MC900/MO644 - Programação Paralela
Laboratório 8

Professor: Guido Araújo
Monitor: Luís Felipe Mattos

Cache Miss & False-Sharing

Neste laboratório, iremos otimizar 2 programas simples. O primeiro, que é um produto escalar entre 2 vetores, tem um problema de cache miss porque o uso da memória é ineficiente. O segundo programa é uma soma de uma lista de escalares, porém que causa false-sharing nas caches.

Para ajudar na análise, vocês devem usar o Vtune, porém ao invés de uma análise de hotspots, vocês devem usar a General Exploration Analysis. Este tipo de análise é capaz de mostrar cache misses em regiões específicas do código, mas para isso, o processador deve ser um Intel Sandy Bridge/Ivy Bridge, que contém os contadores necessários para essa análise.

No Vtune, para poder utilizar a General Exploration, vocês devem desligar uma flag do sistema, o nmi_watchdog. Para isso, basta utilizar o seguinte comando: **sudo sysctl -w kernel.nmi_watchdog=0**.

Este tipo de análise não está disponível na sala 317, porque os computadores não são Sandy Bridge/Ivy Bridge. Vocês devem utilizar seus computadores pessoais para utilizar o Vtune.

Outra alternativa é utilizar o Perf, outro programa de perfilamento. Ele vem no pacote linux-tools e já está instalado nos laboratórios do IC. Uma boa referência, com vários exemplos de execução pode ser econtrado no link:

http://www.brendangregg.com/perf.html

Os dois programas, que serão fornecidos, já estão paralelizados com Pthreads.

Enunciado

Para este exercício, os programas paralelos fornecidos devem ser otimizados para consertar os problemas que apresentam. O programa Dot-Product apresenta um uso ineficiente de memória, sendo necessária a melhor utilização da cache para melhorar o desempenho e o programa Sum-Scalar apresenta uma divisão de dados entre as threads que causa um grande aumento no tempo de execução por causa de false-sharing entre os dados dos vetores.

A entrada e a saída do programa já estão corretas nos códigos que serão fornecidos.

O programa recebe como entrada 2 valores inteiros, o primeiro do tipo long unsigned que é o tamanho do vetor e o segundo que é o número de theads a serem usadas, um exemplo a seguir:

```
10000000
4
```

Seu programa deve ter uma saída que é o resultado da computação e o tempo de execução em us:

```
20000000
419711
```

Tarefa Complementar

A tarefa complementar será resolver o programa Sum-Scalar usando a abordagem PADDING. A tarefa complementar deverá ser submetida nor-

malmente no SuSy, assim como os dois outros problemas.

Testes e Resultado

Quanto aos testes, serão 6 testes de tamanhos diferentes, 3 abertos e 3 fechados, variando de tamanhos de vetores de 10000000 valores até 100000000 valores, para os dois programas.

Submissões

O número máximo de submissões é de **10**.

Compilação

O Susy irá compilar seu programa com as seguintes flags:
-std=c99 -pedantic -Wall -lpthread -lm

Execução

A execução será feita com o seguinte comando:
./(dotproduct/sumscalar) < arqX.in