

Introdução a paralelização de aplicações com OpenMP

Rodrigo Alves Prado da Silva

¹Instituto de Computação
Universidade Federal Fluminense

ERAD RJ 2018

Quem sou eu?



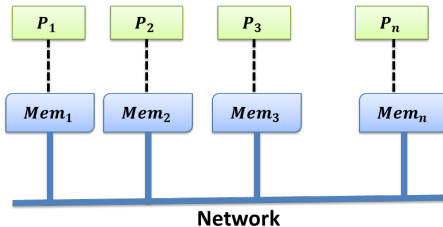
Rodrigo Silva (Foto ilustrativa)

- ▶ Bacharel em Ciência da Computação pela UFF.
- ▶ Mestrando pela mesma instituição.
- ▶ Trabalha com computação distribuída, HPC, cloud computing e paralelismo com MPI, OpenMP e instruções SIMD.

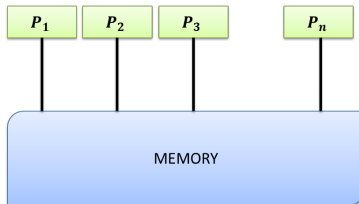
Não importa o quão rápido os computadores são. Neste momento, novas tecnologias estão sendo desenvolvidas para que eles sejam ainda mais rápidos. Nosso apetite por processamento e capacidade de memória parece insaciável.

Chapman *et al.* (2008)

Memória Distribuída vs Memória Compartilhada



Arquitetura de memória distribuída.

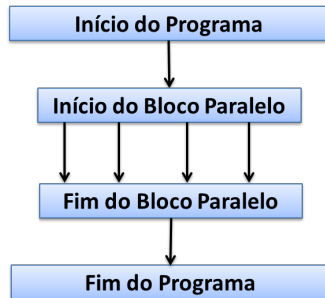


Arquitetura de memória compartilhada.

Introdução ao OpenMP

O que é OpenMP?

- ▶ Sistemas de memória compartilhada
- ▶ Diretivas
- ▶ Funções
- ▶ Variáveis de Ambiente
- ▶ Fork-Join



Algumas bibliotecas

- ▶ MPI, Pthreads, OpenMP, OpenCL, CUDA

Por que OpenMP?

Razões técnicas:

- ▶ Linguagens suportadas: Fortran e C/C++
- ▶ Mesmo código para implementação serial e paralela
- ▶ Implementação portátil e intuitiva

Por que OpenMP?

Por que deveríamos paralelizar nossos códigos:

- ▶ Quantos cores tem o seu computador?
- ▶ Os programas que você escreve aproveitam TODO o poder processamento disponível?
- ▶ Você quer ganhar desempenho?

Principais Componentes do OpenMP

► Diretivas

- *#pragma omp parallel*
- *#pragma omp master*
- *#pragma omp single*
- *#pragma omp for*
- *#pragma omp critical*
- *#pragma omp atomic*
- Etc

► Biblioteca

- *omp_get_num_threads()*
- *omp_set_num_threads()*
- *omp_get_thread_num()*

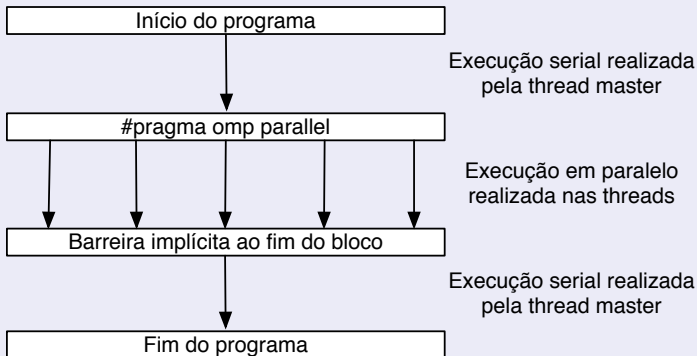
► Variáveis de ambiente

- OMP_NUM_THREADS

Expressando paralelismo

Bloco paralelo

- ▶ *#pragma omp parallel*
- ▶ Inicia bloco de código que será executado por todas as *threads* disponíveis



