Laboratório 5
Sincronização por condição usando variáveis de condição

Programação Concorrente (ICP-361)
Prof. Silvana Rossetto

[1]IC/CCMN/UFRJ

Introdução

O objetivo deste Laboratório é introduzir o mecanismo de **sincronização por condição** usando **variáveis de condição** da biblioteca Pthread.

Atividade 1

Objetivo: Mostrar um exemplo inicial de uso de **variáveis de condição** para controlar a ordem de execução das threads de um programa.

Roteiro:

1. Abra o arquivo `hellobye.c` e identifique qual é o requisito lógico/condicional da aplicação (qual é a ordem de impressão requerida para as expressões HELLO e BYEBYE). Acompanhe a explanação da professora.
2. Execute a aplicação várias vezes e verifique se o requisito lógico é sempre cumprido.

Atividade 2

Objetivo: Avaliar a implementação de uma aplicação que tem requisitos de ordem de execução das threads.

Descrição: O programa `exemplo.c` foi implementado por um colega em uma atividade de laboratório de uma edição anterior da disciplina. O roteiro foi o seguinte:

Implemente um programa com 5 threads: A thread 1 imprime a frase "Oi Maria!"; A thread 2 imprime a frase "Oi José!"; A thread 3 imprime a frase "Sente-se por favor."; A thread 4 imprime a frase "Até mais José!"; A thread 5 imprime a frase "Até mais Maria!".

As threads devem ser criadas todas de uma vez na função main. As regras de impressão das mensagens (execução das threads) serão: (i) A ordem em que as threads 1 e 2 imprimem suas mensagens não importa, mas ambas devem sempre imprimir suas mensagens antes da thread 3. (ii) A ordem em que as threads 4 e 5 imprimem suas mensagens não importa, mas ambas devem sempre imprimir suas mensagens depois da thread 3.

Roteiro:

1. Leia o programa e verifique se os requisitos foram atendidos.
2. Acrescente mensagens de log para acompanhar a execução do programa e confirmar que a ordem de execução das threads está sendo controlada.

Atividade 3

Objetivo: Experimentar o padrão de **sincronização coletiva** (barreira).

Roteiro:

1. Abra o arquivo `barreira.c`. Ele apresenta uma implementação do padrão barreira e um exemplo simples de uso. Acompanhe a explanação da professora.
2. Execute o programa e verifique seus resultados. **As threads estão executando de forma sincronizada, concluindo uma interação para então iniciar outra?**
3. Descomente a linha 44 (que chama a 'barreira'). Execute novamente o programa e avalie os resultados.

Atividade 4

Objetivo: Retomar exercício da aula anterior e implementar um requisito condicional de execução das threads.

Roteiro:

1. Revisite o programa `soma-lock-atom.c` avaliado no Lab4.
2. Altere a implementação da thread `extra` de modo a garantir que ela imprima todos os **20** primeiros valores de `soma` que são múltiplos de 10. O objetivo é fazer a thread `ExecutaTarefa` pausar sua execução quando um múltiplo de 10 é alcançado e somente continuar depois que o seu valor for impresso.
3. Execute o programa e confirme que todos os requisitos foram atendidos.

Entrega do laboratório Disponibilize os códigos implementados na **Atividade 4** em um ambiente de acesso remoto (GitHub ou GitLab). Use o formulário de entrega desse laboratório para enviar o link do repositório do código implementado e responder às questões propostas.