CC4302 Sistemas Operativos – Examen – Semestre Otoño 2019 – Prof.: Luis Mateu

Pregunta 1

I. (3 puntos) Se necesita agregar un scheduler de disco a nSystem. Debe garantizar la exclusión mutua en el acceso a disco e implementar la estrategia SCAN para reducir el movimiento del cabezal. Esta estrategia atiende los accesos pendientes barriendo repetidamente con el cabezal del disco desde las pistas internas hacia las pistas externas. Las funciones que se requiere programar son:

void nRequestDisk(int track);	Solicita acceso a disco indicando la pista
void nReleaseDisk();	Notificación de término de uso del disco

Una tarea invoca *nRequestDisk* para solicitar acceso exclusivo al disco. Debe esperar si el disco está ocupado. La identificación numérica de pista, *track*, es linealmente proporcional a la distancia de la pista al centro del disco. La pista 0 es la más cercana al centro. La tarea accederá a la pista *track* después del retorno de *nRequestDisk*. Luego invocará *nReleaseDisk* notificando el término del uso del disco. Si en ese momento hay varias solicitudes de acceso en espera, y se acaba de acceder a la pista t, entre todos los *nRequestDisk(t')* pendientes Ud. debe autorizar el acceso que requiera el cabezal en la pista t' más cercana a t sujeto a que $t' \geq t$. Autorice haciendo que ese *nRequestDisk(t')* retorne. Si no hay ninguna solicitud con esas características, autorice la solicitud que lleve el cabezal a la pista más cercana al centro del disco. Por ejemplo si el cabezal está en la pista 4 y hay solicitudes en espera para las pistas 2, 2, 3, 4, 6 y 10, el orden de autorización debe ser 4, 6, 10, 2, 2 y 3.

Implemente las funciones pedidas usando los procedimientos de bajo nivel de nSystem (*START_CRITICAL*, *Resume*, *PutTask*, etc.). Ud. no puede usar otros mecanismos de sincronización ya disponibles en nSystem como semáforos, monitores, mensajes, etc. No olvide que puede usar variables globales y modificar el descriptor de tarea.

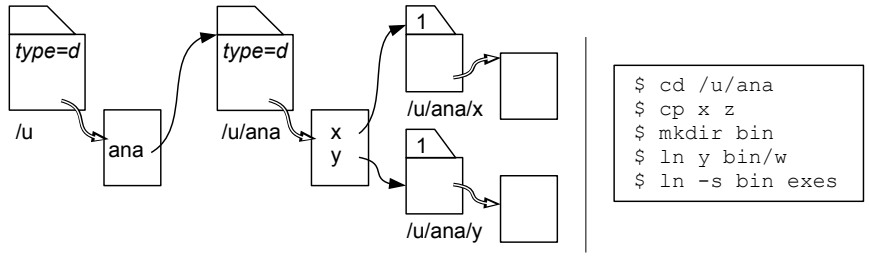
Para resolver este problema use 2 colas de prioridades con esta API:

- *Squeue MakeSqueue( )*: entrega una nueva cola de prioridad.
- *void PutTaskSqueue(Squeue q, nTask t, int p)*: agrega una tarea t con prioridad p a la cola q.
- *nTask GetTaskSqueue(Squeue q)*: retorna y extrae de la cola q la tarea con la mejor prioridad (la de menor valor numérico).
- *int EmptySqueue(Squeue q)*: retorna verdadero si y solo si q está vacía.

II.- (1 punto) ¿Cuál es la principal ventaja de un núcleo clásico de Unix con respecto a un núcleo moderno? ¿Y cuál es su principal desventaja?

III.- (1 punto) Explique si tiene o no sentido usar la estrategia del working set en un sistema operativo que ofrece solo procesos livianos. ¿Se puede hacer swap de un proceso liviano? ¿Cuál otra estrategia sería más eficaz y por qué?

IV.- (1 punto) La figura muestra a la izquierda varios archivos y directorios de la partición /u. El número que aparece en los inodos de los archivos x e y es el contador de links duros. A la derecha se muestran los comandos que ejecuta la usuaria *ana*. Rehaga la figura de acuerdo a los cambios realizados.



Pregunta 2

a.- (3 puntos) Resuelva el mismo problema de la parte *I* de la pregunta 1 considerando ahora una máquina multi-core en la que no existe un núcleo de sistema operativo y por lo tanto no hay un scheduler de procesos. Ud. debe programar las funciones *requestDisk(track)* y *releaseDisk( )*.

Para programar su solución Ud. dispone de spin-locks, la función *coreId( )* y la constante NCORES. Necesitará usar variables globales. Puede usar una cola de prioridad sin implementarla, pero debe especificar los encabezados de las funciones que manipulan la cola.

Restricción: Dado que no hay un núcleo de sistema operativo, la única forma válida de esperar es utilizando un spin-lock. Otras formas de *busy-waiting* no están permitidas. No hay: *fifoqueues*, *Squeue*, *nTask*, etc.

b.- (1 punto) ¿Cuál es el tamaño máximo de un archivo que usa un solo bloque de indirección simple en una partición con bloques de 2 KB? ¿Y si la partición posee bloques de 4 KB? Explique.

c.- (1 punto) Considere un sistema operativo que implementa la estrategia del working set. Para el área de paginamiento se usa un solo disco que tiene una tasa de transferencia de 50 MB/segundo y un tiempo de acceso de 10 milisegundos. Estime (i) el máximo número de *page-faults* por segundo que se puede atender cuando hay penuria de memoria; y (ii) cuanto tiempo tomaría llevar a disco (swap) un proceso cuyas páginas residentes en memoria totalizan 150 MB.

d.- (1 punto) Se acaban de leer las pistas 250 y 300 de un disco duro, en ese orden. 6 procesos se encuentran en espera con peticiones para leer las pistas 100, 800, 200, 700, 900 y 400. En qué orden se atenderán las peticiones cuando la estrategia es: (a) *shortest seek first*, (b) método del ascensor o *look*.