UNIVERSIDAD DE GUADALAJARA

Centro Universitario de Ciencias Exactas e Ingenierías

Departamento de Ciencias Computacionales

6to semestre

Tema: Filósofos, productor y lectores

Materia:
Seminario de Sistemas Operativos

Docente:
Violeta del Rocío Becerra Velázquez

PRESENTA:
Daniel Martínez Martínez

Código:
217565958

Carrera:
Ingeniería en Computación.

Sección:
D02

Fecha de entrega:
Domingo, 26 de Marzo de 2023

Índice

Filósofos, productor y lectores

La cena de los filósofos

Es un problema clásico de sincronización dentro de la informática, el cual ilustra los desafíos que se presentan cuando varios procesos compiten por un conjunto de recursos compartidos. El problema consiste en lo siguiente:

"Cinco filósofos están sentados alrededor de una mesa redonda, y cada uno tiene un plato de comida y un tenedor a su lado. Entre cada dos platos hay un tenedor. El problema es que cada filósofo necesita dos tenedores para comer, uno a su izquierda y otro a su derecha, y solo puede usar un tenedor a la vez. El objetivo es diseñar un algoritmo que permita que todos los filósofos coman sin que se produzca un interbloqueo o deadlock (situación en la que los procesos quedan atrapados y no pueden avanzar)."

Una solución común para el problema de la cena de los filósofos es el llamado "protocolo del camarero". En esta solución, se introduce un "camarero" que actúa como un coordinador y controla el acceso a los tenedores. El protocolo se puede resumir en los siguientes pasos:

1. El camarero coloca todos los tenedores en la mesa y los mantiene bajo su control.
2. Cuando un filósofo quiere comer, debe pedir permiso al camarero.
3. Si el camarero tiene dos tenedores disponibles, se los entrega al filósofo.
4. Si el camarero no tiene dos tenedores disponibles, el filósofo debe esperar hasta que el camarero reciba los tenedores de otro filósofo.
5. Cuando el filósofo ha terminado de comer, devuelve los tenedores al camarero, quien los vuelve a colocar en la mesa.
6. Con este protocolo, se asegura que nunca habrá más de cuatro filósofos comiendo al mismo tiempo, evitando así la posibilidad de interbloqueos. Esta solución puede ser implementada mediante programación concurrente, utilizando por ejemplo semáforos o monitores.

Producto-Consumidor

El problema del productor-consumidor es un clásico problema de sincronización en el que un conjunto de procesos productores producen datos que deben ser consumidos por un conjunto de procesos consumidores. El objetivo es asegurar que los productores no produzcan datos cuando no hay consumidores disponibles para consumirlos, y que los consumidores no intenten consumir datos que aún no han sido producidos.

Una de las más comunes es utilizar un buffer compartido entre los procesos productores y los consumidores. Este buffer actúa como un espacio temporal de almacenamiento de los datos producidos por los productores antes de ser consumidos por los consumidores.

Para implementar esta solución, se utilizan semáforos para controlar el acceso al buffer compartido. Se utiliza un semáforo de conteo para llevar la cuenta del número de espacios libres en el buffer, y otro semáforo de exclusión mutua para evitar que varios procesos intenten acceder al buffer al mismo tiempo.

Esta solución asegura que los productores no produzcan datos cuando no hay espacio disponible en el buffer, y que los consumidores no intenten consumir datos que aún no han sido producidos.

Lectores-Escritores

El problema de lectores-escritores es otro clásico problema de sincronización que surge cuando varios procesos deben acceder a un recurso compartido, en el que algunos procesos sólo leen el recurso, mientras que otros procesos lo escriben. El objetivo es garantizar la exclusión mutua entre los procesos que escriben para evitar la corrupción de datos, mientras se permite que varios procesos puedan leer el recurso simultáneamente sin bloquearse entre sí.

Una de las más comunes es utilizar una estructura de bloqueo (como un semáforo o un mutex) para garantizar la exclusión mutua entre los procesos escritores y un contador de lectores para permitir que múltiples procesos puedan leer el recurso simultáneamente.

Cuando un proceso escritor quiere escribir en el recurso, primero adquiere el bloqueo para garantizar que ningún otro proceso esté escribiendo en ese momento. Luego escribe los datos en el recurso y libera el bloqueo.

Cuando un proceso lector quiere leer el recurso, primero incrementa el contador de lectores. Si es el primer lector, adquiere el bloqueo para garantizar que ningún proceso escritor esté escribiendo en ese momento. Luego lee los datos del recurso y decrementa el contador de lectores. Si es el último lector, libera el bloqueo.

¿En qué consiste el problema de la concurrencia?

La concurrencia de procesos se refiere a las situaciones en las que dos o más procesos puedan coincidir en el acceso a un recurso compartido o, dicho de otra forma, que requieran coordinarse en su ejecución. Para evitar dicha coincidencia, el sistema operativo ofrece mecanismos de arbitraje que permiten coordinar la ejecución de los procesos.

El problema de la concurrencia surge cuando varios procesos o hilos en ejecución acceden de manera simultanea a recursos compartidos, por ejemplo variables, bases de datos o archivos en general, siempre compitiendo por su uso.