Exemplo

Hello World Paralelo

- ▶ A execução é inicializada na *thread master*
- ▶ A diretiva `#pragma omp parallel` inicia um bloco paralelo (*fork*)
- ▶ Cada thread executa os comandos contidos no bloco paralelo
- ▶ Barreira implícita ao fim do bloco (*join*)
- ▶ *Thread master* retoma o processamento serial
- ▶ `gcc -fopenmp -o <exe> <source.c>`

Exemplo

Hello World

Perguntas:

- 1 O que acontece na linha 11 do exemplo helloworld.c?

Exemplo

Hello World

Perguntas:

- ❶ O que acontece na linha 11 do exemplo helloworld.c?
- ❷ Como paralelizar um bloco de código maior?

Exemplo

Hello World

Perguntas:

- ❶ O que acontece na linha 11 do exemplo helloworld.c?
- ❷ Como paralelizar um bloco de código maior?
- ❸ Como preparar um código OpenMP para máquinas que não tem a biblioteca instalada?

Exemplo

Hello World

Perguntas:

- ❶ O que acontece na linha 11 do exemplo helloworld.c?
- ❷ Como paralelizar um bloco de código maior?
- ❸ Como preparar um código OpenMP para máquinas que não tem a biblioteca instalada?
- ❹ O que é escrito na tela na execução da linha 19, helloworld3.c?

Exemplo

Hello World

Perguntas:

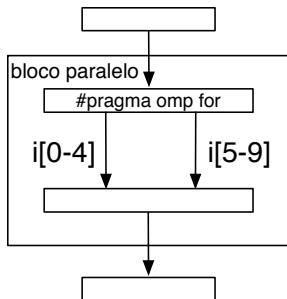
- 1 O que acontece na linha 11 do exemplo helloworld.c?
- 2 Como paralelizar um bloco de código maior?
- 3 Como preparar um código OpenMP para máquinas que não tem a biblioteca instalada?
- 4 O que é escrito na tela na execução da linha 19, helloworld3.c?
- 5 Como alterar o número de threads sem alterar o código?
 - ▶ `export OMP_NUM_THREADS=N`

Exemplo

Divisão de um vetor

- Como dividir o trabalho a ser realizado em um vetor entre as *threads* disponíveis?

```
int i[10];  
omp_set_num_threads(2);
```



Contexto dos Dados

- ▶ Contexto das threads
 - ▶ Variáveis locais
 - ▶ Variáveis globais
- ▶ Variáveis privadas
 - ▶ Private
 - ▶ Firstprivate
- ▶ Variáveis compartilhadas
 - ▶ Default
 - ▶ Shared
- ▶ Variáveis de ambiente
 - ▶ OMP_STACKSIZE
 - ▶ Default: entre 2MB e 4MB

Exemplo

Contexto de variáveis

- ▶ Inicialização da variável X
- ▶ Criação de 20 threads

Pergunta: Qual o comportamento esperado da programa?

Reduções

Redução é uma operação computacional que, aplicada a um conjunto de valores, resulta em um único valor.

Contexto de variáveis

- ▶ Soma: $[1, 2, 3, 4, 5] \rightarrow 15$
- ▶ Multiplicação: $[1, 2, 3, 4, 5] \rightarrow 120$
- ▶ Mínimo: $[1, 2, 3, 4, 5] \rightarrow 1$
- ▶ Máximo: $[1, 2, 3, 4, 5] \rightarrow 5$

Exemplo

Redução de um vetor

- ▶ Dado um vetor de N posições
- ▶ Retornar a soma dos valores contidos em cada uma das N posições

Pergunta: Como esta tarefa pode ser dividida?

Sincronização

Como estabelecer uma ordem na execução das *threads*?

