Universidad de Guadalajara.

Centro Universitario de Ciencias Exactas e Ingenierías.

DIVISIÓN DE TECNOLOGÍAS PARA LA INTEGRACIÓN CIBER-HUMANA.

DEPARTAMENTO DE CIENCIAS COMPUTACIONALES.

TEMA: Actividad 5

NOMBRE DEL ESTUDIANTE: Padilla Perez Jorge Daray.

NOMBRE DE LA MATERIA: Seminario de Sistemas operativos

NOMBRE DEL PROFESOR: Julio Esteban Valdés López

Table of Contents

Investigue en que consiste y las soluciones de:

La cena de los filósofos

Problema de Sincronización: El problema es un ejercicio clásico en la sincronización de procesos. Cinco filósofos se sientan alrededor de una mesa y deben compartir tenedores para comer, lo que puede llevar a condiciones de carrera y bloqueos mutuos.

Soluciones:

Por turno cíclico: En esta solución, se comienza con un filósofo que puede decidir comer si lo desea. Luego, pasa su turno al filósofo de la derecha. Este enfoque garantiza que todos los filósofos tengan la oportunidad de comer y pensar.

Monitores: Utilizando monitores (estructuras de sincronización), cada filósofo debe invocar operaciones como recoger() antes de empezar a comer y dejar() después de comer. Esto evita que dos filósofos adyacentes compitan por el mismo tenedor al mismo tiempo3.

Tensores adicionales: Para evitar el riesgo de interbloqueo, se pueden adquirir tenedores adicionales. Por ejemplo, en lugar de cinco tenedores, podríamos tener diez. Otra solución creativa sería enseñar a los filósofos a comer espaguetis con un solo tenedor

Métodos Sincronizados: Se utilizan métodos sincronizados para evitar que los filósofos se queden esperando indefinidamente por un tenedor. Los métodos coger Tenedores y dejar Tenedores gestionan el acceso a los tenedores de manera controlada.

Ciclo de Vida del Filósofo: Cada filósofo, representado por un hilo, alterna entre pensar y comer. El método run define este ciclo, utilizando los métodos de la clase Mesa para sincronizar el uso de los tenedores.

Productor-Consumidor

La idea central para resolver este problema es que ambos procesos (productor y consumidor) se ejecuten simultáneamente y se “despierten” o “duerman” según el estado del búfer:

Productor:

El productor agrega productos al búfer mientras haya espacio disponible.

Cuando el búfer se llena, el productor se pone a “dormir”.

Si el consumidor toma un producto, notifica al productor para que pueda comenzar a trabajar nuevamente.

Consumidor:

El consumidor toma productos del búfer mientras haya elementos disponibles.

Si el búfer está vacío, el consumidor se pone a “dormir”.

Cuando el productor agrega un producto, crea una señal para despertar al consumidor.

Búfer Compartido

Comencemos definiendo una estructura de datos para el búfer compartido. Puede ser un array, una cola o cualquier otra estructura que admita operaciones de inserción y extracción.

Semáforos

Utilizaremos semáforos para sincronizar los procesos. Dos tipos comunes de semáforos son:

Semáforo Binario: Puede tener dos valores (0 o 1). Se utiliza para representar la disponibilidad del búfer (lleno o vacío).

Semáforo Contador: Puede tener un valor mayor que 1. Representa la cantidad de espacios disponibles en el búfer.

Variables Compartidas

espacios_disponibles: Contador de espacios disponibles en el búfer.

elementos_en_bufer: Contador de elementos actualmente en el búfer.

Consideraciones

La exclusión mutua (mutex) garantiza que solo un proceso modifique el búfer a la vez.

Los semáforos binarios controlan la disponibilidad del búfer.

Asegúrate de evitar el interbloqueo y la inanición.

Lectores-Escritores

Solución cuando el Lector tiene Prioridad sobre el Escritor

Escritor:

El escritor solicita acceso a la sección crítica (el recurso compartido).

