DEI-FCTUC

Computação de Alto Desempenho – 2012/2013
Exercícios N.º 3: Programação com Memória Partilhada em C

1. Escreva um programa que crie 8 threads que imprimam “Hello world, I am thread x”, onde x varia de 1 a 8 conforme a thread (Pthreads & OpenMP).

2. Escreva um programa que adicione dois vetores de 100000 floats em OpenMP.
   a) Como poderá dividir o trabalho por várias threads?
   b) Na diretiva “`for`”, qual a diferença entre as cláusulas `static` e `dynamic`?

3. Considere o problema da distribuição de calor numa superfície quadrada em que a temperatura das arestas é fixada a partir do exterior. A temperatura dos pontos no interior depende apenas dos pontos vizinhos. Isto significa que os pontos nas fronteiras da superfície ficam a uma temperatura constante, enquanto que a temperatura no interior deve ser calculada individualmente para cada ponto. As temperaturas são representados por uma matriz H, de elementos $h_{i,j}$ $(0 <= i <= n, 0 <= j <= n)$. Para um ponto interior $h_{i,j}$, $(0 < i < n, 0 < j < n$, i.e., para $(n-1)\times(n-1)$ pontos interiores) a temperatura é calculada com a seguinte expressão:

$$h_{i,j} = (h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1}) / 4$$

   Na prática, isto significa que o cálculo da temperatura de um ponto na iteração i usa os valores vizinhos na iteração i - 1. Deve calcular a temperatura em iterações sucessivas, até que a diferença em todos os pontos, duma iteração para a seguinte, permanece abaixo 0,01 º C. Deve também considerar um limite para o número de iterações. A temperatura exterior deve ser determinada por um argumento de linha de comando. Deve dividir a superfície em vários segmentos (por exemplo linhas, ou quadrados) de forma a distribuir os segmentos pelas várias threads. Pode utilizar Pthreads ou OpenMP.

4. Antes de implementar este exercício deverá discutir as opções que tomar com o professor. Será particularmente interessante a ideia de utilizar números de 128 bits (4 inteiros de 32 bits) e calcular todos os polinómios módulo $2^{127}-1$. Neste caso, todas as operações são mais lentas, pelo que as vantagens da paralelização poderão ser mais evidentes.

4. A avaliação de um polinómio consiste em atribuir um valor concreto às variáveis desse polinómio. Por exemplo, dado

$$p(x) = x^4 + 8x^3 + 2x^2 + x + 1$$

   a avaliação para $x=3$, resulta em

$$p(3) = 3^4 + 8\cdot 3^3 + 2\cdot 3^2 + 3 + 1 = 319$$

   A forma mais simples de calcular o valor da atribuição, que designaremos de “básica”, passa por calcular sequencialmente todos os termos (monómios) envolvidos, i.e., primeiro o 3^4, ao qual adicionamos depois o resultado de $8\cdot 3^3$, etc. Embora simples, este método é muito ineficiente, porque repete múltiplas vezes os mesmos cálculos (3^4, 3^3, etc.). De forma a evitarmos este problema, podemos recorrer a um método muito mais eficiente, conhecido como “método de Horner”, que reorganiza os termos do polinómio:

$$
\begin{aligned}
p(x) &= x^4 + 8x^3 + 2x^2 + x + 1 \\
&= \left(x^3 + 8x^2 + 2x^1 + 1\right)x + 1 \\
&= \ldots \\
&= \left(\left(\left(x + 8\right)x + 2\right)x + 1\right)x + 1
\end{aligned}
$$

Para um polinómio de grau n, este método requer no máximo n adições e n-1 multiplicações. Pelo contrário, o método básico requer até $(n^2+n)/2$ multiplicações e n adições, embora este número possa ser reduzido para $2n$-1 multiplicações, se calcularmos as potências iterativamente (i.e., x, depois x^2, depois x^3, etc.). É possível também paralelizar o método de Horner, por dois processos, se considerarmos os graus ímpares e pares do polinómio separadamente fazendo a adição no final. Este método é extensível para mais processadores.

Neste exercício os alunos deverão implementar um (ou mais) algoritmo(s) paralelo(s) para avaliação de polinómios. Para tal, poderão basear-se numa variante do *prefix sum problem* (também conhecida por *scan*, *prefix reduction*, ou soma parcial). A ideia é multiplicar determinados elementos de um vetor de acordo com o padrão da Figura 1. Estas multiplicações ocorrem em O(log n) passos síncronos.

