Insper

SuperComputação

Aula 10 – Sincronização em OpenMP

2018 – Engenharia

Igor Montagner, Luciano Soares <igorsm1@insper.edu.br>

Aulas passadas

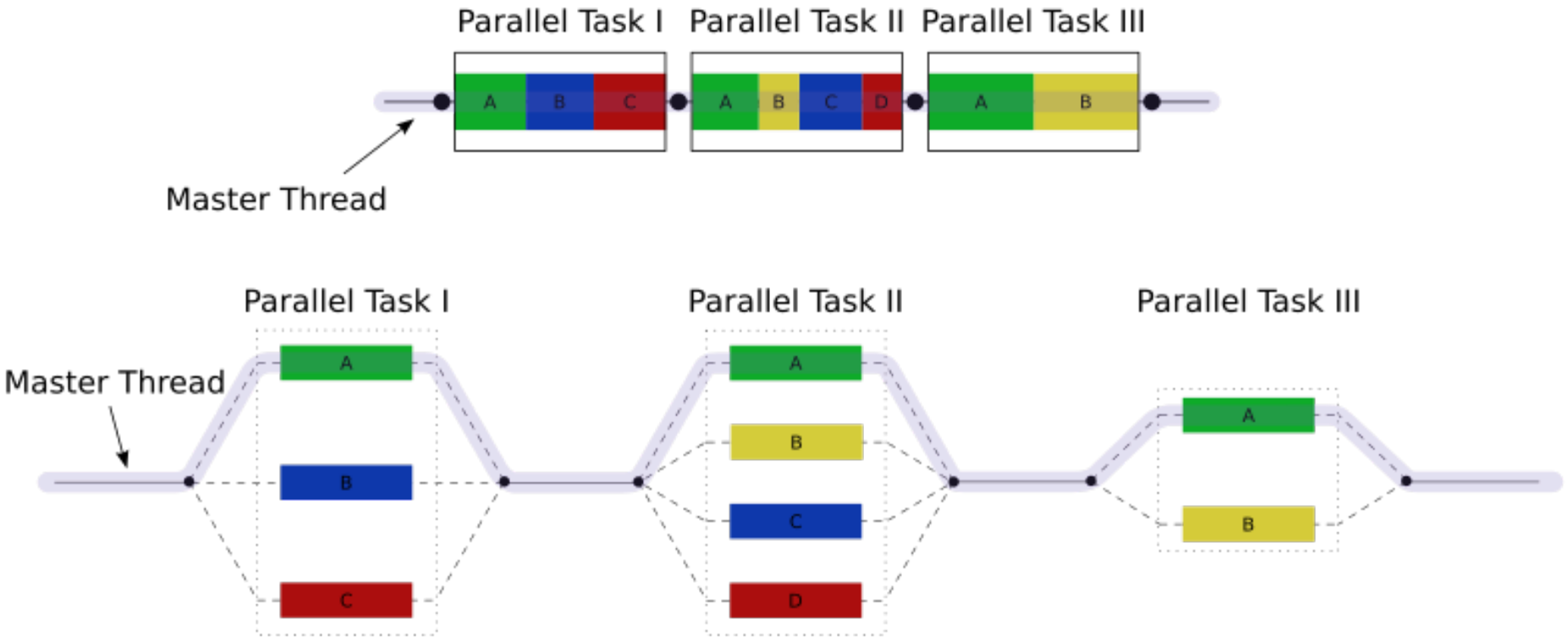


Figura: https://en.wikipedia.org/wiki/File:Fork_join.svg

Aulas passadas

1) Más práticas de programação dificultam paralelização;

2) Alguns problemas são inerentemente sequenciais;

3) *Thread-safe*

4) Mundo real exige <u>também</u> sincronização

Hoje

1) Recursos de sincronização em OpenMP

- execução das threads
- acesso a dados

2) Atividade prática: busca por termos em arquivos

Barreiras

Programa só progride após todas as threads chegarem na barreira

Barreiras - OpenMP

```
double A[big], B[big], C[big];
#pragma omp parallel
{
    int id=omp_get_thread_num();
    A[id] = big_calc1(id);
    #pragma omp barrier
    #pragma omp for
        for(i=0;i<N;i++) { C[i]=big_calc3(i,A); }
    #pragma omp for nowait
        for(i=0;i<N;i++){ B[i]=big_calc2(C, i); }
    A[id] = big_calc4(id);
}
```

