Gustavo T. Mastrobuono, NUSP 10734411

Henrique de S. Q. dos Santos, NUSP 10819029

Jhordan P. V. Pesantes, NUSP 11733353

Witor M. A. de Oliveira, NUSP 10692190

Yorvin A. R. Carrion, NUSP 11733332

# **Projeto PCAM**

# **Particionamento**

Iremos particionar os dados da matriz A por linhas e os dados da matriz B por colunas, de forma que geramos 2 tarefas para cada linha/coluna das matrizes. Dessa forma, considerando matrizes de tamanho NxN, serão geradas 2N tarefas.

N tarefas ficarão encarregadas de enviar os dados necessários para a multiplicação de cada linha e coluna. As outras N tarefas receberão uma linha i da matriz A, uma coluna j da matriz B e realizar o produto escalar entre essa linha e coluna recebidas (O resultado desta operação corresponde ao elemento da matriz C).

# **Comunicação**

A comunicação se dá pelo envio das linhas da matriz A e colunas da matriz B da tarefa 0 para as demais, como também pelo retorno do resultado obtido pela multiplicação escalar entre as linhas da matriz A e as colunas da matriz B das demais tarefas para a tarefa 0. Em cada tarefa haverá uma comunicação ao realizar a multiplicação escalar, ou seja, existe um acumulador que armazena a soma da multiplicação de cada elemento de uma linha e coluna, onde esse acumulador será compartilhado entre cada tarefa.

# **Aglomeração**

Através do MPI, o processo de rank zero irá ficar responsável pela distribuição das tarefas entre os outros processos. O processo de rank 0 irá criar, alocar e inicializar as matrizes A, B e C. Feito isso, o processo de rank 0 irá repartir em paralelo, utilizando o OMP (parallel for), as linhas da matriz A e as colunas da matriz B entre os processos existentes através do MPI\_Send. Através do MPI\_Recv, cada processo irá receber um pedaço da matriz A e da matriz B.

Os processos de rank diferente de zero ficarão responsáveis por efetuar a multiplicação da linha recebida da matriz A e a respectiva coluna da matriz B. Cada processo efetuará essa multiplicação em paralelo, utilizando o OMP (parallel for).

No final, o processo de rank zero irá receber cada parte de C que fora calculada entre os processos com rank diferente de zero. Para tal, cada processo terá que efetuar um MPI\_Send para o processo rank zero, que receberá as partes de C através do MPI\_Recv. Por fim, o processo de rank zero irá exibir na tela o resultado obtido na matriz C.

# **Mapeamento**

Considerando os nós de um cluster que possuem desempenho homogêneo e sabendo que os P processos gerados possuem a mesma carga de trabalho, podemos distribuir os P processos igualmente entre os elementos de processamento. Isto geraria um processo P para cada elemento de processamento, caso P = PROC Elementos de Processamento. Caso o desempenho dos nós seja diferente, deve-se considerar o mapeamento dinâmico (em tempo de execução), atribuindo o(s) processo(s) ao(s) nó(s) do cluster que está com menos carga de trabalho, por exemplo. Isto pode ser feito, por exemplo, considerando-se alguma métrica de desempenho, uso da CPU etc.