UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Bacharelado em Ciência da Computação

Programação Paralela e Distribuída

Profº: Gabriel Pereira da SIlva

Relatório - Trabalho 1

Eduardo Barbosa - 116150432

Gabriel Villares - 114089936

Questão 4

Objetivo

A quarta questão propõe a paralelização de um programa que calcula um fractal de mandelbrot. Além disso, é desejável que as comunicações necessárias entre os processos sejam paralelas a realização dos cálculos, ou seja, que sejam utilizados comunicadores não bloqueantes.

Metodologia

O processo raiz (processo 0) é responsável por toda parte do código que faz a preparação da janela de exibição e que faz a impressão da figura do fractal. Deste modo, a parte paralelizada consiste apenas dos 2 loops responsáveis pelos cálculos dos pontos pertencente ao fractal.

A parametrização dos loops, consiste em cada processo realizar os cálculos de uma linha e pular n processos para a próxima linha. A cada ponto encontrado é utilizado a função MPI_Isend para enviar o ponto encontrado para o processo 0, que posteriormente irá imprimir. A utilização da função MPI_Isend se faz necessária, pois evita que os processos fiquem bloqueados esperando que a mensagem seja recebida por outro processo, deste modo os processos ficam livres para buscar por outros prontos. Vale ressaltar que o processo 0 também contribui na busca pelos pontos e que a cada mensagem enviada os processos devem incrementar o seu total de mensagens enviadas.

Após encontrar todos os pontos é realizado um MPI_Reduce para que o processo 0, seja informado do total de mensagens enviadas, por todos os processos durante a busca, posteriormente o processo 0 irá realizar o recebimentos (MPI_Recv) de todos os pontos e imprimi-los.

Para realização da contagem do tempo de execução foi considerado apenas os cálculos para encontrar os pontos pertencentes a um fractal, desconsiderando o tempo gasto para preparação da janela de exibição e impressão da do fracta. Os testes foram realizados com 1, 2, 4, 8, 16 e 32 processos executando cada um 3 vezes e tirando a média do tempo.

Conclusão

Durantes os testes foi observado que para figuras com resoluções maiores que 2000 x 2000 ocorria um gargalo na função MPI_Reduce. Acreditamos que isso ocorra pela a grande quantidade de mensagens enviadas pelo MPI_Isend que ainda não tinham sido recebidas. Deste modo os testes foram realizados para resolução de 1000 x 1000.

Após a realização dos testes foi possível observar que acima de 16 processo o tempo de execução passou aumentar e consequentemente obtivemos um retorno negativo no speedup. Assim as melhor situação foi a execução com 16 processos.