AULA 8

Programação Paralela e Distribuída

Sistemas Distribuídos

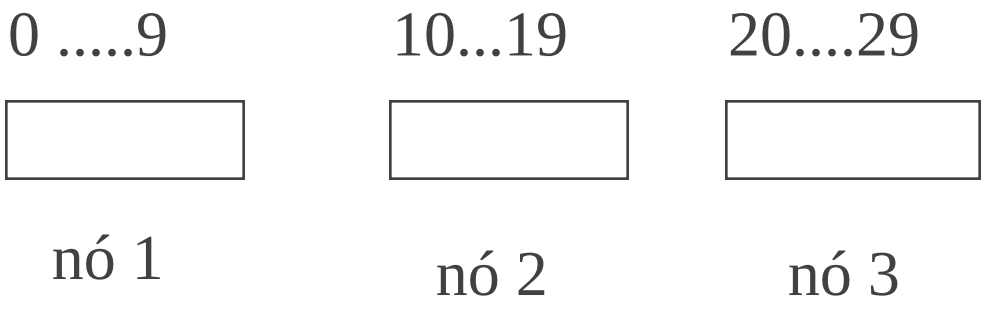
- Não possui memória compartilhada
- Bibliotecas: PVM, MPI
- Sistemas DSM (distributed shared memory): simula memória compartilhada, permitindo a programação no paradigma de variáveis compartilhadas

Programação com bibliotecas

- PVM (parallel virtual machine)
- MPI (message passage interface)
- programação utilizando passagem de mensagem

Modelo de computação: spmd (single program multiple data)

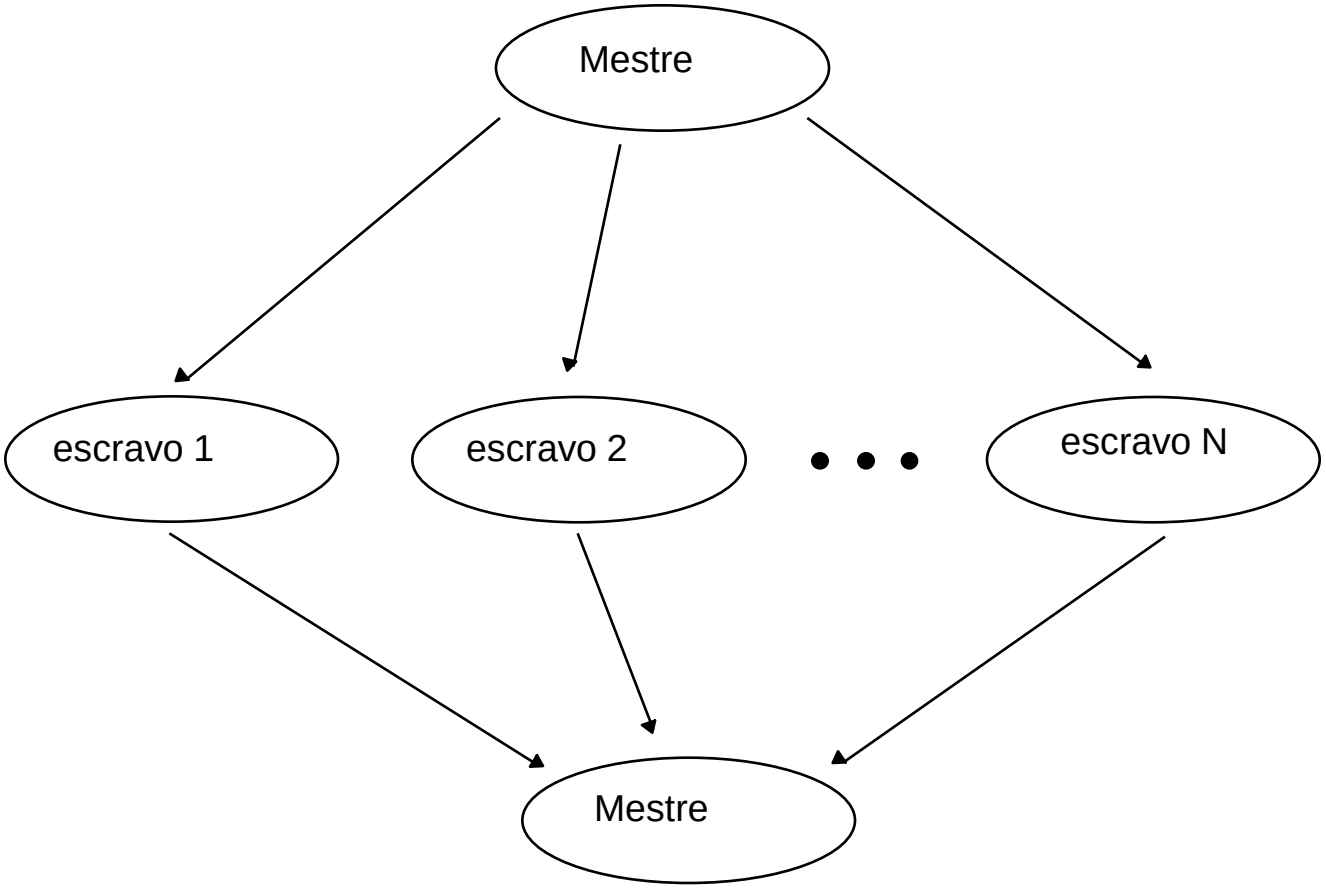
- todos os nós executam o mesmo programa sobre dados múltiplos



