Calculo de Determinante MPI

Leonardo Pereira Medeiros - Super Computação

Como Rodar

.Instalação das dependências:
sh install.sh

.Compilação do codigo:
make

.configuração do hostfile:
Deve-se alterar o arquivo hf, referente ao arquivo de hosts, ajustando os ips das máquinas que deseja-se executar os processos MPI.

.Execução do script:
sh main.sh

Introdução

Este projeto consiste na implementação de um código que calcula a determinante de uma matriz quadrada, distribuindo o cálculo usando recursos MPI(Message Passing Interface). Com a finalidade de utilizar suas ferramentas para a análise de ganhos de desempenho.

Para calcular a determinante de uma matriz quadrada foi utilizado o método de laplace, o qual define o determinante de uma matriz pela soma algébrica dos produtos dos elementos de uma linha, ou coluna, pelos respectivos cofatores(complemento algébrico).

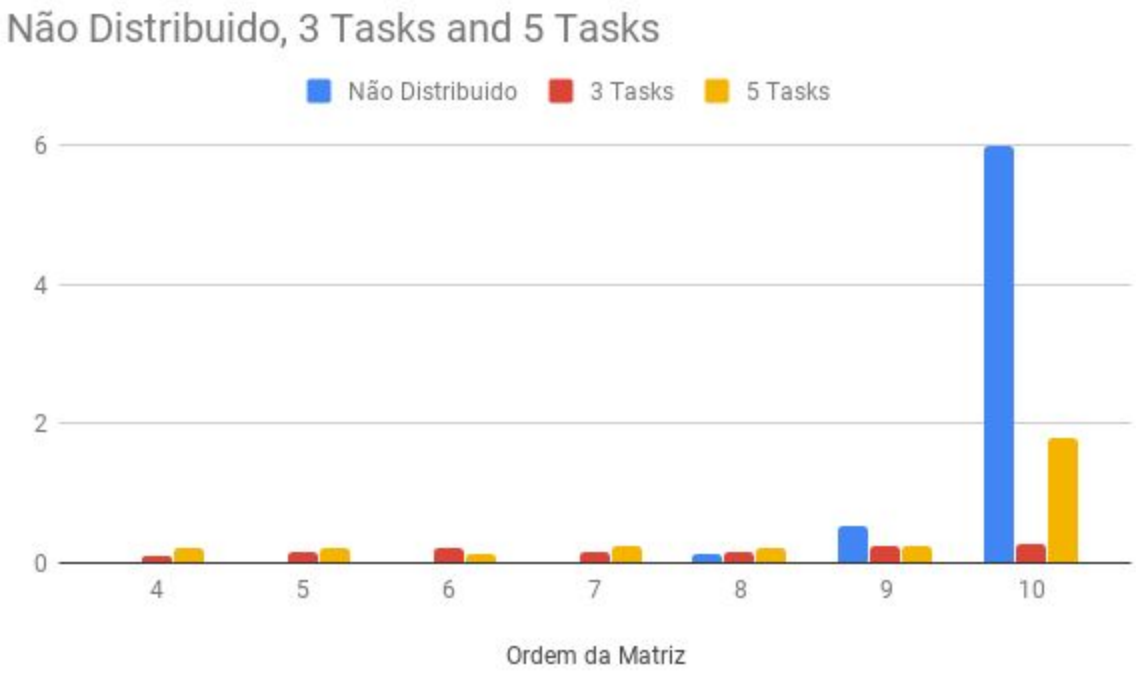
Funcionamento Do Programa

O script funciona de forma distribuída, e pode ser escalado com a demanda de processamento, ao setar o número de processos MPI requisitados ao executar o script, essa distribuição ocorre na segmentação do código entre o master e os workers. O master é o responsável por gerar a matriz inicial com valores “aleatórios”, enviando esse dado através do método send para os processos workers, que recebem essa informação com o método Reciv, que então realizam o cálculo da determinante parcial para um intervalo de elementos da linha da matriz original, assimilado ao seu id, o valor calculado por cada worker é então enviado para o master que apenas soma esses valores apresentando o resultado final(determinante).

Cada processo pode ser atrelado a uma máquina, essas são listadas no arquivo de hosts, denominado hf, caso o número de processos declarados seja maior que o número de máquinas o computador utiliza os seus cores para executar os processos remanescentes .

Analise de desempenho

Para a análise de desempenho foi mensurado o tempo de execução do código para o caso não distribuído, distribuído com 3 tasks (um master e dois workers) e distribuído com 5 tasks (um master e quatro workers), foi obtido o seguinte resultado:



Tempo de Execução (seg) x Ordem da Matriz

Ordem da Matriz	Tempo de execução		
	Não Distribuido	3 Tasks	5 Tasks
4	5.47E-05	0.0849472	0.204099
5	9.10E-05	0.162623	0.203636
6	0.000121162	0.20721	0.119445
7	0.000448073	0.162464	0.246593
8	0.120842	0.167225	0.205862
9	0.536378	0.246959	0.249918
10	5.98212	0.283736	1.79982

Conclusão

Pode-se observar que para matrizes de dimensões pequenas o código sem distribuição é mais rápido, isso ocorre por que há um custo para a transferência de dados entre o master e os workers, custo esse que é maior que o custo do cálculo da determinante parcial para matrizes de ordem pequena, já para matrizes de ordem alta os códigos com distribuição são mais efetivos, e mais regulares quanto ao aumento de tempo devido ao aumento de ordem.

Esses testes foram realizados em 3 máquinas t2.micro, por ser uma máquina de baixo performance os testes se limitaram para um valor de ordem 10, com valores maiores a diferença de tempo de execução entre os códigos se mostraria mais acentuada, e neste caso provavelmente o código com 5 workers se demonstraria mais eficiente.

Logo fica evidente que o uso de MPI é vantajoso apenas quando se tem uma grande quantidade de dados e operações, e mesmo assim, deve-se ponderar o número correto de workers para determinado código.