

SuperComputação

Aula 16 – Passagem de Mensagens

2018 – Engenharia

Igor Montagner, Luciano Soares [<igorsm1@insper.edu.br>](mailto:igorsm1@insper.edu.br)

Visão geral do curso

- 1. Multi core – tarefas independentes
 - 2. Concorrência – sincronização
 - 3. Sistemas com memória distribuída
- } Memória compartilhada

Objetivos

- Disparar vários processos em um mesmo computador
- Sincronizar processos por troca de mensagens
- Trocar mensagens de forma assíncrona

Troca de mensagens

- Programa dividido em processos na mesma ou em outras máquinas;
- Não existe memória compartilhada, dados são trocados enviando (e recebendo) mensagens;
- Mensagens podem ser síncronas ou não;

MPI (Message Passing Interface)

- Criado pelo fórum MPI organizado em 1992 com participação de: IBM, Intel, TMC, SGI, Convex, Meiko
- Várias versões (MPI-1 – 1994, MPI-2 – 1997, MPI3 – 2012)
 - <http://www.mpi-forum.org><http://www.mpi-forum.org>
- MPI não é um produto (implementação), é um protocolo

MPI (Message Passing Interface)

Debian/Ubuntu:

```
apt install libboost-mpi-dev
```

MPI (Message Passing Interface)

Ambiente (environment):

- Responsável pela inicialização do ambiente MPI
- Obrigatório no início de todo programa. Quando sai de escopo finaliza o MPI

Comunicador (communicator):

- Grupo de processos, cada um com seu id (rank)
- Comunicador padrão é chamado de world e contém todos os processos
- Usado para enviar e receber mensagens entre os processos

MPI (Message Passing Interface)

send(recipient, tag, data):

- Envia dados para *recipient*
- *tag* para registrar o tipo da mensagem

recv(sender, tag, data_by_ref):

- Recebe dados de *sender*
- Mensagem precisa estar marcada com a mesma *tag*
- Guarda dados em *data_by_ref*

Mensagens são síncronas

MPI (Message Passing Interface)

- Custo de comunicação: enviar dados tem um custo. Pode não valer a pena enviar muitas mensagens pequenas;
- Divisão do trabalho: cuidado para não sobrecarregar um processo e deixar outros ociosos
- Deadlocks: toda mensagem enviada deve ser recebida. Se isto não for verdade seu programa pode travar.
- Sincronização: pares *send/recv* podem bloquear seu programa se não forem bem posicionados.

Atividade prática

Roteiro de introdução a MPI com C++

Referências

- Livros:
 - Hager, G. ; Wellein, G. **Introduction to High Performance Computing for Scientists and Engineers**. 1^a Ed. CRC Press, 2010.
- Internet:
 - https://www.boost.org/doc/libs/1_67_0/doc/html/mpi/tutorial.html
 -

Insper

www.insper.edu.br