GIOVANNI ALEIXO DE SOUZA
RONNI APARECIDO FERNANDES DE OLIVEIRA

ATIVIDADE INTEGRADORA [AVA3]:

SISTEMAS OPERACIONAIS

Londrina
2023

GIOVANNI ALEIXO DE SOUZA
RONNI APARECIDO FERNANDES DE OLIVEIRA

ATIVIDADE INTEGRADORA [AVA3]:

SUBTÍTULO DO TRABALHO, SE HOUVER

Atividade Integradora da máteria de Sistemas Operacionais apresentado ao Curso de Tecnológo Análise e Desenvolvimento de Sistema do Centro Universitário Filadélfia – UniFil.

Orientador: Prof. Dr. Kleber Marcio de Souza

Londrina
2023

As condições de corrida, são situações indesejáveis que podem ocorrer quando múltiplos processos ou threads tentam acessar e manipular recursos compartilhados de forma concorrente. Em um sistema operacional, essas condições podem resultar em resultados inconsistentes ou inesperados devido à execução não determinística e não coordenada das operações.
Existem três condições de corrida comumente encontradas:

- Condição de corrida na leitura-gravação, onde ocorre quando um processo lê o valor de um recurso compartilhado enquanto outro processo o está modificando. Isso pode levar a leituras inconsistentes ou incorretas.
- Condição de corrida na gravação-gravação, onde ocorre quando dois ou mais processos tentam gravar simultaneamente em um recurso compartilhado. Nesse caso, o resultado final depende da ordem de execução dos processos, levando a inconsistências.
- Condição de corrida na leitura-leitura, onde ocorre quando dois ou mais processos tentam ler simultaneamente um recurso compartilhado. Embora essa condição não cause problemas funcionais, pode resultar em leituras incorretas se um processo estiver modificando o recurso ao mesmo tempo.

Regiões Críticas, são seções de código em um programa que acessam recursos compartilhados e, portanto, estão sujeitas a condições de corrida. A ideia é garantir que apenas um processo ou thread tenha acesso exclusivo a uma região crítica em um determinado momento, evitando assim problemas de inconsistência.
Para garantir a exclusividade no acesso, geralmente são aplicadas as seguintes condições para uma região crítica:

- Mutual Exclusion: garante que apenas um processo ou thread esteja executando a região crítica por vez.
- Progress: garante que, se nenhum processo estiver executando uma região crítica e algum processo quiser entrar nela, apenas os processos não impedidos de entrar podem tomar uma decisão sobre qual processo deve entrar.
- Bounded Waiting: garante que haja um limite no número de vezes que outros processos podem entrar na região crítica após um processo ter solicitado acesso.

Exclusão Mútua, é um mecanismo que garante que apenas um processo ou thread possa executar uma região crítica por vez. Isso evita condições de corrida, assegurando que os recursos compartilhados sejam acessados de forma controlada e consistente.
Existem várias abordagens para implementar a exclusão mútua, incluindo:

- Semáforos: um semáforo é uma variável inteira usada para controlar o acesso a recursos compartilhados. Ele pode ser usado para bloquear e desbloquear o acesso à região crítica.
- Mutex (Mutual Exclusion Lock): é um objeto que atua como uma chave para uma região

Os mecanismos mencionados acima são conceitos fundamentais na área de sistemas operacionais e são aplicados em diversos sistemas operacionais. Eles são implementados em sistemas operacionais tanto para desktops quanto para servidores.
Sistemas operacionais Windows, suporta esses mecanismos por meio de APIs como semáforos e variáveis de condição.
O sistema operacional Linux, possui suporte embutido para esses mecanismos, fornecendo APIs como pthreads (biblioteca de threads POSIX) e syscalls como semáforos e mutexes.
O macOS, sistema operacional desenvolvido pela Apple, é baseado no kernel BSD do Unix e suporta mecanismos como mutexes e semáforos.