Programação Concorrente

Barreiras

Prof. Eduardo Alchieri

Barreiras

- Uma barreira é um mecanismo de sincronização que determina um ponto na execução de uma aplicação onde vários processos ou threads esperam uns pelos outros
  - Quando um processo/thread chega na barreira, executa uma operação para indicar sua chegada e entra em estado inativo
  - Depois que um certo número de processos/threads atinge a barreira, todos os processos/threads acordam (ela é vencida!)
    - Definição por um contador e um limite
      - Inicialmente zero
      - Incrementa a cada thread que atinge a barreira
      - Libera quando contador = limite

Barreiras

- Como implementar barreiras?
  - Semáforos, locks, variáveis condição ?
- Implementar barreiras usando semáforos

Primeira tentativa

```
int c;
sem barreira = 0;
```

```
atomic_inc(c);
if (c == N) sem_post(barreira);
sem_wait(barreira);
```

Barreiras

- Implementar barreiras usando semáforos

Segunda tentativa

```
int c;
sem barreira = 0;
```

```
atomic_inc(c);
if (c == N) sem_post(barreira);
sem_wait(barreira);
sem_post(barreira);
```

Barreiras

- Implementar barreiras usando semáforos

Terceira tentativa

```
int c;
sem barreira = 0;
```

```
atomic_inc(c);
if (c == N) sem_post(barreira);
sem_wait(barreira);
sem_post(barreira);
atomic_dec(c);
if (c == 0) sem_wait(barreira);
```

Barreiras

- Implementar barreiras usando semáforos

Quarta tentativa

```
int c;
sem barreira = 0;
```

```
local_c = atomic_inc(c);
if (local_c == N) sem_post(barreira);
sem_wait(barreira);
sem_post(barreira);
local_c = atomic_dec(c);
if (local_c == 0) sem_wait(barreira);
```

Barreiras

- Implementar barreiras usando semáforos

Quinta tentativa

Roleta dupla

```
int c;
sem roleta_entrada = 0, roleta_saida = 1;
```

```
local_c = atomic_inc(c);
if (local_c == N)
   sem_wait(roleta_saida);
   sem_post(roleta_entrada);

sem_wait(roleta_entrada);
sem_post(roleta_entrada);
```

```
local_c = atomic_dec(c);
if (local_c == 0)
  sem_wait(roleta_entrada);
  sem_post(roleta_saida);

sem_wait(roleta_saida);
sem_post(roleta_saida);
```

Barreiras

- Implementar uma barreira usando locks e variáveis condição

```
pthread_mutex_t barrier;  /* mutex lock for the barrier */
pthread_cond_t go;        /* condition variable for leaving */
int numWorkers;
int numArrived = 0;       /* number who have arrived */

/* a reusable counter barrier */
void Barrier() {
  pthread_mutex_lock(&barrier);
  numArrived++;
  if (numArrived == numWorkers) {
    numArrived = 0;
    pthread_cond_broadcast(&go);
  } else
    pthread_cond_wait(&go, &barrier);
  pthread_mutex_unlock(&barrier);
}
```

Barreiras

- Barreiras em C (Pthreads)
  - pthread_barrier_t (struct)
  - pthread_barrier_init (inicializar a barreira)
  - pthread_barrier_wait (atinge a barreira)

Barreiras

- Exercício
  - Somar os números de uma matriz
  - Multiplicação de matrizes