Aula Prática 2007 - 02

Linux Pthreads
e
Sincronização com Semáforos

Disciplina: INF01151
Prof. Dr. Cláudio Fernando Resin Geyer
Monitor: Eder Stone Fontoura

Pthreads

- Objetivo
  - apresentar na prática o funcionamento de threads e do mecanismo de sincronização semáforos
- Roteiro
  - Threads
  - Pthreads
  - Semáforos

Threads

- Processo – é uma unidade de gerenciamento de recursos que inclui espaço de endereçamento e uma ou mais threads
  - fork() - cria cópia do processo em execução e prossegue execução de forma independente
  - dados do programa idênticos na criação do novo processo
  - o compartilhamento de recursos entre atividades relacionadas é complicado e dispendioso

Threads

- Thread – fluxos de execução associados aos processos
  - morre quando o processo que a criou morrer
  - compartilha os recursos do processo ao qual pertence
  - algumas situações propiciadas pelo compartilhamento:
    - uma thread pode fechar um arquivo que outra está utilizando
    - dados globais são compartilhados e os acessos a eles devem ser sincronizados

Threads

- Threads *versus* Processos
  - criação de uma nova thread é menos oneroso do que criar um novo processo
  - chaveamento para uma thread diferente dentro do mesmo processo é menos oneroso do que chavear entre threads pertencentes a processos diferentes
  - threads dentro de um processo podem compartilhar dados e outros recursos eficientemente quando comparado com processos

Pthreads

- POSIX (Portable Operating Systems Interface) é um conjunto de normas definidas pela IEEE para a interface de programação com sistemas operacionais
- uma dessas normas (IEEE POSIX 1003.1c std) define uma interface de programação para threads – API pthreads.h
- implementações que seguem essa norma são chamadas de Pthreads
- essa interface define um conjunto de funções e tipos em C que permitem a manipulação de threads

Pthreads - Criação

- **int pthread_create(**pthread_t * thread, attributes * attr, void * (*function)(void *), void * arg**)**
  - cria e dispara a execução de uma thread
  - **thread** -> um ponteiro para a thread criada
  - **attr** -> ponteiro para atributos de criação da thread
  - **function** -> nome da função que será chamada
  - **arg** -> argumento que será passado para a função
  - retorno -> maior que zero indica erro ou zero caso contrário

Pthreads - Manipulação

- **void pthread_exit(**void *retval**)**
  - termina a execução da thread que chamou a função
    - **retval** -> ponteiro para o valor de retorno da thread
- **int pthread_join(**pthread_t th, void **thread_return**)**
  - suspende a execução da thread que chamou a função até que a thread identificada por th termine a sua execução
  - **thread_return** -> contém o valor de retorno da thread
  - retorna um valor maior que zero indicando erro ou zero caso contrário

Semáforos

- variáveis acessadas somente através de duas primitivas atômicas:
  - P e V
- uso
  - exclusão Mútua, controle de recursos com n instâncias
- API especificada por outra norma POSIX (IEEE Std 1003.1b-1993)
- implementada em semaphore.h

Semáforos – Criação e Manipulação

- **int sem_init**(sem_t *sem, int pshared, unsigned int value)
  - inicializa o semafóro apontado por sem
  - **pshared** -> indica se o semáforo será compartilhado por mais de um processo
  - **value** -> determina o valor inicial do semáforo
- **int sem_wait**(sem_t * sem) “Primitiva P”
  - suspende a thread que chamou a função até que o valor do semáforo seja maior que zero
- **int sem_post**(sem_t * sem) “Primitiva V”
  - incrementa atomicamente o valor do semáforo
- **int sem_destroy**(sem_t * sem)
  - destrói o semáforo; nenhuma thread deverá estar aguardando

Exercícios

1. Criar um programa em C com 5 threads. Cada thread deve imprimir a mensagem “Sou a thread X”, onde X corresponde ao índice de criação da thread que deve ser passado por parâmetro para a função que a thread executará.
2. O programa ap2_ex2.c efetua uma sequência de débitos e créditos. Considerando os possíveis problemas causados pela concorrência apresentados em aula responda:
   a) O programa apresenta algum problema relacionado a concorrência? Qual?
   b) Caso tenha algum problema, apresente uma versão do programa, com o problema solucionado.

Exercícios

3. O programa ap2_ex3.c apresenta um programa com leitores/escritores. Alterar o programa para apresentar uma solução para o problema com prioridade para os leitores.
   - Obs: O programa já está parcialmente implementado. Somente a função leitor necessita alteração para solucionar o problema. A função leitor apresenta alguns comentários para facilitar a alteração do programa.