CC4302 Sistemas Operativos – Examen – Semestre Otoño 2016 – Prof.: Luis Mateu

Pregunta 1

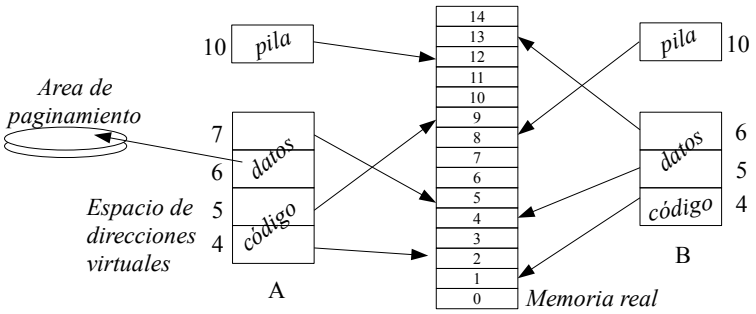
I. (3 puntos) Se desea agregar un sistema de mutex con prioridades a nSystem como herramienta de sincronización nativa. Concretamente la API de este sistema es:

- *nMutex *nMakePriMutex( )*: Entrega un mutex con prioridades. En todo instante puede existir a lo más un solo propietario de este mutex.
- *void nLock(nMutex *mtx, int pri)*: Solicita la propiedad de *mtx* con prioridad *pri* (un valor entero entre 0 y 3). Si *mtx* está libre, el solicitante adquiere su propiedad de inmediato y *nLock* retorna. Si actualmente otra tarea tiene la propiedad de *mtx*, el solicitante espera hasta adquirir en forma exclusiva *mtx* cuando otra tarea invoque *nUnlock*.
- *void nUnlock(nMutex *mtx)*: Libera *mtx*. Si existen varias tareas esperando adquirir *mtx*, se entrega su propiedad a la tarea que lo haya solicitado con la prioridad más alta (3 es la más alta, 0 la menor).

Defina la estructura de datos para nMutex e implemente esta API 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.

II. (1,5 puntos) Ud. necesita garantizar la exclusión mutua al acceder a la cola de procesos "ready" de un sistema operativo. Confeccione una tabla de 3 columnas por 2 filas, considerando para las columnas las siguientes 3 herramientas: 1 semáforo, 1 *spin-lock*, inhibición de interrupciones, y para las filas estos 2 tipos de núcleo: clásico mono-core y multi-threaded para multi-core. En cada celda de la tabla indique con un sí o un no la factibilidad de usar la herramienta correspondiente a esa columna para el tipo de núcleo de esa fila. En caso de duda complemente con una explicación.

III. (1,5 puntos) El diagrama muestra la asignación de páginas en un sistema Unix que ejecuta los procesos A y B. Las páginas son de 4 KB.



El núcleo utiliza la estrategia *copy-on-write* para implementar *fork*. (A) Construya la tabla de páginas del proceso A después de que éste invoca *sbrk* pidiendo 2 KB adicionales. En la tabla indique página virtual, página real y atributos de validez y escritura. (B) Considere que el proceso B invocó *fork* y que el proceso hijo modificó la página 6. Construya la tabla de páginas para el proceso B y su hijo.

Pregunta 2

Parte a. (3 puntos) Resuelva el mismo problema de la pregunta 1 considerando ahora una máquina *octa-core* en la que no existe un núcleo de sistema operativo y por lo tanto no hay un scheduler de procesos. La estructura de datos y las funciones que Ud. debe programar son las siguientes:

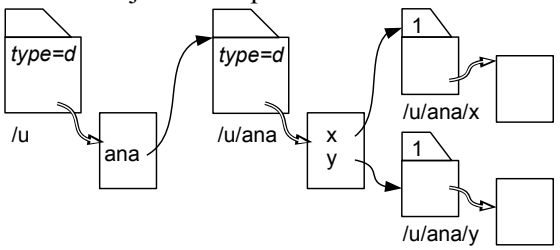
```
typedef struct { … } Mutex;
void initMutex(Mutex *mtx);
void lock(Mutex *mtx);
void unlock(Mutex *mtx);
```

La función *lock* no recibe un parámetro para la prioridad porque ésta está dada por el número del core que solicita la operación. El core 0 tiene la mejor prioridad y el 7 la peor. Para programar su solución Ud. dispone de *spin-locks* y la función *coreId( )* que entrega el número del core que la invoca.

Restricción: Dado que no hay un núcleo de sistema operativo, la única forma válida de esperar a que el mutex se libere es utilizando un spin-lock. Otras formas de *busy-waiting* no están permitidas. No hay *malloc* ni *fifoqueues*.

Parte b. (1 punto) ¿Cuales serían las consecuencias de implementar un sistema de paginamiento sin una TLB (*Translation Lookaside Buffer*)?

Parte c. (1 punto) La figura de la izquierda muestra varios archivos y directorios de la partición /u en un sistema Unix. A la derecha se muestran los comandos ejecutados por *ana*:




```
$ cd /u/ana
$ mkdir tmp
$ ln y tmp/w
$ cp tmp/w z
$ ln -s tmp/w v
```

Rehaga la figura de acuerdo a los cambios realizados por ana. No olvide indicar el contador de links para los archivos normales.

Parte d. (1 punto) Se acaban de leer los bloques 150 y 300 en ese orden en un disco duro. Ahora 6 procesos se encuentran en espera con peticiones para leer los bloques 100, 800, 200, 700, 900 y 500. En qué orden se harán las lecturas cuando la estrategia es: (i) *shortest seek first*, (ii) método del ascensor o *look*.