Barreiras - OpenMP

```
double A[big], B[big], C[big];
#pragma omp parallel
{
    int id=omp_get_thread_num();
    A[id] = big_calc1(id);
    #pragma omp barrier
    #pragma omp for
        for(i=0;i<N;i++) { C[i]=big_calc3(i,A); }
    #pragma omp for nowait
        for(i=0;i<N;i++){ B[i]=big_calc2(C, i); }
    A[id] = big_calc4(id);
}
```

Barreira implícita no fim do *contruct* de loop for

Sem barreira implícita devido ao *nowait*

Barreira implícita devido ao fim da região *parallel*

Single – barreira implícita

A thread que chegar primeiro executa, outras só progridem quando ela acabar

```
#pragma omp parallel
{
    faz_alguma_coisa();
    #pragma omp single
    { somente_faz_uma_vez(): }
    faz_outras_coisas();
}
```

Master – sem barreira implícita

A thread id=0 executa o bloco, outras passam direto.

```
#pragma omp parallel
{
    faz_alguma_coisa();
    #pragma omp master
    { somente_faz_uma_vez(): }
    #pragma omp barrier
    faz_outras_coisas();
}
```

Necessário se quisermos
Que as outras esperem

Sections

Cada thread executa uma seção. Número é determinado no momento da compilação

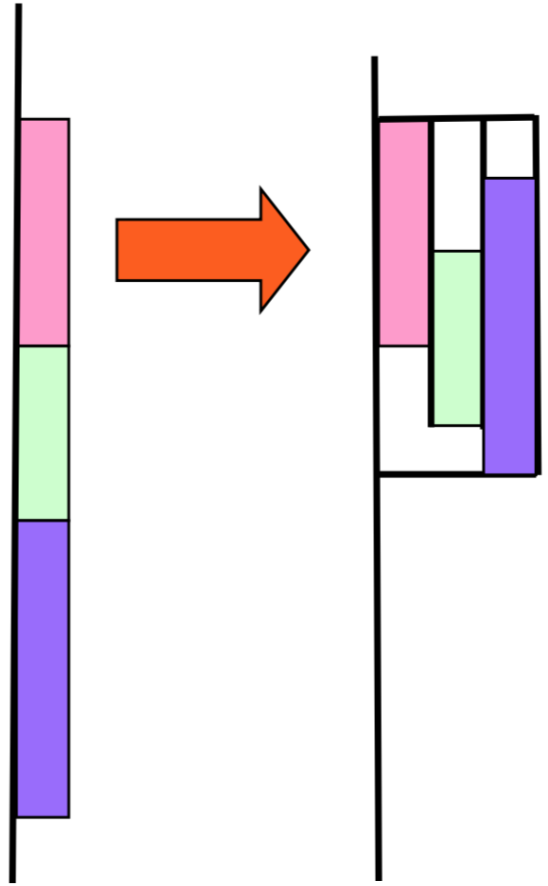
```
#pragma omp parallel
{
    #pragma omp sections
    {
        #pragma omp section
            calcula_X();
        #pragma omp section
            calcula_Y();
        #pragma omp section
            calcula_Z();
    }
}
```

Barreira implícita devido ao fim da região *parallel.*

Insper

Tarefas - tasks

- Unidade de trabalho com
  - dados
  - código
- Criada durante a execução
  - pode gerar outras tarefas
- Gerenciada pelo OpenMP
- Troca entre tarefas é cara