Barreira

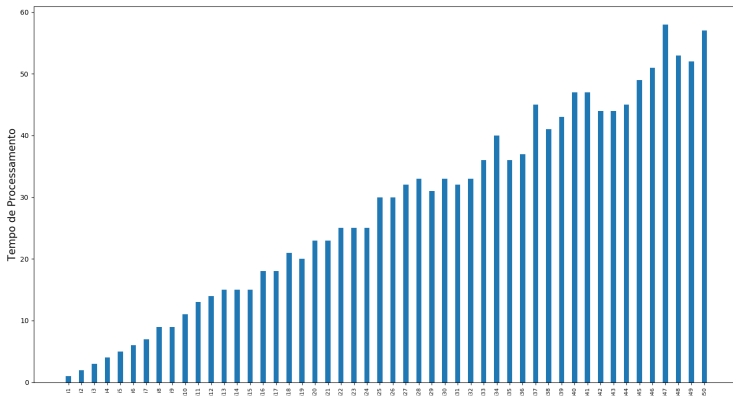
- ▶ *#pragma omp barrier*
- ▶ Todas as *threads* devem alcançar a barreira para que a execução possa seguir
- ▶ Garante consistência

Sessão crítica

- ▶ *#pragma omp critical*
- ▶ Trecho executado por somente uma *thread* por vez
- ▶ Garante exclusão mútua, evita condição de corrida

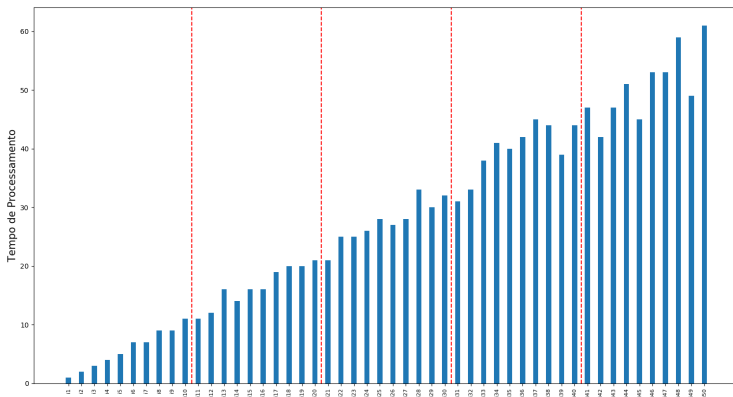
Escalonamento de Iterações

O ocorre quando o processamento das iterações não é constante?



Escalonamento de Iterações

O ocorre quando o processamento das iterações não é constante?



Escalonamento de Iterações

Escalonamento Estático

- ▶ Escalonamento default
- ▶ Chunksize default = número de iterações / número de threads
- ▶ *`#pragma omp parallel for schedule(static[,chunksize])`*

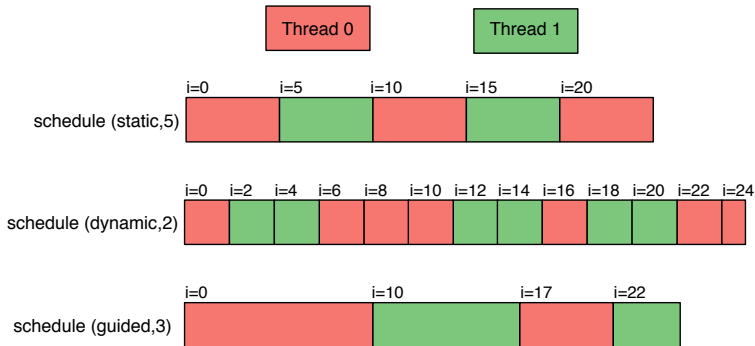
Escalonamento Dinâmico

- ▶ Chunksize default = 1
- ▶ *`#pragma omp parallel for schedule(dynamic[,chunksize])`*

Escalonamento Guiado

- ▶ Compilador define tamanho dos chunks, de forma decrescente
- ▶ Lastchunksize define o tamanho do último chunk
- ▶ *`#pragma omp parallel for schedule(guided[,lastchunksize])`*

Escalonamento de Iterações



Exemplo

Contar números primos

- ▶ Dado um valor N
- ▶ Qual o número de primos no intervalo $[1, N]$?