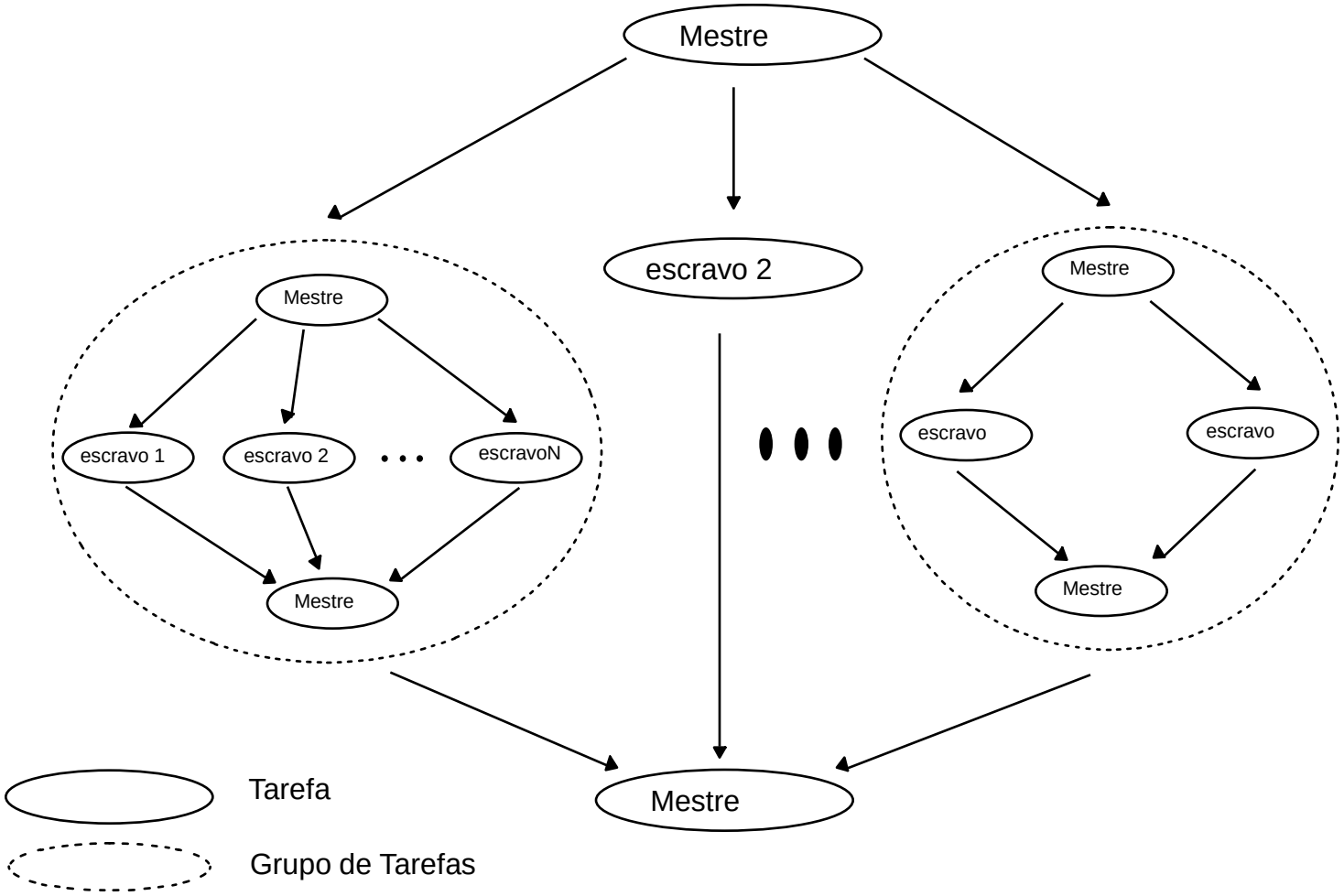
modelo de computação: esquema mestre escravo com spmd

- esquema mestre-escravo utilizando spmd: todos executam o mesmo programa, sendo que através de um controle interno ao programa um nó executa a função mestre eos demais são escravos.

