MAC0438 – Programação concorrente – 1s2013
EP2
Datas de entrega: 06/05/2013 e 20/05/2013

Prof. Daniel Macêdo Batista

1 Problema

A busca por uma fórmula que calcule o valor de π com a máxima precisão possível é um problema que tem atraído a atenção de matemáticos desde a época de Arquimedes em 250 a.c. . Diversas fórmulas tem sido propostas desde então e muitas delas têm a propriedade de serem escritas como séries infinitas, onde o valor aproximado de π equivale ao resultado de alguma operação (adição por exemplo) sobre todos os termos da série. Quanto maior a quantidade de termos na série, mais preciso o valor de π.

Aproximações calculadas por séries infinitas podem facilmente ser mapeadas para algoritmos paralelos. Cada termo da série pode ser armazenada em um vetor e cada thread/processo de tempos em tempos lê o valor de uma posição do vetor e calcula o próximo valor. Para garantir que o programa termine, algum critério de parada deve ser definido, como por exemplo, a diferença entre o valor anterior de π e o valor atual de π. O critério de parada pode ser verificado através de uma barreira de sincronização onde uma variável global contendo, por exemplo, a soma de todos os termos até o momento, é atualizada.

Sua tarefa neste EP será implementar um algoritmo paralelo que calcule uma aproximação para o valor de π. Seu programa deve parar quando a diferença entre o valor de π em duas iterações consecutivas for menor do que um valor `f` passado como entrada para o programa.

2 Requisitos e entregas

2.1 Fórmula para cálculo de π

Você pode implementar qualquer fórmula que calcule uma aproximação de π, desde que essa fórmula seja definida como uma série infinita. Sinta-se livre para escolher a fórmula que você quiser. Converse com amigos dos outros cursos do IME, pesquise no google, na wikipedia, etc... Escolher a fórmula é a primeira etapa do EP.

Após escolher a fórmula você precisa especificar o vetor que você utilizará na sua implementação e qual cálculo cada thread do seu programa realizará sobre cada posição do seu vetor.

Você precisará entregar um .pdf no paca apresentando qual fórmula você escolheu, com referências (livros, artigos, páginas na web) de onde você encontrou a fórmula, e uma explicação de como você vai paralelizar essa fórmula – o que será armazenado em cada posição do vetor e qual cálculo cada thread realizará em uma posição do vetor.

O .pdf precisa ser entregue no paca até as 08:00 do dia 06/05. Ele vale 1,0. A não entrega deste .pdf implicará em nota ZERO no EP (nem adianta enviar a segunda entrega). A cada hora de atraso a nota

máxima desta entrega será descontada de 0,1 (por exemplo, se você entregar entre 08:01 e 08:59, a nota máxima que você poderá tirar nesta entrega será 0,9).

O nome do arquivo .pdf deve ser `ep2-membros_da_equipe.pdf`. Por exemplo, `ep2-joao-maria.pdf`.

2.2 Threads e barreira de sincronização

O seu EP deverá criar uma quantidade de threads igual à quantidade de núcleos do computador (n) e essas threads deverão se sincronizar sempre que todas elas terminarem de encontrar n novos termos da série infinita. A cada iteração do seu programa, cada thread deve ler um termo armazenado em uma posição do vetor e atualizar esse termo. Ao fim de cada iteração uma variável global contendo o valor de π até o momento deve ser atualizada. O programa deve parar quando a diferença entre dois valores de π calculados por iterações consecutivas for menor do que o valor `f` passado como parâmetro para o seu programa.

A barreira de sincronização implementada no seu programa deve ser a barreira de disseminação ou a barreira borboleta. Você deve informar no LEIAME do seu programa qual das duas barreiras você implementou e indicar as linhas do código onde ela está implementada.

