Sistemas Paralelos Distribuídos

EP 1

Henrique Lisboa de Sousa

Neste trabalho, exercitei os conteúdos relacionados à sincronização de threads utilizando a biblioteca pthreads, mutex e variáveis de condição. Nele, deveríamos receber como entrada várias threads a serem criadas, dos tipos 1,2 ou 3, que possuem um tempo específico para executarem sozinhas e outro tempo para ser executado em trio. Uma estrutura trio deveria permitir que apenas threads que já executaram seu tempo solo pudessem entrar, considerando que pode existir apenas um trio por vez e que um novo trio só é formado quando todas as threads do trio atual terminarem seu tempo de execução em trio.

Para isso foi proposta a seguinte solução:

Struct trio_t:

Struct que representa um trio de threads. Possui três variáveis:

Um mutex lock, que será usado para garantir que as operações realizadas na estrutura sejam feitas de forma exclusiva;

Um vetor types de tamanho 3, que guarda o tipo das threads (1, 2 ou 3) que estão no trio em questão;

Um contador count, que armazena quantas threads estão atualmente no trio;

Uma variável cond, que será usada para implementar a condição de espera das threads.

Função init_trio:

Função que inicializa a estrutura trio_t. Ela recebe um ponteiro para a estrutura e utiliza as funções pthread_mutex_init e pthread_cond_init para inicializar os campos lock e cond, respectivamente. Além disso, o campo count é inicializado com 0.

Função trio_enter:

Função que faz com que uma thread entre em um trio. Ela recebe um ponteiro para a estrutura trio_t e o tipo da thread (1, 2 ou 3). A função utiliza um mutex lock para

garantir que as operações realizadas na estrutura sejam feitas de forma exclusiva. Em seguida, ela entra em um loop while que verifica se a quantidade de threads no trio é igual a 3 ou se a thread do tipo passado por argumento já está presente no trio. Caso uma dessas condições seja satisfeita, a thread aguarda na variável cond até que uma outra thread invoque a função pthread_cond_broadcast, sinalizando que o trio está disponível novamente. Caso contrário, a thread é adicionada ao trio e o contador count é incrementado. Se o trio já estiver completo (count == 3), a função sinaliza isso com a função pthread_cond_broadcast e aguarda até que todas as três threads estejam presentes no trio. Por fim, o mutex lock é liberado.

Função trio_leave:

Função que faz com que uma thread saia de um trio. Ela recebe um ponteiro para a estrutura trio_t e o tipo da thread (1, 2 ou 3). A função utiliza um mutex lock para garantir que as operações realizadas na estrutura sejam feitas de forma exclusiva. Em seguida, ela remove a thread do tipo passado por argumento do trio, decrementa o contador count e, caso o contador seja igual a 0, sinaliza com a função pthread_cond_broadcast que o trio está novamente vazio. Por fim, o mutex lock é liberado.

Função thread_start:

Função que as threads executam. Ela recebe um argumento que contém o id da thread, o tipo da thread (um número de 1 a 3), o tempo que a thread deve gastar em sua seção crítica (tsolo) e o tempo que o trio deve gastar em sua seção crítica (ttrio). A função usa a função spend_time para imprimir uma mensagem indicando que a thread começou sua seção crítica (com o parâmetro "S") e o tempo que a thread irá gastar nessa seção. Em seguida, a thread chama a função trio_enter para entrar em um trio. Depois que o trio estiver completo, a função spend_time é chamada novamente para imprimir uma mensagem indicando que o trio começou sua seção crítica (com o parâmetro "T") e o tempo que o trio irá gastar nessa seção. Por fim, a função trio_leave é chamada para a thread sair do trio.

Função main:

A função principal do programa é responsável por criar as threads e iniciar sua execução. Ela lê os dados de entrada, cria as threads usando pthread_create e

passa os argumentos para a função thread_start. A função principal também espera que as threads terminem sua execução usando pthread_join.

O programa depende da biblioteca padrão do C (stdio.h e stdlib.h), da biblioteca pthread.h para criação e manipulação de threads e da biblioteca bool.h para uso de variáveis booleanas.