Universidade Federal do Rural do Semi-Árido
Centro Multidisciplinar de Pau dos Ferros
Departamento de Engenharias e Tecnologia
PEX1272 - Programação Concorrente e Distribuída
Professor: Ítalo Assis

Lista de Exercícios - Porque computação paralela?

1. Suponha que precisamos calcular n valores e somá-los. Suponha que também tenhamos p núcleos e p seja muito menor que n. Então cada núcleo pode calcular uma soma parcial de aproximadamente valores n/p da seguinte maneira:

```
minha_soma = 0;
meu_pri_i = . . . ;
meu_ult_i = . . . ;
for (meu_i = meu_pri_i; meu_i < meu_ult_i; meu_i+=meu_desl) {
  meu_x = Compute_prox_valor(. . .);
  minha_soma += meu_x;
}
```

   Aqui o prefixo $meu\_$ indica que cada núcleo está usando suas próprias variáveis privadas e cada núcleo pode executar este bloco de código independentemente dos outros núcleos.

   Supondo que cada chamada para $Compute\_prox\_valor$ requer aproximadamente a mesma quantidade de trabalho, elabore fórmulas para calcular meu_pri_i, meu_ult_i e meu_desl. Lembre-se de que cada núcleo deve receber aproximadamente o mesmo número de elementos de computação no loop.

   Dica: primeiro considere o caso em que n é divisível por p. A partir daí, elabore fórmulas para o caso em que n não é divisível por p.

2. Suponha que precisamos calcular n valores e somá-los. Suponha também que tenhamos p núcleos e p seja muito menor que n. Então cada núcleo pode calcular uma soma parcial de aproximadamente n/p valores da seguinte maneira:

```
minha_soma = 0;
meu_pri_i = . . . ;
meu_ult_i = . . . ;
for (meu_i = meu_pri_i; meu_i < meu_ult_i; meu_i+=meu_desl) {
  meu_x = Compute_prox_valor(. . .);
  minha_soma += meu_x;
}
```

   Aqui o prefixo $meu\_$ indica que cada núcleo está usando suas próprias variáveis privadas e cada núcleo pode executar este bloco de código independentemente dos outros núcleos.

Considerando que a chamada com $i = k$ requer $k+1$ vezes mais trabalho que a chamada com $i = 0$ (se a chamada com $i = 0$ requer 2 milissegundos, a chamada com $i = 1$ requer 4, a chamada com $i = 2$ requer 6...), elabore fórmulas para calcular meu_pri_i, meu_ult_i e meu_desl. Comparado com a distribuição em blocos contíguos de iterações, sua fórmula deve reduzir a diferença do tempo de execução entre os núcleos.

3. Escreva um pseudocódigo para uma soma global estruturada em árvore. Suponha que o número de núcleos seja uma potência de dois.

4. Podemos usar os operadores bit a bit de C para implementar uma soma global estruturada em árvore. Implemente este algoritmo em pseudocódigo usando o operador OU EXCLUSIVO e o operador *shift* para esquerda.

5. Escreva um pseudocódigo para uma soma global estruturada em árvore. Considere que o número de núcleos pode não ser uma potência de dois.

6. Elabore fórmulas para o número de recebimentos e adições que o núcleo 0 realiza usando

   (a) uma soma global onde apenas o núcleo 0 realiza as adições;

   (b) uma soma global estruturada em árvore.

   Faça uma tabela mostrando os números de recebimentos e adições realizados pelo núcleo 0 quando as duas somas são usadas com 2, 4, 8, $\cdots$, 1024 núcleos.

7. Descreva um problema de pesquisa em sua área (Tecnologia de Informação, Engenharia de Software...) que se beneficiaria com o uso da computação paralela. Forneça um esboço de como o paralelismo seria usado. Você usaria paralelismo de tarefas ou de dados? Por quê?

Referência

- PACHECO, Peter S. An introduction to parallel programming. Amsterdam Boston: Morgan Kaufmann, c2011. xix, 370 p. ISBN: 9780123742605.