CC4302 Sistemas Operativos – Examen – Semestre Primavera 2017 – Prof.: Luis Mateu

Pregunta 1

(i) ¿Tiene sentido usar spin-locks en una máquina monocore? Explique. Responda entonces cómo implementaría la exclusión mutua en un núcleo moderno de Unix para una máquina monocore.

(ii) Ud. debe elegir una herramienta para garantizar la exclusión mutua en una sección crítica de un módulo del núcleo de Linux. Considere una máquina multicore. ¿Bajo qué condiciones sería más eficiente usar un spin-lock y cuando sería más eficiente un semáforo?

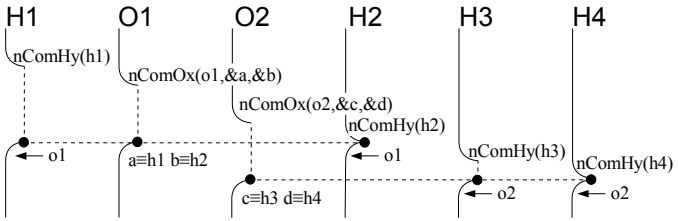
(iii) ¿Qué es lo más importante que perderían los usuarios si los computadores modernos no ofrecieran paginamiento (ni segmentación)? Explique considerando qué pasa cuando se "caen" aplicaciones como chrome, word, windows media player, waze, whatsapp, etc. debido a problemas con los punteros.

(iv) Compare ventajas y desventajas de usar un disco tradicional para paginamiento versus un solid state drive (SSD). En su comparación considere número de page faults atendidos por segundo, velocidad (en MB/seg.) para cargar en memoria un proceso que fue llevado a disco (estado swap) y tiempo de vida del dispositivo de paginamiento.

Pregunta 2

a.- (4,5 puntos) Se requiere implementar de manera nativa en nSystem las siguientes funciones:

```
void nCombineOxy(Oxygen *oxy, Hydrogen **pH1, Hydrogen **pH2);
Oxygen *nCombineHydro(Hydrogen *hydro);
```



La figura de arriba muestra la sincronización requerida. La tarea *H1* aporta un átomo de hidrógeno *h1* llamando a *nCombineHydro*. Debe esperar por otro átomo de hidrógeno y uno de oxígeno. La tarea *O1* aporta el átomo de oxígeno *o1* invocando *nCombineOxy*. Finalmente la tarea *H2* aporta el último átomo de hidrógeno *h2*. Entonces recién *H1*, *O1* y *H2* pueden continuar. *H1* y *H2* retornan *o1* y *nCombineOxy* entrega *h1* en *a* y *h2* en *b*. La misma sincronización ocurre con las tareas *O2*, *H3* y *H4*.

Implemente las funciones *nCombineOxy* y *nCombineHydro* 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. Declare las variables globales que necesite y especifique los campos que agregará al descriptor de tarea. La ocupación de los átomos debe hacerse por orden de llegada. Debe evitar cambios de contexto innecesarios. Use una cola (*Queue*) para las tareas que aportan hidrógeno y otra cola para las tareas que aportan oxígeno. Considere que la función *LengthQueue(q)* entrega el largo de la cola *q*.

b.- (1,5 puntos) 5 procesos se encuentran en estado de espera porque hicieron peticiones para leer bloques del disco en el siguiente orden: 300, 600, 200, 900, 800. El último bloque leído fue el 500 y el penúltimo el 700. Indique en qué orden se harán las lecturas de estos 5 procesos cuando la estrategia de scheduling de disco es: (i) *first come first served*, (ii) *shortest seek first*, (iii) método del ascensor (o *look*).

Pregunta 3

I. (4,5 puntos) Considere una máquina multi-core en la que no existe un núcleo de sistema operativo y por lo tanto no hay un scheduler de procesos. La siguiente es una implementación incompleta del mismo problema de la pregunta 2 parte a.-

```
Hydro *_hydro; // Buffer de 1 item
Oxy **_pOxy;
int _m= OPEN;  // mutex
// condición buffer vacio
int _empty= OPEN;
// condición buffer lleno
int _full= {CLOSED, CLOSED};
// 2 condiciones adicionales
int _wait[2]= CLOSED;
int _k= 0;
Oxygen *combineHydro(
  Hydrogen *hydro) {
  ... complete ...
  _k= (_k+1)%2;
  ... complete ...
}
```

```
void combineOxy(Oxy *oxy,
  Hydro **pH1, Hydro **pH2) {
  spinLock(&_m);
  spinLock(&_full); // wait buffer lleno
  *pH1= _hydro; // 1er átomo de hidrógeno
  *_pOxy= oxy;
  // signal buffer vacio
  spinUnlock(&_empty);
  spinLock(&_full); // wait buffer lleno
  *pH2= _hydro; // 2do átomo de hidrógeno
  *_pOxy= oxy;
  // signal 1er y 2do hidrógeno
  spinUnlock(&_wait[0]);
  spinUnlock(&_wait[1]);
  // signal buffer vacio
  spinUnlock(&_empty);
  spinUnlock(&_m);
}
```

En esta pregunta la ocupación de los átomos se hace en cualquier orden. Complete la función *combineHydro* sin alterar el resto de la implementación. Por claridad las variable globales empiezan con _. Observe que algunos spin-locks se usan como condiciones.

II. (1,5 puntos) Considere un sistema Unix que implementa la estrategia del reloj. El sistema posee 6 páginas reales disponibles y corre un solo proceso. La figura indica el estado inicial de la memoria, mostrando las páginas residentes en memoria, la posición del cursor y el valor del bit R. Dibuje los estados por los que pasa la memoria para la siguiente traza de accesos a páginas virtuales: 5, 7, 4, 7, 2, 6.