Exame de Programação Paralela e Distribuída

Departamento de Ciência de Computadores
Faculdade de Ciências – Universidade do Porto
6 de Fevereiro de 2008
Duração: 2 horas e 30 minutos

Parte I
(responda em folhas separadas a cada uma das partes)

1. Em programação paralela é habitual designar a execução de uma aplicação como sendo uma computação concorrente, uma computação paralela e/ou uma computação distribuída. Indique os conceitos e/ou características que permitem diferenciar estas três formas de computação.

2. Um dos principais objectivos da programação paralela é reduzir o tempo necessário para resolver um problema. É comum referir-se uma medida de *speedup* para medir o grau de desempenho de uma aplicação paralela. Defina a medida de *speedup* e descreva de forma sucinta quais os factores que podem limitar o desempenho de uma aplicação paralela.

3. Defina o conceito de comunicador no modelo de programação MPI e explique a mais valia das funções `MPI_Comm_split` e `MPI_Cart_sub` na criação de novos comunicadores.

4. O seguinte código OpenMP exibe uma instrução onde pode verificar-se competição entre processos. Identifique qual a instrução em que tal se verifica, explique como é que pode de facto acontecer competição entre dois threads A e B na execução dessa instrução, e como é que poderia proteger essa região crítica de código.

```
  double area, pi, x;
  int i, n;
  ...
  area= 0.0;
#pragma omp parallel for private(x)
  for (i=0; i<n; i++) {
    x= (i+0.5)/n;
    area= area + 4.0/(1.0 + x*x);
  }
  pi= area/n;
```

5. Considere que uma determinada aplicação sequencial executa em 3200 segundos em 1 processador e que, por experimentação, verificou-se que 95% do tempo de execução é passado em procedimentos que se julga ser possível paralelizar. Segundo a Lei de

Amdahl, qual é o speedup máximo que se pode alcançar com uma versão paralela da aplicação executando em 10 processadores? E o limite máximo de speedup que se pode alcançar?

Parte II
(responda em folhas separadas a cada uma das partes)

1. A função `MPI_Allreduce` do MPI permite realizar operações globais de resumo em todos os processos de um comunicador. Escreva um procedimento de nome `Allsum(int *send, int *recv)` que resuma em todos os processos do comunicador universal a soma de um valor inteiro proveniente de cada um dos processos. Por outras palavras, o comportamento a obter pela execução de `Allsum(send, recv)` deverá ser igual ao conseguido com `MPI_Allreduce(send, recv, 1, MPI_INT, MPI_SUM, MPI_COMM_WORLD)`. Assuma que o número de processos no comunicador universal é uma potência de 2. Não utilize comunicações colectivas, apenas funções de comunicação ponto-a-ponto.

2. Escreva um programa OpenMP para determinar o maior elemento de uma matriz A[N][N] de valores reais (double), sendo necessário apenas as partes relevantes do programa, i.e. podem ignorar a parte de inicialização da matriz. O seu algoritmo deve definir um número de threads que seja adequado para abordar o problema.