Resumo SO pt2

Concorrência: é a condição de um sistema no qual múltiplas tarefas estão logicamente ativas em um determinado momento

Paralelismo: é a condição de um sistema no qual múltiplas tarefas estão realmente sendo executadas ao mesmo tempo.

Execução não paralela: alternância das tarefas, não são executadas ao mesmo tempo

Execução paralela: tarefas executadas ao mesmo tempo

Vantagens da programação concorrente: aplicações envolvendo grande quantidade de cálculos, aplicações criticas (deadlines), melhor uso de recursos computacionais (processador e memoria) e alta capacidade de processamento

Dificuldades programação concorrente: Desenvolvimento, corretude, debug

Ordem em que processos concorrentes executam não é determinística

**Concorrencia:** Processos cooperativos podem afetar outros processos em execução ou por ser eles afetados. Processos cooperativos pode: compartilhar um espaço de endereçamento lógico, compartilhar dados através de arquivos ou mensagens.

Processos produtor e consumidor:

Um recurso é compartilhado entre dois processos, o produtor que insere dados no buffer e o consumidor que retira dados do buffer

Um seção crítica é um segmento de código em que o processo pode estar: alternando variáveis comuns, atualizando uma tabela, gravando um arquivo...

Quando um processo estiver executando sua seção crítica, NENHUM, outro processo deve ter autorização para fazer o mesmo. Dois processos não podem estar simultaneamente nas seções críticas.

Cada processo deve solicitar permissão para entrar em sua seção crítica (protocolo da seção crítica)

Uma solução para o problema da seção crítica deve satisfazer os quatro requisitos a seguir:

-- Somente um processo por vez acessa a região critica

-- Ser independente do número e velocidade de CPU’s

--Um processo fora da região crítica não bloqueia outro processo

--Nenhum processo espera indeterminadamente para acessar a região critica

Soluções região critica:

Soluções de algoritmo (algoritmo de lamport), hardware (inibição de interrupções) e SO (semáforos, variáveis condicionais, mutex)

Hardware de sincronização: Muitos sistemas de computação modernos fornecem instruções, de hardware especiais que nos permitem testar e modificar o conteúdo de uma palavra ou trocar os conteúdos de duas palavras atomicamente: TestAndSet: testa e modifica um valor de uma variável

Swap: Muda o valor de uma variável em uma única instrução

Abstração: muito usada para proteger regiões críticas, que evita a espera ocupada

**Mecanismos de sincronização de processos**

São variáveis que não armazenam valores específicos, mas representam condições que podem ser aguardadas por alguns processos.

Se um processo está aguardando uma condição, ele é inserido em uma fila de espera até que a condição seja verdadeira

As variáveis condicionais evitam espera ocupada

**Monitores**

Um monitor consiste de: recurso compartilhado, conjunto de variáveis ao monitor.

Um conjunto de procedimentos que permitem o acesso a essas variáveis

Um mutex ou semáforo para controlar a exclusão mutua. Cada procedimento de acesso ao recurso deve obter o semáforo antes de iniciar e liberar o semáforo ao concluir.

Um invariante (condição sobre as variáveis internas do monitor) sobre o estado interno do recurso