Message Passing Interface (MPI): MPI e OMP

Paulo Sérgio Lopes de Souza
pssouza@icmc.usp.br

Universidade de São Paulo / ICMC / SSC – São Carlos
Laboratório de Sistemas Distribuídos e Programação Concorrente

Thread Safe

- MPI com OMP permite
  - Executar diferentes processos MPI em máquinas (ou nós ou *hosts*) distintas
  - Cada processo MPI com suas *threads* compartilhando memória na máquina local

- MPI precisa ser *thread* safe
  - Rotinas MPI devem funcionar corretamente durante execuções simultâneas de várias *threads*
  - Implica em respostas corretas mesmo com compartilhamento de dados/recursos entre *threads*

- Há um conjunto de funções no MPI para o suporte de *threads* com OMP
  - Principal/primeira a saber ***MPI_Init_thread( )***

int MPI_Init_thread(int *argc, char ***argv, int required, int *provided)

  - Análoga à ***MPI_Init( )***

  - Requisita suporte específico às *threads* em ***required***
  - Recebe em ****provided*** um retorno com o nível de suporte possível
    - Nível depende da versão do MPI e de como ele foi instalado
    - Padrão não garante que ****provided*** será maior ou igual ao solicitado em ***required***

Níveis de Suporte às *Threads* em MPI

int MPI_Init_thread(int *argc, char ***argv, int required, int *provided)

- Opções para ***requided***
  - ***MPI_THREAD_SINGLE***
    - Somente uma *thread* executará
    - Equivalente à função ***MPI_Init()***
  - ***MPI_THREAD_FUNNELED***
    - Permite que a *thread* que executou ***MPI_Init_thread( )*** faça chamadas MPI
  - ***MPI_THREAD_SERIALIZED***
    - Somente uma *thread* fará chamadas à biblioteca MPI por vez
  - ***MPI_THREAD_MULTIPLE***
    - Múltiplas *threads* poderão fazer chamadas ao MPI sem restrições

Importância do PCAM & Aspectos Práticos

- O uso de diferentes modelos de paralelismo (SHM e MP) dificulta projeto
  - PCAM deve especificar tais níveis
  - Projeto detalhado é imperativo para garantir qualidade e controle da complexidade

- Algumas informações práticas...

  - A função ***ompi_info | grep -i thread***
    - Informa se o MPI instalado tem suporte à programação *multithreading*

  - Para compilar
    - ***mpicc fonte.c -o binario -fopenmp***

  - Para executar (um exemplo simples)
    - ***mpirun –np <nr-processos> binario***

  - Inclua no seu código
    - ***#include <mpi.h>***
    - ***#include <omp.h>***

Exemplos

- Exemplo 01 - *Hello World*
  - Cada processo MPI criado com ***mpirun*** gera ***NT*** *threads* que imprimem

- Exemplo 02 – *Send & Recv* fora da região paralela do OMP
  - Processos *MPI != 0* (criados com ***mpirun***)
    - Geram ***NT*** *threads*
    - Incrementam ***i*** compartilhado e imprimem
    - Fora da região paralela enviam msg para o processo 0
  - Processo 0 recebe mensagens de todos os demais processos e as imprime

- *Exemplo 03 – Sends* dentro da região paralela do OMP
  - Semelhante ao Exemplo 02, mas agora as *threads* mandam as mensagens
    - Há ***NR_PROCS * NT*** mensagens enviadas ao processo 0

- Exemplo 04 – *Sends & Recvs* em *threads* OMP
  - Todos os processos executam ***NT*** *threads*, incluindo o processo 0
  - A *thread* ***T*** do processo 0 recebe todas as msgs enviadas pelas *threads* ***T*** dos demais processos
    - O rótulo ***msgtag*** organiza essas transmissões e garante pareamento de *threads de processos diferentes*

Referências

Barlas, G. (2014). *Multicore and GPU Programming: An integrated approach*. Elsevier. Capítulo 5, Seções 5.16 e 5.21.2.

https://www.open-mpi.org/doc/v3.0/man3/MPI_Init_thread.3.php

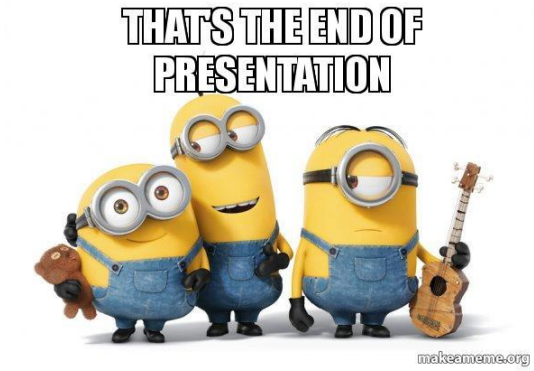
Rauber, T., & Rünger, G. (2013). *Parallel Programming*. Springer. Second edition. Capítulo 5.

Pacheco, P. (2011). *An introduction to parallel programming*. Elsevier. Capítulo 3.

Message Passing Interface (MPI): MPI e OMP

Paulo Sérgio Lopes de Souza
pssouza@icmc.usp.br

Universidade de São Paulo / ICMC / SSC – São Carlos
Laboratório de Sistemas Distribuídos e Programação Concorrente



CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico
CAPES
FAPESP