MAC0438 – Programação concorrente – 1s2011
EP2
Data de Entrega: 14/06/2011

Prof. Daniel Macêdo Batista

1 Objetivo

O objetivo deste EP é implementar algum dos algoritmos de barreira de sincronização vistos em sala de aula.

2 Problema

A constante de Brun é calculada através da soma dos inversos dos números primos gêmeos, sendo que dois números primos p e q são gêmeos se $q = p + 2$.

Portanto, a constante de Brun pode ser escrita como

$$\left(\frac{1}{3} + \frac{1}{5}\right) + \left(\frac{1}{5} + \frac{1}{7}\right) + \left(\frac{1}{11} + \frac{1}{13}\right) + \ldots.$$

Diferente da soma dos inversos dos números primos, que diverge, a soma dos inversos dos pares de primos gêmeos converge.

A tarefa deste EP será implementar um programa que retorna a soma dos inversos dos primeiros pares de primos gêmeos da constante de Brun. A soma retornada deve ser maior do que o valor passado como parâmetro pelo usuário.

Por exemplo, se o usuário rodar:

```
./constantebrun 1.36
```

o programa pode retornar qualquer uma destas saídas:

`1.369120` (nesse caso ele terá calculado até o décimo par de gêmeos)
`1.383614` (nesse caso ele terá calculado até o décimo primeiro par de primos gêmeos)
`1.396948` (nesse caso ele terá calculado até o décimo segundo par de primos gêmeos)
`1.408059` (nesse caso ele terá calculado até o décimo terceiro par de primos gêmeos)
...

3 Requisitos

O programa deve ser multithread e cada thread deve ser responsável por calcular a soma do inversos de um par específico de primos gêmeos. Ou seja, a thread deve encontrar o par e calcular a soma dos inversos desse par.

Todas as threads do programa devem implementar uma barreira de sincronização. O objetivo da barreira é verificar o valor da soma até aquele ponto e definir qual par de primos gêmeos cada thread buscará na próxima iteração. Todos os EPs devem fazer a verificação da soma e definição dos próximos primos na barreira. EPs que não façam isso terão nota ZERO. Outras ações podem ser feitas na barreira, antes das próximas iterações começarem.

O algoritmo para implementar a barreira de sincronização deve ser algum dos vistos em sala de aula (qualquer um da seção 3.4 do livro do Andrews). O algoritmo implementado deve ser informado no arquivo `LEIAME.txt`. EPs que não utilizem nenhum dos algoritmos vistos em sala de aula terão nota ZERO. Qualquer mecanismo visto em sala de aula até o prazo de entrega do EP pode ser utilizado para auxiliar na implementação do algoritmo.

A quantidade de threads pode ser definida de forma estática ou dinâmica. No caso de definição dinâmica, o programa pode ter uma fase de adaptação na qual ele verifica quantos núcleos o computador tem.

O programa só deve receber o valor mínimo da soma como parâmetro na linha de comando. Nenhum outro valor deve ser passado. EPs que não atendam a esse requisito terão nota ZERO.

3.1 Linguagem

O programa pode ser escrito em C, C++ ou java. Programas escritos em C++ podem utilizar a biblioteca TBB (http://threadingbuildingblocks.org/).

3.2 Detalhes do programa

O programa não pode utilizar nenhuma tabela de primos pré-calculados e nem utilizar uma função que não tenha sido implementada pelos integrantes para encontrar os primos. O programa também não pode devolver sempre o mesmo valor da soma sem considerar o limite passado pelo usuário (por exemplo, devolver sempre a soma até o milésimo par). EPs que não obedeçam a essas regras terão nota **ZERO**.

4 Sobre a entrega

Você deverá entregar um arquivo .tar.gz contendo os seguintes itens:

- fonte(s);
- Makefile (ou similar);
- arquivo LEIAME.

O desempacotamento do arquivo .tar.gz **deve** produzir um diretório contendo os itens. O nome do diretório **deve** ser ep2-membros_da_equipe. Por exemplo: ep2-joao-maria.

A entrega do .tar.gz **deve** ser feita através do Paca.

O EP pode ser feito individualmente ou em grupos de até três integrantes.

5 Avaliação extra

Os EPs corretos passarão por uma avaliação extra para verificar qual é o mais eficiente. Os códigos serão executados em 3 configurações diferentes e aquele que rodar mais rápido em no mínimo 2 das 3 configurações será o vencedor. Os autores deste EP ganharão 1,0 ponto extra na média final.

Os EPs competirão contra aqueles feitos na mesma linguagem. Portanto, até 3 grupos podem ganhar 1,0 ponto extra na média final.