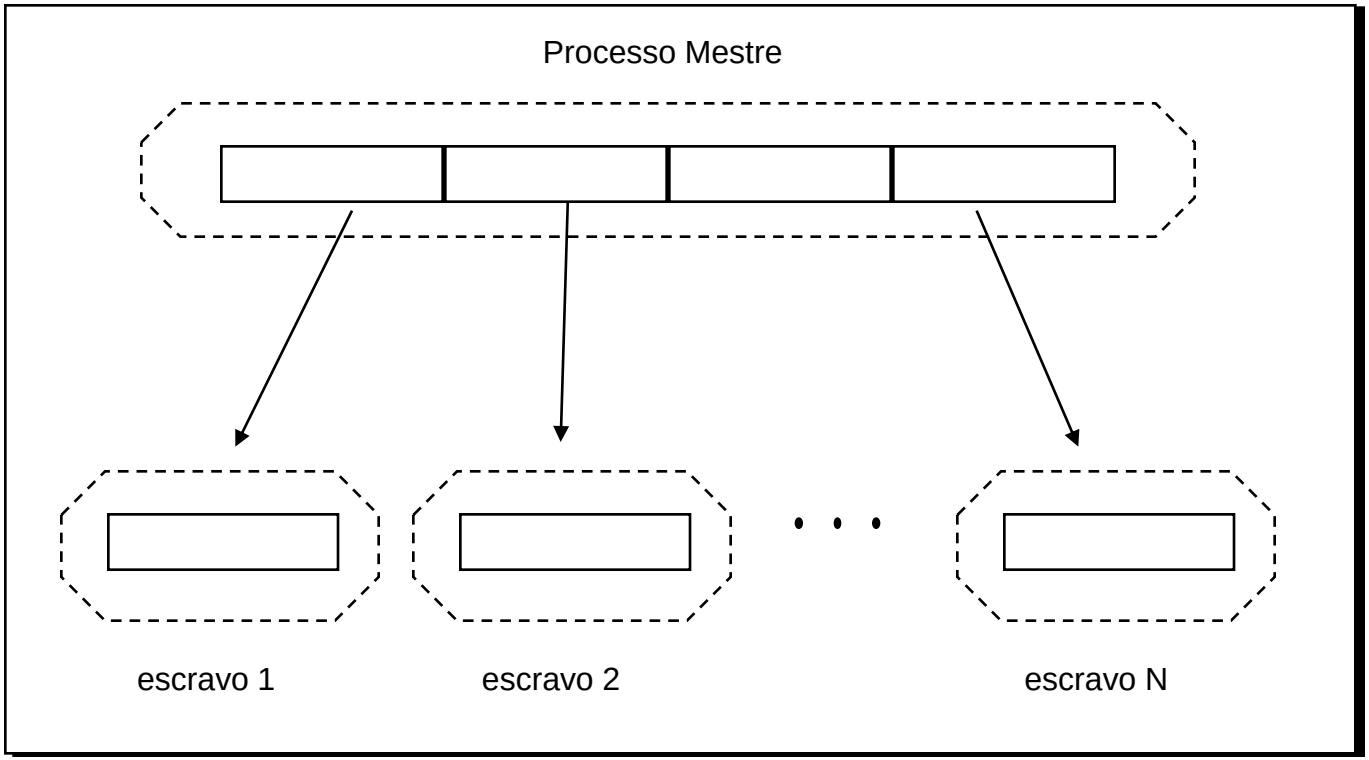
Modelo de computação



Grupos de tarefas



Exemplo : Soma dos elementos de um vetor



Para obter um bom desempenho

- Use granularidade grossa
- minimize o # de mensagens
- maximize o tamanho de cada mensagem
- use alguma forma de Balanceamento de carga

MPI

OPEN MPI

http://www.open-mpi.org/

MPI

- Processos: são representados por um único “rank”(inteiro) e ranks são numerados 0, 1, 2 ..., N-1. (N = total de processos)
- Enter e Exit

MPI_Init(int *argc,char *argv);

MPI_Finalize(void);

- Quem eu sou?

MPI_Comm_rank(MPI_Comm comm,int *rank);

- informa total de processos

MPI_Comm_size(MPI_Comm comm,int *size);

MPI

- Enviando Mensagens

MPI_Send(void *buf,int count,MPI_Datatype dtype,int dest,int tag,MPI_Comm comm);

- Recebendo Mensagens

MPI_Recv(void *buf,int count,MPI_Datatype dtype,int source,int tag,MPI_Comm comm,MPI_Status *status);

status: status.MPI_TAG

status.MPI_SOURCE

OPENMPI

Compilação: mpicc -o trivial trivial.c

mpiCC –o trivial trivial.cpp (ou trivial.c)

Arquivo contendo especificação dos hosts

(exemplo: hosts)

192.168.10.1

192.168.1.2

192.168.1.3

192.168.1.4

execução: mpirun -np 2 - -hostfile hosts trivial

Computador local: não é necessário especificar hosts

mpirun –np 4 trivial

exemplo

Soma dos elementos de um vetor

Processo rank0 : inicializa o vetor e envia um subconjunto de dados (divisão uniforme) para os demais processos.

Processo rank0 e demais processos: obtem as somas parciais

Processo 0: obtem o resultado final

exemplo: mpi-soma.c

Exercício para casa

- Obter o valor máximo contido em um vetor de inteiros.
- O processo com rank=0 inicializa o vetor.