Cálculo de ¶ usando threads

O objetivo deste projeto é compreender melhor o conceito de thread e como elas podem ser usadas para utilizar melhor os recursos de processamento disponíveis nos computadores modernos. Para isso, será construído um programa em C usando a biblioteca Posix Threads para calcular ¶ usando N threads simultâneas. O valor de ¶ pode ser aproximado pela série de Leibniz-Grégory:

$$\frac{\pi}{4} = \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1}$$

Observações

- Devem ser calculados pelo menos 1 bilhão (10^9) de termos da série; use variáveis reais de dupla precisão (`double`) nos cálculos;
- O programa deve dividir o espaço de cálculo uniformemente entre as N threads; cada thread efetua uma soma parcial de forma autônoma;
- Para evitar o uso de mecanismos de sincronização, cada thread `T[i]` deve depositar seu resultado parcial na posição `result[i]` de um vetor de resultados parciais. Após o término das threads de cálculo, o programa principal soma os resultados parciais obtidos por elas e apresenta o resultado final na tela;
- Para garantir um bom desempenho, evite operar muito com variáveis globais dentro das threads; copie as variáveis globais necessárias para variáveis locais (alocadas automaticamente na pilha da thread) e ao final dos cálculos copie os resultados finais de volta para as variáveis globais correspondentes;
- Devem ser medidos os tempos de execução **do programa** para execuções com 1, 2, 4, 8, 16 e 32 threads, em dois computadores com configuração distinta, como por exemplo:
    1. uma máquina mono-processada
    2. uma máquina dual-processada
- Para determinar o tempo de cada execução, use o comando `time` do UNIX.
- Compilar com os flags `-lm` (biblioteca matemática) e `-lpthread` (*threads* Posix).
- Ao executar seus experimentos, verifique se a máquina não está muito carregada, o que pode falsear os resultados (use o comando `top` para ver a carga da máquina);
- Para que os resultados tenham valor estatístico, devem ser feitas pelo menos 5 medidas de cada experimento e calculados o **tempo médio** e o **coeficiente de variação** entre elas (coeficiente de variação = desvio-padrão / média, em percentagem); são aceitáveis coeficientes de variação de até 5%; caso contrário, as medições devem ser refeitas.

A entregar

Deverá ser entregue ao professor o código-fonte desenvolvido (um arquivo `calcpi.c`) e um relatório (PDF no formato exigido) contendo as seguintes informações:

- Tabelas com os tempos de execução obtidos, valores médios e coeficientes de variação;
- Gráficos separados para as curvas de tempo médio de execução em cada plataforma e sua análise. Nos gráficos, o eixo X indica o número de threads e o eixo Y o tempo de execução;
- Um gráfico consolidado (todas as curvas no mesmo gráfico) com as curvas de tempo normalizadas e sua

análise crítica. Uma curva normalizada guarda apenas as proporções entre os valores de tempo; nela, os tempos de execução com N threads são divididos pelo tempo de execução obtido com apenas uma thread, na mesma plataforma;
- O diagrama de tempo da execução (simplificado, não precisa representar TODAS as threads);
- Respostas às seguintes questões:
    1. As threads implementadas são preemptivas ou cooperativas? Explique sua resposta.
    2. Que modelo de threads o sistema operacional que você usou implementa (N:1, 1:1 ou N:M) ? Como isso pode ser deduzido a partir dos experimentos?

Referências

- Posix Threads Programming (LLNL)

From:
https://wiki.inf.ufpr.br/maziero/ - **Prof. Carlos Maziero**

Permanent link:
https://wiki.inf.ufpr.br/maziero/doku.php?id=so:calculo_de_pi_com_threads

Last update: **2019/04/25 16:55**