¿Cuáles son los procesos concurrentes cooperantes?

Los procesos concurrentes cooperativos son procesos que colaboran entre sí para alcanzar un objetivo común. En otras palabras, estos procesos interactúan y comparten información con el fin de lograr una tarea en particular.

¿En qué consiste la exclusión mutua?

Es una propiedad de la sincronización de procesos que establece que dos procesos no pueden estar en la sección critica de un programa al mismo tiempo, es decir, que no pueden existir al mismo tiempo en el área del programa que engloba a los recursos que son compartidos por los procesos. La exclusión mutua es necesaria para prevenir situaciones en las que dos o más procesos intentan modificar un recurso compartido al mismo tiempo, lo que puede provocar resultados impredecibles e incluso errores en el sistema.

Defina interbloqueo

Un interbloqueo (Deadlock en inglés) es una situación donde un conjunto de procesos están bloqueados en un estado de espera de manera indefinida, sin posibilidad de continuar en su ejecución debido a que cada uno de ellos esta esperando a que otro proceso libere un recurso necesitado.

Defina Inanición

Es un problema dentro de los sistemas operativos que se da cuando un proceso no puede acceder a un recurso que necesita para continuar su ejecución. Sucede principalmente cuando varios procesos compiten por los mismos recursos, tales como el CPU, memoria, acceso a la red, etc. Este fenómeno propicia que solo unos pocos procesos sean beneficiados.

Defina excesiva cortesía

La excesiva cortesía se produce cuando un proceso espera por mucho tiempo un recurso debido a que otro proceso lo bloquea durante demasiado tiempo, incluso cuando no es necesario, para asegurarse de que ningún otro proceso pueda acceder al recurso compartido. Esto puede provocar una degradación del rendimiento del sistema, ya que los recursos compartidos bloqueados no están disponibles para otros procesos que los necesiten, lo que puede provocar cuellos de botella y retrasos en la ejecución de los procesos.

¿Qué son los Hilos?

Un hilo es un flujo independiente de instrucciones el cual puede ser programado por el sistema operativo para correr en algún momento. Desde la perspectiva de un programador, un hilo podría describirse como un procedimiento que se ejecuta de manera independiente a un programa principal.

¿Qué son los semáforos?

Son un mecanismo de sincronización entre procesos en los sistemas operativos. Son usados para evitar problemas de concurrencia en situaciones donde varios procesos o hilos tratan de acceder a los mismos recursos compartidos.

Los semáforos se implementan como una variable de contador que puede ser accedida y modificada por diferentes procesos. Se utilizan dos tipos de operaciones: "wait" (esperar) y "signal" (señal).

Cuando un proceso necesita acceder a un recurso compartido, primero intenta decrementar el contador del semáforo utilizando la operación "wait". Si el valor del contador es mayor que cero, el proceso puede acceder al recurso compartido y continúa su ejecución. Si el valor del contador es cero, el proceso se bloquea y espera a que otro proceso aumente el valor del semáforo utilizando la operación "signal".

¿Qué es lo que mejora el tener más de un núcleo?

Tener más de un núcleo en un procesador permite que el dispositivo pueda realizar varias tareas simultáneamente, lo que aumenta significativamente su capacidad de procesamiento y mejora su rendimiento. Cada núcleo puede procesar instrucciones de manera independiente y ejecutar programas en paralelo, lo que significa que el dispositivo puede realizar varias tareas al mismo tiempo sin ralentizarse.

Conclusión

En esta actividad exploramos muchos conceptos y casos de estudio que se dan dentro del contexto del uso de recursos compartidos, principalmente todos aquellos relacionados a los conflictos

derivados de la falta de la sincronización para acceder a estos. Estos problemas se dan principalmente por la falta de recursos que funjan como barrera ante los problemas de concurrencia, algunos de los cuales podrían ser los semáforos, los buffer, las banderas, entre otros.

Es importante evitar que se den problemas de este tipo, debido a que pueden mermar en gran medida el desempeño de nuestro sistema, o inclusive, podrían causar que éste deje de operar por completo.

Bibliografía

- *Concurrencia de procesos - Wiki de Sistemas Operativos*. (n.d.). https://1984.lsi.us.es/wiki-ssoo/index.php/Concurrencia_de_procesos
- *Tema 1.9 Procesos cooperativos. Threads. - Plataformas operativas de tecnolog*  *as de informaci**n - Instituto Consorcio Clavijero*. (n.d.). https://cursos.clavijero.edu.mx/cursos/147_poti/modulo1/contenidos/tema1.9.html#:~:text=Procesos%20concurrentes%3A&text=Cooperativos%3A%20puede%20afectar%20o%20ser,con%20otro%20proceso%20es%20cooperativo.
- GeeksforGeeks. (2021, December 13). *Mutual Exclusion in Synchronization*. https://www.geeksforgeeks.org/mutual-exclusion-in-synchronization/
- *Inanición (informática), the Glossary*. (n.d.). https://es.unionpedia.org/Inanici%C3%B3n_(inform%C3%A1tica)
- GeeksforGeeks. (2023, March 21). *Semaphores in Process Synchronization*. https://www.geeksforgeeks.org/semaphores-in-process-synchronization/