Serial **Paralela**

Tarefas - sincronização

- #pragma omp taskwait

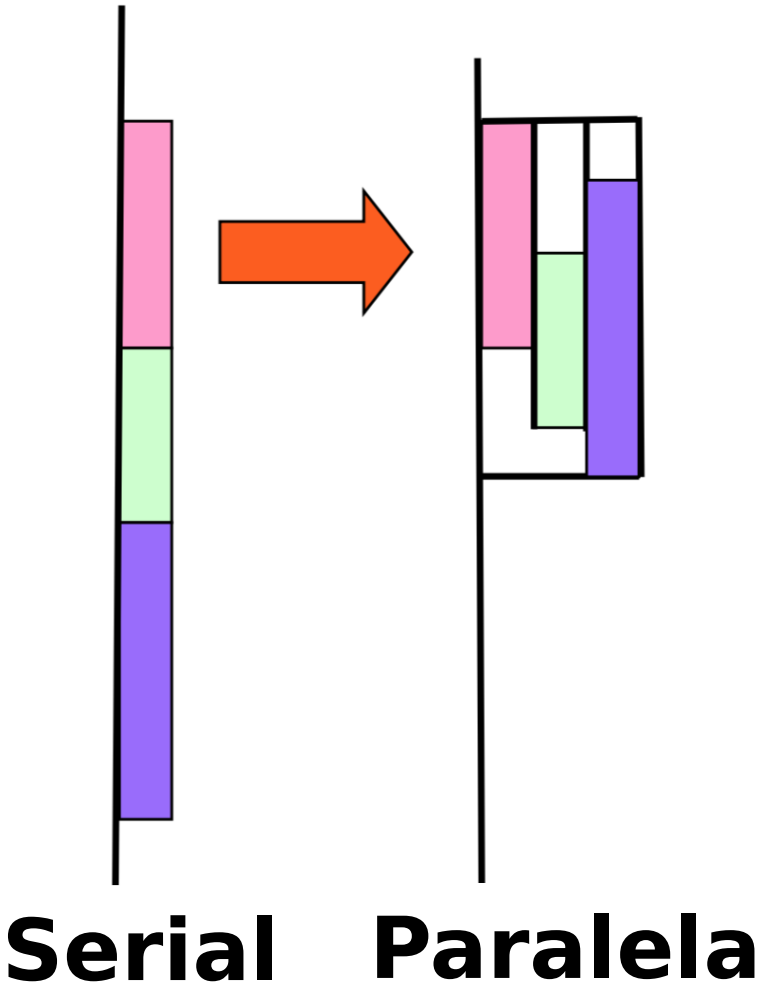
  espera pelas tarefas

  criadas pela tarefa atual

- #pragma omp taskgroup

  espera por todas tarefas

  descendentes



Tarefas - tasks

```
#pragma omp parallel
{
   #pragma omp master
   {
      p = listhead;
      while(p) {
         #pragma omp task firstprivate(p)
         {
            process(p);
         }
         p = next(p);
      }
   }
}
```

Somente um thread organiza e cria as tarefas

faz uma cópia de p quando a tarefa é empacotada

Acesso sincronizado a dados

- Suporte do OpenMP é precário
- Somente um tipo: lock
    - Alterna entre travado e destravado
    - Só quem travou pode destravar
    - Qualquer um pode travar se estiver destravado
- Granular: controla acesso a um elemento de dados

Acesso sincronizado a dados

```
#pragma omp parallel for
    for(i=0;i<NBUCKETS; i++) {
        omp_init_lock(&hist_locks[i]);
        hist[i] = 0;
    }
#pragma omp parallel for
    for(i=0;i<NVALS;i++){
        ival = (int) sample(arr[i]);
        omp_set_lock(&hist_locks[ival]);
        hist[ival]++;
        omp_unset_lock(&hist_locks[ival]);
    }
for(i=0;i<NBUCKETS; i++) {
    omp_destroy_lock(&hist_locks[i]);
```

Acesso sincronizado a dados

```
#pragma omp parallel for
    for(i=0;i<NBUCKETS; i++) {
        omp_init_lock(&hist_locks[i]);
        hist[i] = 0;
    }
#pragma omp parallel for
    for(i=0;i<NVALS;i++){
        ival = (int) sample(arr[i]);
        omp_set_lock(&hist_locks[ival]);
        hist[ival]++;
        omp_unset_lock(&hist_locks[ival]);
    }
for(i=0;i<NBUCKETS; i++) {
    omp_destroy_lock(&hist_locks[i]);
```

Um lock por elemento do histograma

Força a exclusão mútua nos elementos do histograma

Destrói lock após uso

Atividade prática

Buscar por conteúdo em cada arquivo de um diretório.

Vamos trabalhar com tasks e sincronização usando locks.

Referências

- Livros:
  - Hager, G. ; Wellein, G. **Introduction to High Performance Computing for Scientists and Engineers**. 1ª Ed. CRC Press, 2010.
- Artigos:
  - Duran, Alejandro, Julita Corbalán, and Eduard Ayguadé. "Evaluation of OpenMP task scheduling strategies." In *International Workshop on OpenMP*, pp. 100-110. Springer, Berlin, Heidelberg, 2008
- Internet:
  - https://www.youtube.com/playlist?list=PLLX-Q6B8xqZ8n8bwjGdzBJ25X2utwnoEG
  - http://www.openmp.org/wp-content/uploads/omp-hands-on-SC08.pdf
  - http://extremecomputingtraining.anl.gov/files/2016/08/Mattson_830aug3_HandsOnIntro.pdf

Insper

www.insper.edu.br