Mini EP 9: Thread Pool

Alfredo Goldman, Elisa Silva e Luciana Marques
MAC 0219-5742 – Programação Concorrente e Paralela 2021

Entrega até 10 de julho de 2021

1. Introdução

Thread pool ou grupo de *threads* é uma técnica usada para reduzir o sobrecusto em aplicações que precisam criar muitas *threads.* Isso é feito através do conceito de tarefas e um grupo de *threads* que é reutilizado. O grupo de *threads* funciona através de uma fila que mantém as tarefas a serem executadas e as *threads* consomem tarefas dessa fila até receberem uma tarefa de encerrar a thread.

Neste mini EP vocês irão terminar de implementar um programa que demonstra como grupos de *threads* funciona.

Não iremos implementar neste momento o suporte a tarefas complexas que podem pausar sua execução para um retorno posterior (*yield*).

2. Tarefas

Baixe do eDisciplinas o arquivo **"**threadpool.c". Você deve ler ele e completar a implementação do grupo de *threads*.

Vocês podem modificar o programa como bem quiserem e provavelmente vão usar os **mutexes** do pthreads para desenvolver a solução.

Segue um exemplo da saída do programa completo modificado para gerar apenas 9 tarefas e rodar em 3 threads:

```
Adding 0th task to sleep 1
Adding 1th task to sleep 4
Adding 2th task to sleep 5
Adding 3th task to sleep 8
Adding 4th task to sleep 7
Adding 5th task to sleep 2
Adding 6th task to sleep 6
Adding 7th task to sleep 1
Adding 8th task to sleep 4
Thread 0 Running task sleep 1s
Thread 2 Running task sleep 5s
Thread 1 Running task sleep 4s
Thread 0 Running task sleep 8s
Thread 1 Running task sleep 7s
Thread 2 Running task sleep 2s
Thread 2 Running task sleep 6s
Thread 0 Running task sleep 1s
Thread 0 Running task sleep 4s
Thread 1 exiting
Thread 2 exiting
Thread 0 exiting
```

3. Entrega

Adicione no início do programa seu nome e número USP e envie o arquivo do programa completo no eDisciplinas.

Entrega até 10 de julho de 2021.

4. Critério de Avaliação

Os Mini EPs usam um critério de avaliação binária (ou 1 ou 0). Para tirar 1 envie o arquivo conforme especificado no eDisciplinas. Não será avaliada a qualidade do código, mas tenha zelo.

Vale reforçar parágrafo II do artigo 23 do **Código de Ética da USP**:

> Artigo 23 - É vedado aos membros do corpo discente e demais alunos da Universidade:
> [...]
> II. lançar mão de meios e artifícios que possam fraudar a avaliação do desempenho, seu ou de outrem, em atividades acadêmicas, culturais, artísticas, desportivas e sociais, no âmbito da Universidade, e acobertar a eventual utilização desses meios.

Mini EPs plagiados receberão nota 0.

Se tiver dúvidas, envie uma mensagem no fórum do curso ou envie e-mails para elisa@silva.moe, lucianadacostamarques@gmail.com ou gold@ime.usp.br com *[miniEP9]* no assunto do e-mail. Divirta-se!

5. Extra

Uma dica para implementar uma função que bloqueia em certas condições é dar unlock condicional. Então considere que a função A deu o lock, se ela tiver que bloquear a próxima chamada de A até que uma função B seja chamada, ela não dá unlock e quando B for chamada, ela dá o unlock do mutex de A, desbloqueando eventualmente uma chamada futura da função A.

Pesquisem na internet sobre thread pool e mutexes. Modifiquem partes do programa como número de tarefas geradas ou tamanho máximo da fila para auxiliar no desenvolvimento do programa.