NOME : ENZO OLIVEIRA FERNANDES

A solução realizada foi alterar a ordem na qual os loops são realizados, ao invés de i->j->k como na função matrix_dgemm_0 foi utilizado a ordem i->k->j. Esta sutil mudança gerou uma enorme mudança no tempo gasto para realizar os cálculos.
Foram realizadas 30 execuções de ambas, em um notebook cujo processador é um Ryzen 3500U e o tempo médio da função matrix_dgemm_0 foi de 72.8s, enquanto o tempo de execução médio da função matrix_dgemm_1 foi de 6.3s, ou seja, menos de um décimo do tempo da função original. Também solicitei a um colega que compilasse o miniep em sua máquina para checar a diferença. O teste foi rodado apenas uma vez na máquina deste colega, com a função matrix_dgemm_0 gastando 81.2s e a função matrix_dgemm_1 gastando 7.64s. Apesar de proporcionalmente o resultado ser similar aos obtidos na minha máquina, é necessário ter cuidado ao realizar afirmações pois o teste foi rodado apenas uma vez, diferentemente da média de 30 execuções obtidas em meu notebook.

A razão pela ampla diferença de performance é o maior aproveitamento da memória cache do processador. Quando o controlador não encontra um valor necessário para realizar uma operação na memória cache, ele necessita buscar esse valor na memória RAM (mais distante do processador e com maior latência), tornando o cálculo mais lento. Uma boa analogia é imaginar a preparação de um bolo. Suponha que necessitamos de manteiga para realizar um dos passos da receita, se a manteiga estiver na geladeira, esse passo ocorre de maneira rápida, pois a geladeira se encontra muito próxima do forno. No entanto, caso não tenhamos manteiga na geladeira, devemos ir até o mercado para comprar e trazer para a cozinha. Nessa analogia, a geladeira é a memória cache(mais rápida, de fácil acesso e menor) e o mercado é a memória RAM(mais lenta, porém maior e mais longe).

No caso da mudança de (i-j-k) para (i-k-j), ocorre um maior aproveitamento pois é necessário percorrer menos endereços para encontrar os dados para realizar o cálculo. Em matrix_dgemm_0, o algoritmo é mais simples para nós humanos entender, pois visualizamos as matrizes em locais distintos, porém para o computador, tudo é uma imensa cadeia de bits em sequência, de forma que se quisermos calcularmos cada elemento da matriz resultado, é necessário realizar um percurso muito maior. Quando alteramos a ordem para (i-k-j), a operação "C(i,j) = C(i,j) + (A(i,k)*B(k,j));" percorre menos endereços, sendo que a probabilidade de ocorrerem cache hits é muito maior visto que os endereços dos valores requisitados estão próximos aos utilizados anteriormente.