Si se le permite, verifica que no haya lectores presentes en la sección crítica.

Si no hay lectores, procede a escribir en el recurso.

De lo contrario, espera en la cola hasta que se cumpla la condición.

Al salir de la sección crítica, señala que está vacía.

Lector:

Los lectores solicitan acceso a la sección crítica.

Si se les permite, verifican que no haya escritores presentes.

Si no hay escritores, ingresan a la sección crítica para leer.

Si hay escritores, se ponen en cola en la exclusión mutua.

Los lectores posteriores aún pueden ingresar mientras no haya escritores presentes.

Problema de los Lectores-Escritores con Prioridad de Escritor

Variables Utilizadas:

readers: Lleva un conteo de cuántos lectores están en la sección crítica.

idle: Un semáforo que indica el número de subprocesos (lectores o escritores) en la sección crítica. Si no hay subprocesos, su valor es 1; de lo contrario, es 0.

mutex: Un semáforo que protege las variables compartidas.

readSwitch y writeSwitch: Estos son “interruptores de luz” (lightswitches) que controlan el acceso de los lectores y escritores a la sección crítica.

noReaders y noWriters: Semáforos adicionales para bloquear lectores y escritores.

Solución del Escritor:

El escritor solicita acceso a la sección crítica.

Si se le permite, verifica que no haya lectores presentes en la sección crítica.

Si no hay lectores, procede a escribir en el recurso.

De lo contrario, espera en la cola hasta que se cumpla la condición.

Al salir de la sección crítica, señala que está vacía.

Solución del Lector:

Los lectores solicitan acceso a la sección crítica.

Si se les permite, verifican que no haya escritores presentes.

Si no hay escritores, ingresan a la sección crítica para leer.

Si hay escritores, se ponen en cola en la exclusión mutua.

Los lectores posteriores aún pueden ingresar mientras no haya escritores presentes.

Preguntas.

1. ¿En qué consiste el problema de la concurrencia?

En los sistemas de tiempo compartido (aquellos con varios usuarios, procesos, tareas, trabajos que reparten el uso de CPU entre estos) se presentan muchos problemas debido a que los procesos compiten por los recursos del sistema. Imagine que un proceso está escribiendo en la unidad de cinta y se le termina su turno de ejecución e inmediatamente después el proceso elegido para ejecutarse comienza a escribir sobre la misma cinta. El resultado es una cinta cuyo contenido es un desastre de datos mezclados. Así como la cinta, existen una multitud de recursos cuyo acceso debe ser controlado para evitar los problemas de la concurrencia.

El sistema operativo debe ofrecer mecanismos para sincronizar la ejecución de procesos: semáforos, envío de mensajes, 'pipes', etc. Los semáforos son rutinas de software (que en su nivel más interno se auxilian del hardware) para lograr exclusión mutua en el uso de recursos. Para entender este y otros mecanismos es importante entender los problemas generales de concurrencia, los cuales se describen enseguida.

2. ¿Cuáles son los procesos concurrentes cooperantes?

Los procesos concurrentes cooperantes son aquellos que trabajan juntos para lograr un objetivo común mientras se ejecutan de manera simultánea. Estos procesos colaboran y se comunican entre sí para resolver una tarea específica.
Hilos en un programa:
Los hilos son subprocesos dentro de un proceso principal.
Pueden compartir recursos y comunicarse entre sí.
Ejemplo: Un programa de procesamiento de imágenes puede tener un hilo para cargar imágenes y otro para aplicar filtros.
Servidores web:
Los servidores web manejan múltiples solicitudes de clientes al mismo tiempo.
Cada solicitud se procesa en un hilo o proceso separado.
La cooperación entre los hilos permite que el servidor atienda a varios clientes simultáneamente.
Sistemas distribuidos:
En un sistema distribuido, múltiples computadoras trabajan juntas para realizar una tarea.
La cooperación entre nodos permite la escalabilidad y la redundancia.
Ejemplo: Un sistema de almacenamiento en la nube distribuido.

