Sistemas Operacionais II N

Race Conditions and POSIX threads

Comunicação entre Processos

- Via arquivos
- Pipes
- Memória Compartilhada

  http://www.cs.cf.ac.uk/Dave/C/node27.html
- Mensagens (sockets)
- RPC - Chamadas de Procedimento Remoto

Conteúdo de Hoje

- Sincronismo em IPC
  - Indeterminismo
  - Condições de Corrida
  - Seções Crríticas
  - Condições para Concorrência
- Threads

Comunicação - IPC

- como passar informações entre processos?
- como fazer sincronismo por dependência de dados?
- como evitar interferências indesejáveis?

Comparilhamento de Memória

- Para comunicarem-se, processos usam:

memória

- ou com segmentos de dados compartilhados
- ou como threads

Arquivos, pipes, etc…

Threads - Processos leves

São linhas de execução dentro de um mesmo processo

Cada processo

- um único segmento de código
- um único segmento de dados
- um único descritor de processo no kernel

Cada thread

- uma pilha própria
- PC,SP e registradores próprios

Quem gerencia, bloqueia threads???

Indeterminismo na Concorrência

A ordem na qual as instruções são executadas não é determinística em um programa concorrente, mesmo em uma máquina monoprocessada.

Condições de Corrida

- "O resultado da computação depende da ordem em que as instruções são executadas"
- A ordem não é determinística com T.S.
- Condições de corrida devem ser evitadas!

Exemplo

Seções ou Regiões Críticas

- "Parte do código que acessa dados compartilhados e os deixa em estados intermediários inconsistentes"
- Garantir exclusão mútua: somente um processo pode executar dentro da seção (ou região) crítica ao mesmo tempo

Condições para Concorrência

1. Dois processos não podem executar dentro da região crítica ao mesmo tempo
2. Não pode haver nenhuma suposição sobre a velocidade de execução ou número de CPUs.
3. Nenhum processo fora da região crítica pode bloquear outro processo
4. Não pode haver espera "infinita" para entrar na região crítica

Primitivas

- diversos mecanismos para atingir esses objetivos