Universidade de Aveiro
Departamento de Física

Recurso Teórico

Computação Paralela — Módulo MPI
2019/2020

15 de julho de 2020

Duração: 1h15 (Teste com consulta)

Todas as respostas devem ser devidamente justificadas.

1. Num programa MPI, pretende-se que o processo 0 envie para o processo 1 um *array* `b` constituído por `N` números reais de precisão simples.

a) Escreva as linhas de código que devem ser executadas pelo processo 0, de tal maneira que o envio seja *buffered* e não bloqueante, incluindo toda a gestão da memória usada pelo *buffer*. Não tem que escrever o código do processo 1.

b) Explique as diferenças entre um envio *buffered* bloqueante e um envio *buffered* não bloqueante. Suponha que poucas linhas depois do envio, o processo 0 executa instruções que alteram `b`. É necessário ter algum cuidado com o envio *buffered* bloqueante? E com o envio *buffered* não bloqueante? Justifique as suas respostas.

2. Num dado programa, o processo de *rank* 2 tem que enviar parte de `a`, um *array* quadrado de números inteiros, de dimensões 100×100, para o processo de *rank* 1, através de um única chamada de uma função de comunicação ponto a ponto do MPI. Escreva as linhas de código que permitem ao processo preparar e realizar o envio quando a parte do *array* a enviar é

a) os primeiros 50 elementos da coluna 6;

b) as linhas $0, 2, 4, \ldots, 98$;

c) as colunas $1, 3, 4, \ldots, 99$;

d) todos os elementos da diagonal principal.

Não tem que escrever a parte do código respeitante à receção pelo processo 1.

3. Comunicações coletivas.

a) Num programa de simulação molecular paralelizado, cada processo é responsável pela atualização de um certo subconjunto de partículas e tem registado na variável real de precisão dupla `Eparcial` o valor da energia total instantânea desse subconjunto. Pretende-se que, com uma periodicidade regular, o processo de *rank* 0 vá registando a energia total `Etotal` do sistema. Escreva e explique a linha de código da comunicação coletiva correspondente.

b) Num dado instante da execução de um programa MPI, cada processo tem registado um dado valor na variável inteira `m`. Pretende-se que, antes da execução dos seguintes passos, o valor mais pequeno de entre esses valores de `m` seja re-atribuído ao processo de *rank* 0, o segundo mais pequeno ao processo de *rank* 1, e assim sucessivamente. Escreva as linhas de código que permitem alcançar esse objetivo. Assuma que está disponível uma função **`sortint`** que recebe como input um *array* de inteiros e devolve um outro *array*, com os valores ordenados por ordem crescente.

4. A um certo ponto de um programa MPI, acabou de ser criado um novo comunicador cartesiano **`comm1`**, com duas dimensões, sem nenhuma periodicidade. Para o algoritmo que se está a programar, é necessário que cada processo conheça o *rank*, no novo comunicador, não só dos seus vizinhos à esquerda, à direita, em baixo e em cima, mas também do vizinho próximo diagonal à direita e em cima.

a) Escreva as linhas de código que permitem a cada processo identificar, no novo comunicador, o seu *rank* **`newid`** e os dos seus vizinhos próximos, **`left`**, **`right`**, **`top`** e **`bottom`**.

b) Para que cada processo fique a conhecer o *rank* do vizinho diagonal pretendido, pode-se usar uma única chamada de uma comunicação ponto a ponto, em que cada processo envia para o seu vizinho da esquerda o *rank* do seu vizinho de cima. Escreva essa linha de código.

c) Antes da linha de código da alínea anterior, cada processo tem que executar

```
topright = MPI_PROC_NULL;
```

Explique porquê.

d) O que significa afirmar que o comunicador cartesiano não é periódico? Se fosse periódico, as suas respostas às alíneas anteriores seriam as mesmas? Justifique.