Procesos de comunicación:
Los procesos que se comunican entre sí a través de canales compartidos.
Ejemplo: Dos aplicaciones que intercambian mensajes a través de sockets de red.
Procesos de producción y consumo:
Un productor genera datos y un consumidor los utiliza.
La cooperación entre ellos garantiza que los datos se manejen correctamente.
Ejemplo: Un productor que genera pedidos y un consumidor que procesa esos pedidos en una tienda en línea.

3. ¿En qué consiste la Exclusión mutua?

La exclusión mutua en informática se refiere a un concepto fundamental para garantizar la integridad y el correcto funcionamiento de sistemas concurrentes.

La exclusión mutua asegura que solo un proceso pueda acceder a una sección crítica (un fragmento de código) a la vez.
La sección crítica es donde se modifica un recurso compartido (como una variable o una estructura de datos). El objetivo es evitar que varios procesos accedan simultáneamente a la misma sección crítica y causen conflictos o resultados incorrectos.
Técnica para lograr la exclusión mutua:

Inhabilitar las interrupciones durante el conjunto de instrucciones más pequeño que protege la sección crítica. Esto evita que el código de una interrupción se ejecute en medio de la sección crítica. En sistemas multiprocesador de memoria compartida, se utiliza la operación test-and-set sobre una bandera para esperar hasta que otro procesador la libere. Esta técnica se conoce como spin lock o espera activa.
Requisitos esenciales para la exclusión mutua:

- Seguridad (): A lo sumo, un proceso puede estar ejecutándose una vez en la sección crítica.
- Supervivencia (): Las peticiones para entrar y salir de la sección crítica deben concederse, evitando deadlocks e inanición.
- Ordenación (): Si una petición para entrar en la sección crítica ocurrió antes que otra, se garantiza la entrada en ese orden.
- Algoritmos de exclusión mutua:

Basados en tokens:
Un único token se comparte entre varios nodos. El proceso que posee el token tiene acceso a la región crítica. Ejemplos: Algoritmo de servidor centralizado, algoritmo en anillo, algoritmo de Ricart y Agrawala.
No basados en tokens:
El acceso a la región crítica se otorga mediante turnos y mensajes. Se utilizan marcas de tiempo para accesos equitativos. Ejemplo: Algoritmo de cola compartida.
Basados en quorums:
Cada nodo pertenece a una red dividida en subconjuntos (quorums).
El quorum se bloquea cuando entra en la región crítica.

4. ## Defina Interbloqueo.

   El interbloqueo ocurre cuando varios procesos se bloquean entre sí porque cada uno está esperando algo que el otro tiene. Es como si Juan y María se quedaran atrapados esperando el control de la computadora, y ninguno de los dos puede jugar.

5. ## Defina Inanición.

   Cuando una persona esta muriendo de hambre extrema.

6. ## Defina Excesiva Cortesía.

   Cuando una persona es demasiado cortes al grado de ser excesivo.

7. ## ¿Qué son los Hilos?

   En programación, un hilo es una entidad que permite que un programa realice múltiples acciones al mismo tiempo

8. ## ¿Qué son los Semáforos?

   un semáforo es una técnica de señalización que utiliza variables especiales dentro de un lenguaje de programación.

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

   Que se puede hacer la multitarea, mejora en rendimiento y la gestión de procesos.

10. Bibliografía

Título de la Página: Concurrencia URL: Concurrencia en Sistemas Operativos Fecha de Acceso: 7 de mayo de 2024 Concurrencia - sistemas operativos (weebly.com)

Jorge. (2024). procesos concurrentes. Sistemas operativos avanzados. Recuperado de [SESION 6 SISTEMAS OPERATIVOS AVANZADOS.pdf (aiu.edu)]