Universidade Federal de São Carlos
Centro de Ciências Exatas e de Tecnologia
Departamento de Computação

Programação Paralela e Distribuída

Exercício Programa 1: Aproximação de Pi por integral

Professor: Hermes Senger

Hugo da Silva e Souza 761211
Engenharia de Computação

São Carlos, 6 de Dezembro de 2022

1. Introdução e Objetivos

O objetivo deste exercício proposto é realizar a execução de um job no cluster openHPC da UFSCar, assim como configurar um experimento de escalabilidade e analisar seus resultados a partir da aplicação da aproximação do número PI pelo método da integral.

2. Desenvolvimento

A partir do entendimento teórico e prático do experimento, prosseguiu-se para a criação do container Singularity a partir do arquivo de definição, no qual possuía a aplicação do cálculo da aproximação de Pi utilizando o método por integral. Em seguida foi escrito um arquivo shell script com a finalidade de executar a aplicação do container no cluster por meio de um job do sistema de agendamento de supercomputadores Slurm.

A execução da aplicação foi realizada utilizando os códigos disponibilizados pelo docente em três versões: a versão sequencial, com paralelização com Pthread e com paralelização com OpenMP. Todas as execuções foram realizadas com o número de passos para o cálculo de Pi com um valor fixo de 1000000000 (um bilhão), e as aplicações com paralelização foi variado o número de threads em 1, 2, 5, 10, 20 e 40.

Com o job escrito, foi submetido o mesmo no cluster no qual foi executado com sucesso, sem apresentar nenhum erro e foram obtidos os resultados de tempo de execução de todas versões das aplicações em um arquivo .out.

3. Apresentação dos Resultados

Com o arquivo de resultados em mãos, foi realizada a análise de speedup e eficiência das aplicações paralelas em relação a aplicação sequencial, como exemplificado nas tabelas abaixo (Tabela 1 - 3).

Sequencial	
Tempo de Execução	3,429607

Tabela 1: Resultado da aplicação sequencial.

Pthread			
# processos	Tempo de Execução	Speedup	Eficiência
1	3,446181	0,995191	0,995191
2	1,717631	1,996708	0,998354

5	0,730340	4,695905	0,939181
10	0,412291	8,318413	0,831841
20	0,278122	12,331304	0,616565
40	0,190734	17,981099	0,449527

Tabela 2: Resultado da aplicação paralela com Pthread.

OpenMP			
# processos	Tempo de Execução	Speedup	Eficiência
1	3,425398	1,001229	1,001229
2	1,716441	1,998092	0,999046
5	0,712013	4,816776	0,963355
10	0,421429	8,138042	0,813804
20	0,282085	12,158062	0,607903
40	0,155445	22,063154	0,551579

Tabela 3: Resultado da aplicação paralela com OpenMP.

4. Análise e Conclusão

A figura 1 apresenta o gráfico de speedup para as aplicações paralelas com Pthread e OpenMP. Podemos observar que para a execução com OpenMP obtemos melhores resultados de speedup e por consequência a eficiência com mais de 10 processadores.

Podemos analisar que o speedup linear e a eficiência ideal máxima não acontecem na prática em função das partes não paralelizáveis das aplicações. A paralelização com OpenMP possui maior eficiência em relação ao Pthread, correlacionado ao melhor balanceamento de carga fornecido pela política de distribuição de processamento do paradigma.

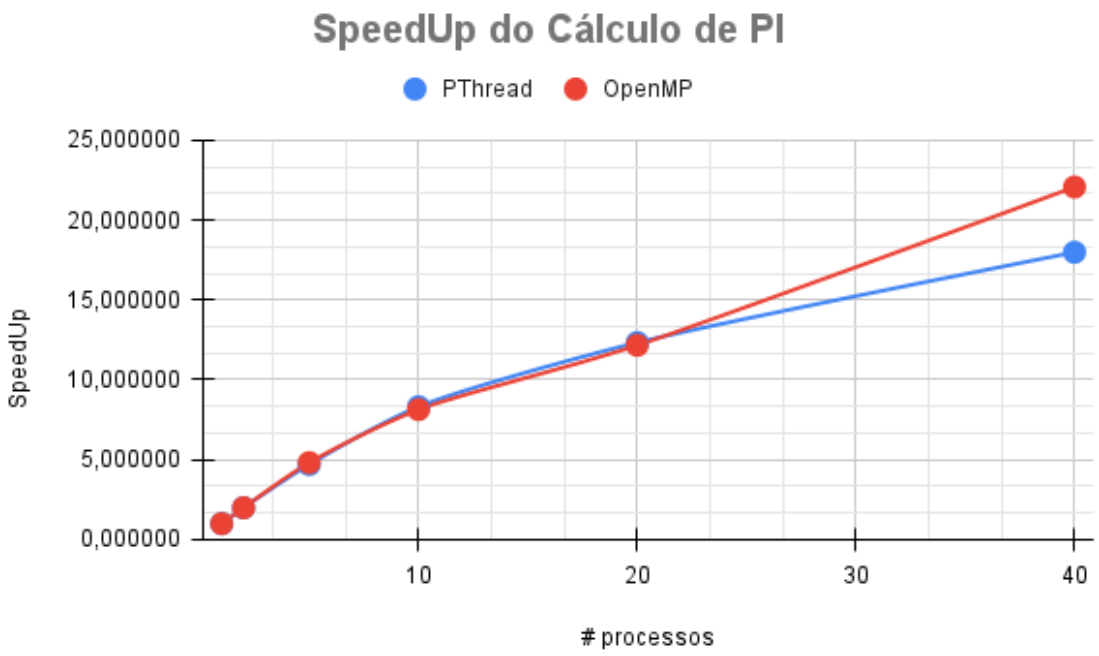

Figura 1: Gráfico do Speedup da execução paralela do cálculo da aproximação de Pi

A correlação linear do speedup ocorreu de forma bem sucinta quando foram proporcionados para aplicação a quantidade de 2, 5, e 10 processadores, após isso não ocorreu relação linear. Concluímos que a escalabilidade da aplicação não ocorre como o ideal, pois como já citado anteriormente, há partes não paralelizadas.