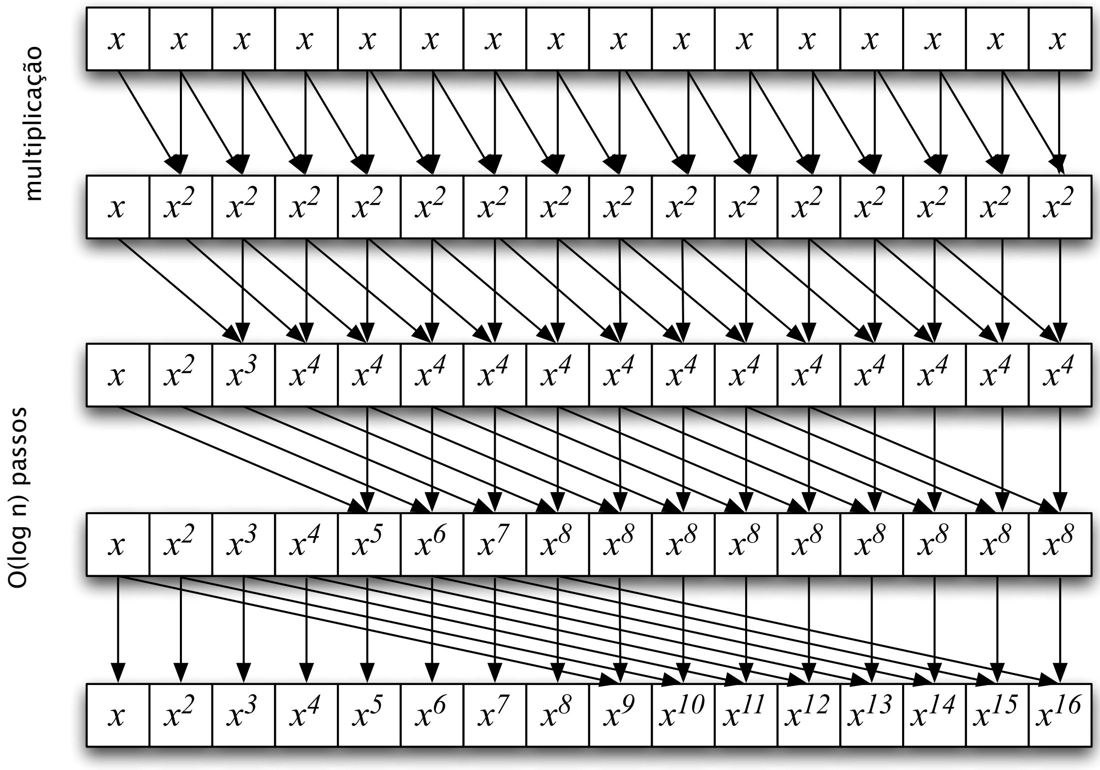

Figura 1 – Variante do *prefix sum problem*

Os alunos terão de criar uma implementação deste ou de outro algoritmo paralelo, utilizando, para isso, uma tecnologia à escolha, de entre as seguintes:

- Pthreads
- OpenMP
- Cilk
- SSE

Podem ainda combinar estas tecnologias para otimizar os resultados.

Os alunos terão acesso a código escrito em C, não só para fazer a leitura dos polinómios e valores a avaliar, mas também para escrever os resultados para a saída padrão[1]. A Figura 2 representa o formato dos dados de entrada. O polinómio em questão é $p(x) = x^4 + 8x^3 + 2x^2 + x + 1$ e terá de ser avaliado para 3 valores: x=3, x=2 e x=11. Os alunos devem notar que tanto o grau do polinómio (4), como o número de pontos a avaliar (3) são indicados no ficheiro.

```
4 1 8 2 1 1
3
3
2
11
```

Figura 2 – Formato de entrada de dados

O resultado deverá ser a avaliação do polinómio para os múltiplos valores dados, de acordo com a função evaluate() que é dada. Não deve ser adicionado qualquer texto ao formato desta função. Cada programa executável deve executar um algoritmo diferente. Se os alunos criarem três versões (p.ex., básico, Horner e paralelo), devem ter três executáveis diferentes e devem listar estes nomes no relatório.

Entre os ficheiros disponibilizados, encontram-se vários exemplos de polinómios, bem como um script de python, que permite gerar novos ficheiros de polinómios. Em Unix pode-se usar a seguinte linha de comando, para gerar um polinómio de grau 400 e com 200 valores para avaliar:

```
echo 400 200 | python generate-inputs.py > nome-ficheiro
```

Todos os coeficientes e valores a avaliar são inteiros e estão dentro de certos limites, sendo trivial alterar esses comportamentos, se for necessário criar polinómios diferentes.

É importante referir, que os resultados dos diferentes algoritmos (por exemplo, básico e Horner) podem diferir, para os **mesmos** valores de entrada. Isto resulta do facto de as operações de vírgula flutuante não serem nem associativas nem distributivas.

[1] A leitura e a escrita dos valores processa-se na íntegra para e a partir da memória, o que poderá constituir um problema para polinómios excecionalmente grandes. A resolução deste eventual problema ficará a cargo dos alunos.