Perguntas:

- ▶ Como dividir este problema?

Exemplo

Contar números primos

- ▶ Dado um valor N
- ▶ Qual o número de primos no intervalo $[1, N]$?

Perguntas:

- ▶ Como dividir este problema?
- ▶ Esta solução está correta?

Exemplo

Contar números primos

- ▶ Dado um valor N
- ▶ Qual o número de primos no intervalo $[1, N]$?

Perguntas:

- ▶ Como dividir este problema?
- ▶ Esta solução está correta?
- ▶ Todas as *threads* terão o mesmo trabalho?

Outras formas de divisão de trabalho

- ▶ Como paralelizar problemas irregulares?
 - ▶ Algoritmos recursivos
 - ▶ *Loops* sem limite definido

Seções

- ▶ *#pragma omp sections*
- ▶ Define seções a serem processadas por uma única *thread*
- ▶ *Threads* executam seções sequencialmente

Tarefas

- ▶ *#pragma omp task*
- ▶ Tarefas são criadas de acordo com uma determinada condição
- ▶ *Threads* executam tarefas sequencialmente
- ▶ Padrão *Producer-Consumer*

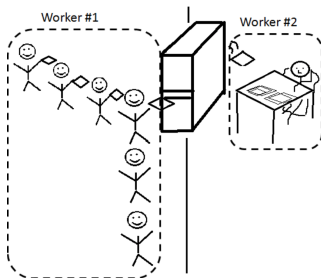
Exemplo

Padrão produtor-consumidor

- ▶ Lista é preenchida com itens produzidos em tempo de execução
- ▶ Itens da lista são consumidos
- ▶ Tempo de consumo é variável
- ▶ Tempo de produção é sempre menor que o tempo de consumo

Pergunta:

- ▶ Como dividir este problema?

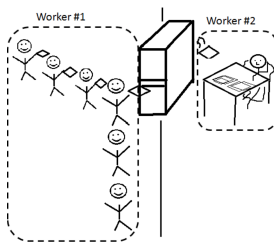


Tarefas

- ▶ Como paralelizar problemas irregulares?
 - ▶ Algoritmos recursivos
 - ▶ *Loops* sem limite definido

Tarefa

- ▶ *#pragma omp task*
- ▶ Tarefas são criadas de acordo com uma determinada condição
- ▶ *Threads* executam tarefas sequencialmente
- ▶ Padrão *Producer-Consumer*



Desafio!

Mutually Friendly Numbers

Dois números são considerados *mutually friendly* se a razão entre a soma de todos os divisores de um número e ele próprio é igual a mesma razão do outro número.

Exemplo:

$$\frac{1+2+3+5+6+10+15+30}{30} = \frac{72}{30} = \frac{12}{5}$$

$$\frac{1+2+4+5+7+10+14+20+28+35+70+140}{140} = \frac{336}{140} = \frac{12}{5}$$

30 e 140 são *mutually friendly*

Problema:

Dados um limite inferior e um limite superior, retornar todos os números *mutually friendly* contidos no intervalo.

Desafio!

Detalhes da máquina

- ▶ 12 cores lógicos
- ▶ 32 GB RAM

Soluções

- ▶ Enviar código-fonte para **rodrigo_prado@id.uff.br**
- ▶ Dúvidas?

Referências

- ▶ Chandra, Rohit. **Parallel programming in OpenMP**. Morgan Kaufmann, 2001.
- ▶ <http://openmp.org>

Dúvidas e Sugestões

- ▶ rodrigo_prado@id.uff.br

Agradecimentos

Agradeço ao Luan Teylo e o Leonardo Oliveira pelo material disponibilizado.

GitHubs:

- ▶ [*https://github.com/luanteylo/minicurso-OpenMP/*](https://github.com/luanteylo/minicurso-OpenMP/)

Meu GitHub

Esta apresentação e o material apresentado no curso estão no GitHub apresentado abaixo:

- ▶ *<https://github.com/rodrigo-prado/introducao-openmp/>*