes resultados, com os tempos de execução medidos em segundos:

matrix_dgemm_0	matrix_dgemm_1
476,080352	32,979183
464,148421	32,985493
465,701479	33,010109
468,831830	33,001695
476,762674	33,035587
469,540155	33,000184
491,736555	32,997194
489,355595	32,971503
490,264520	33,012089
484,391282	32,997934

Temos uma média amostral $\mu_0 = 477,681286s$ para o algoritmo 0 e $\mu_1 = 32,999097s$ para o algoritmo 01

Segue também um desvio padrão amostral $s_0 = 10,603217$ e $s_1 = 0,018189$, respectivamente.

Será adotado um nível de confiança de 95%, sendo assim o intervalo de confiança pode ser definido da forma:

$$(\mu_0 - \frac{t_0 * s_0}{\sqrt{n}}, \mu_0 + \frac{t_0 * s_0}{\sqrt{n}})$$

Onde t_0 é o valor crítico da distribuição t de Student a um nível de confiança 95% e n é o tamanho da amostra, ou seja, 10

De uma tabela com os valores de t de Student, segue que o valor crítico para nível de confiança 95% e 9 graus de liberdade é $t_0 = 2,262$

Efetuando os cálculos, obtemos:
$IC_0 = (469,686461; 485,676112)$
$IC_1 = (32,985383; 33,012812)$

Sendo assim, é possível concluir que uma melhora considerável nessa implementação, de cerca de $14,47$ vezes, uma vez que o $algoritmo - 01$ foi implementado com a intenção de otimizar o uso das linhas da matriz armazenados no cache, uma vez que ele percorre sempre a mesma linha da matriz B a ser multiplicada, variando os elementos em ordem de coluna na matriz A para permitir tal cálculo mais eficiente.

Ao seguir para a coluna seguinte de A, apenas a linha seguinte de B será iterada, otimizando o uso limitado do cache.

02.

Os experimentos para *matrix_dgemm_2* são análogos aos mostrados no item 01, utilizando os mesmos parâmetros.

Foram obtidos tais resultados para o tempo de execução, em segundos:

matrix_dgemm_2
29,451578
28,492932
28,772256
28,487838
28,277709
29,004078
28,075746
28,702391
28,089922
29,642938

Neste experimento, obteve-se média amostral $\mu_2 = 28,699739s$, com um desvio padrão amostral $s_2 = 0,403685$

Podemos obter o intervalo de confiança de maneira análoga ao do item 01, resultando em:

$IC_2 = (28,296053; 29,103424)$

O *speedup* nessa implementação não foi tão evidente como na anterior, sendo cerca de 1,15 vez mais rápido que o de *matrix_dgemm_1*, que para tais parâmetros adotados (tamanho de matriz $N = 2048$) resulta em um ganho de cerca de 4 segundos.

A melhora se dá pois a técnica de blocagem aproveita o cache de maneira mais eficiente que *matrix_dgemm_1*, pois reduz o problema da multiplicação de uma matriz de N muito grande para a multiplicação de várias submatrizes $8x8$ (tamanho de bloco escolhido, pois equivale ao tamanho do bloco de memória alocado no cache por vez, de acordo com as especificações do computador em que ocorreu a execução), o que garante que as duas submatrizes a serem multiplicadas por vez possam estar alocadas no cache

03.

Conforme exposto no item 02, a técnica de blocagem aplicada divide as matrizes a serem multiplicadas em várias pequenas matrizes de tamanho $8x8$. Este tamanho de bloco foi adotado pois equivale ao tamanho que é alocado no cache a cada *cachemiss*, de acordo com as especificações do hardware em que os experimentos foram efetuados.

Assim, garante-se que as duas submatrizes estarão alocadas no cache, garantindo uma multiplicação mais rápida.