Final de Sistemas Operativos

20/02/2024

Nota:

Apellido y Nombre	*Profesor*	*Tomé conocimiento de la nota: (Sólo aplazos)*

Preguntas teóricas					Ejercicios	
1	2	3	4	5	1	2

A) Teoría: Explícitamente defina como VERDADERA o FALSA cada una de estas afirmaciones justificando brevemente.

1) Si un recurso puede ser accedido a la vez (en paralelo) por un conjunto de procesos, el mismo no puede ser causa de un deadlock entre ellos, pero sí de un livelock.

2) En un sistema monoprocesador, el algoritmo de planificación de corto plazo podría generar que se produzca una condición de carrera. Esto no es cierto si los procesos están correctamente sincronizados.

3) En entornos que no permiten E/S asincrónicas, estas pueden ser simuladas utilizando hilos.

4) En un sistema con una alta tasa de TLB hits, es posible que un *page fault* no genere accesos a disco.

5) La compactación es una estrategia útil en todos los esquemas de asignación de bloques en disco.

B) Práctica: Resuelva los ejercicios justificando las respuestas

1. El siguiente pseudocódigo simula la interacción de N usuarios en la red social "Z" y un proceso analizador que toma cada uno de los posts generados para analizarlos. A pesar de estar sincronizado, el sistema se desempeña más lento de lo esperado y frecuentemente deja de funcionar hasta que el administrador reinicia el proceso Analizador:

Usuario (N instancias)	**Analizador (1 instancia)**

While(1) {
wait(mutexPosts)
post = generarPost();
postear(post, postsNuevos);
signal(hayPosts);
mostrarEnPantalla(post);
signal(mutexPosts);
}
	While(1) {
	wait(mutexPosts);
	wait(hayPosts);
	post = obtenerPost(postsNuevos);
	resultado = procesar(post)
	guardarEnDisco(resultado)
	signal(mutexPosts)
	}

Variables compartidas: **postsNuevos** -> cola que contiene los nuevos posts a analizar Semáforos: **hayPosts** -> semáforo contador inicializado en 0 **mutexPosts** -> semáforo mutex	

Encuentre al menos 3 errores y/o mejoras en la sincronización planteada (no sincronizar nuevamente, solamente marcar y explicar los errores encontrados).

2. Se tienen dos procesos, P1 y P2 que generan respectivamente las siguientes secuencias de referencias a memoria (expresadas en número de página solicitada) que se repiten indefinidamente:

P1: 10 11 12 13 14 15 16 17 . . . (repite desde el principio)
P2: 10 11 0 3 4 3 11 4 0 3 4 3 0 11 . . . (repite desde el principio)

Se sabe que cada proceso tiene 4 frames asignados, la sustitución de páginas es local mediante LRU y se tiene una TLB de 2 entradas con algoritmo de sustitución FIFO.
Para cada proceso y de forma aislada, responda:

a) Si se actualizara el sistema con una TLB de 4 entradas ¿Mejoraría en algo su ejecución? Justifique.
b) ¿Podría generarse thrashing? Justifique. En caso afirmativo indique cómo podría solucionarse.

El tiempo de duración del examen final será de 90' a contar desde el momento de comienzo del mismo. Si el alumno por algún motivo comenzará más tarde sólo podrá utilizar el tiempo remanente. Utilice hojas separadas para la teoría / ejercicios.

Final de Sistemas Operativos

20/02/2024

Nota:

RESOLUCIÓN

TEORÍA

1. Falso. Si el recurso es compartido no se cumple la condición de mutua exclusión, que es necesaria para la ocurrencia tanto de deadlock como de livelock.
2. Verdadero. El algoritmo de planificación decide en qué momento interrumpir a un proceso, haciendo que tal vez no complete una operación que debía ser atómica, generando la condición de carrera. Sin embargo, si los procesos están correctamente sincronizados el resultado siempre será el correcto sin importar el orden en que el planificador elija ejecutarlos.
3. Verdadero. Por ejemplo, al crear un hilo y realizar la E/S de manera sincrónica en el mismo, el resto del proceso sigue ejecutando como si la misma fuera asincrónica.
4. Falso. Un fallo de página siempre accede a disco al menos una vez para traer la página faltante a memoria.
5. Puede justificarse falso, los esquemas de asignación de bloques enlazado e indexado pueden asignar cualquier bloque aunque los mismos no estén contiguos por lo que no es necesario compactar el disco. También podría justificarse verdadero dado que, independientemente del esquema de asignación, tener los bloques de un archivo contiguos en disco puede mejorar los tiempos de acceso.

PRÁCTICA

1.
En Usuario:

- El wait de “mutexPosts” debería estar luego de ”generarPost”, para que cada Usuario no se bloquee innecesariamente incluso antes de generar el post.
- El signal de “mutexPosts” debería estar inmediatamente luego de “postear” para que la sección crítica sea lo más chica posible.

En Analizador:

- El orden de los wait mutex debería ser al revés ya que los mismos podrían generar un deadlock.
- Luego de “obtenerPost” se debería realizar el signal inmediatamente para no bloquear innecesariamente al resto de los procesos mientras se realiza una operación larga en disco.

2.

a) Solamente el Proceso 2 se vería beneficiado por la mejora al repetir las mismas 4 páginas varias veces, ahorrándose muchos accesos a tablas de páginas.

b) Se genera thrashing en el Proceso 1 ya que su localidad temporal es de 8 páginas, se solucionaría aumentando su asignación de frames a 8.

El tiempo de duración del examen final será de 90’ a contar desde el momento de comienzo del mismo. Si el alumno por algún motivo comenzará más tarde sólo podrá utilizar el tiempo remanente. Utilice hojas separadas para la teoría / ejercicios.