IMPORTANTE: Seu programa não pode utilizar informações pré-calculadas que facilitem o cálculo do valor de π. Por exemplo, se a fórmula que você utilizar para calcular o valor de π precisa computar números primos para cada termo, você deve encontrar os primos em tempo de execução. Utilizar uma tabela com números primos pré-calculados ou qualquer outra tabela com números pré-calculados que facilitem o cálculo dos termos implicará em nota ZERO no EP.

IMPORTANTE: Você deve implementar a barreira de sincronização. Programas que utilizem barreiras já prontas de alguma biblioteca implicarão em nota ZERO no EP.

2.3 Linguagem

Seu programa deve ser escrito em C, C++ ou java. Programas escritos em outra linguagem terão nota ZERO.

2.4 Entrada e saída

O seu programa receberá como entrada obrigatoriamente o valor de `f` e dois parâmetros opcionais: `DEBUG` e `SEQUENCIAL` (ou nenhum desses dois parâmetros será passado ou apenas um será passado)

Quando nenhum parâmetro opcional for passado na linha de comando o seu programa deverá imprimir apenas ao término da execução:

- Número de iterações, que vai ser um contador que armazena quantas vezes as threads se encontraram na barreira;
- Valor encontrado de π.

Quando a opção `DEBUG` for passada na linha de comando o seu programa deverá imprimir:

- A cada iteração, ordem com que as threads chegaram na barreira. Cada thread precisa de um identificador;
- A cada iteração, valor parcial de π;

- Ao término da execução, o número de iterações, que vai ser um contador que armazena quantas vezes as threads se encontraram na barreira;
- Ao término da execução, o valor encontrado de π.

Quando a opção `SEQUENCIAL` for passada na linha de comando o seu programa deverá calcular a fórmula de forma sequencial **sem a utilização de threads ou processos paralelos**. O programa deverá imprimir:

- Valores parciais de π calculados **a cada novo termo encontrado**;
- Ao término da execução, o valor encontrado de π.
- Ao término da execução, a quantidade de termos encontrados.

O seu programa precisa ser entregue no paca até as 08:00 do dia 20/05. Ele vale 9,0. A cada hora de atraso a nota máxima desta entrega será descontada de 1,0.

2.5 Relatório

Junto com o seu programa você deve entregar um relatório em .pdf apresentando uma análise de desempenho que você realizou no seu programa. O formato do relatório é livre mas deve conter obrigatoriamente:

- Configuração da(s) máquina(s) onde você executou o seu programa;
- Entradas (valores de `f`) que foram passadas para o seu programa;
- Quantidade de repetições que foram executadas para cada entrada;
- Gráficos comparando o desempenho, em termos do tempo de execução, do seu programa paralelo e da versão sequencial para diversas entradas. Esses gráficos precisarão conter médias aritméticas e alguma informação de dispersão como desvio padrão, intervalo de confiança, etc...;
- Explicação de como o tempo de execução foi medido.
- Explicações sobre os resultados obtidos: foi o esperado? Não foi o esperado? etc...

O relatório deve ser entregue junto com o programa no arquivo .tar.gz. Ou seja, este .tar.gz deve conter os seguintes itens:

- fonte, Makefile (ou similar), arquivo LEIAME;
- relatório em .pdf.

O nome do arquivo .tar.gz deve ser `ep2-membros_da_equipe.tar.gz` e quando desempacotado ele deve produzir um diretório contendo os itens. O nome do diretório deve ser `ep2-membros_da_equipe`. Por exemplo: `ep2-joao-maria`.

Obs.: O EP pode ser feito individualmente ou em dupla.

3 Bônus para o programa mais eficiente

Os EPs que receberem nota 10,0 passarão por uma avaliação extra para verificar qual é o mais eficiente. Os códigos serão executados em diversos cenários definidos pelo professor e aquele que rodar, em média, mais rápido devolvendo os valores mais precisos de π será considerado o mais eficiente. Os autores deste EP ganharão 1,5 extra na média final. A título de curiosidade, segue abaixo o valor de π com 100 casas decimais:

3.1415926535897932384626433832795028841971693993751058209749445923078164062862089986280348253421170679