Final de Sistemas Operativos

26/05/2016

Nota:

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

Preguntas teóricas					**Ejercicios**	
1	**2**	**3**	**4**	**5**	**1**	**2**

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

1) Una entrada-salida bloqueante puede ser realizada como no bloqueante, aunque esto carece de utilidad

2) Un sistema operativo siempre puede solucionar los problemas que ocurran por el fenómeno de inversión de prioridades

3) La tabla de páginas invertidas no permite compartir memoria a los procesos

4) Los ULTs de un mismo proceso pueden quedar en deadlock utilizando semáforos

5) Por lo general, el acceso directo a un bloque en un esquema de asignación indexada como ext2 es más rápido que un esquema enlazado pero más lento que un esquema continuo.

B) <u>Práctica</u>: Resuelva los ejercicios justificando las respuestas

1) Un proceso corre el código que se ve en la siguiente tabla, utilizando 3 frames que se encuentran inicialmente cargados con las páginas 5, 9 y 1, en ese orden. Se sabe también que:

- El stack se encuentra en la página 5
- Los datos se encuentran en las páginas 8 y 9
- El algoritmo de reemplazo es clock, y aún no ha ejecutado

a) Determine cuántos fallos de página ocurrirán al intentar ejecutar la sentencia resaltada, sabiendo que el instruction pointer se encuentra allí. Considere que dicha sentencia es la primera de la página.

b) Indique qué valores tiene el stack en cada momento.

CALCULOS.C	PÁGINA

`int main() {`
`  int c = 3;`
`  ...`
	1

`  int resultado_final = factorial(c);`
`  ...`
`} // fin de main`
	2

`…`
`int factorial(int num) {`
`  if (num == 1) return 1;`
`  int resultado = factorial(num - 1);`
`  return (num * resultado);`
`}`
	3

2) Un sistema corre dos instancias del mismo proceso, utilizando el algoritmo Virtual Round Robin, con Q = 3. Cuando es necesario, se utiliza una biblioteca que planifica hilos utilizando FIFO, y que permite replanificar luego de una E/S. El proceso mencionado se compone de dos hilos que ejecutan como lo indica la tabla.

Indique mediante un diagrama de Gantt el resultado de la ejecución si:

a) Los hilos son KLTs, la primera instancia del proceso se crea en el instante 0 y la segunda en el instante 1

b) Los hilos son ULTs, la primera instancia del proceso se crea en el instante 0 y la segunda en el instante 3

	Inicio	CPU	E/S	CPU
Hilo A	0	2	3	4
Hilo B	2	1	1	2

Considere el inicio de cada hilo como el tiempo que pasa desde que el proceso padre se crea, hasta que el hilo puede ejecutar.

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

26/05/2016

Nota:

Resolución

Teoría

1) F. Por ejemplo, cuando se usan ULTs con jacketing se evita bloquear a la unidad de planificación asociada. Si usamos una IO bloqueante, el SO pasa el KLT/proceso completo a estado bloqueado, por lo que no se pueden planificar otros ULTs que estén listos. Si se usa una IO no bloqueante, la biblioteca de ULTs puede elegir otro hilo de usuario para ejecutar.

2) V. Dependiendo de qué tipo de recurso se trate, puede solucionarlo. Por ejemplo, si se trata de un recurso administrado por el SO, este podría detectar cuando un proceso no puede obtener un recurso si el mismo está siendo retenido por uno de menor prioridad, y en ese caso aplicar la herencia de prioridades.
- Falso si se explicita que cuando el recurso no lo maneja el so, no puede darse cuenta.

3) Opciones de respuesta:
V. Porque en la definición original de la estructura de la tabla, solo se tiene # página, pid, y sgte; por lo tanto no hay forma de asociar dos o más procesos a un mismo frame.
F. Si se modificara la estructura y esta tuviera una lista de pares pagina/PIDs, se podría lograr. También se podría tener una tabla de páginas por proceso sólo para aquellas que están compartidas (nota: HP-UX hace algo así). En ambos casos, se lograría el cometido de forma limitada y con ciertos costos en performance.

4) F: si se bloquea un hilo, se bloquean los otros y por lo tanto no llega a formarse la espera circular.
Nota: si alguno aclara que el proceso entró en deadlock consigo mismo y lo argumenta bien, dejando en claro que entiende las condiciones de deadlock, tal vez podría considerarse bien también. Queda a criterio del corrector.

5) V. Más rápido que el enlazado porque no hay que leer todos los bloques previos del archivo para conocer la ubicación del bloque a leer. Más lento que continua porque en el caso de archivos grandes, podría ser necesario leer algunos bloques de punteros antes de conocer la ubicación del bloque en cuestión (mientra que en continua eso nunca pasa)

Práctica

1.a)

Inicialmente hay que ir a buscar la página 2, que es en la que se encuentra la línea que apunta el IP/PC. (1er PF)

Como tiene asignación fija de 3 frames y los 3 están ocupados, hay que reemplazar uno. Estado inicial:

5 1 ←
9 1
1 1

Como aún no se aplicó el algoritmo de sustitución, todos los bits de U están en 1, por lo que en este caso la decisión será igual que si se tratara de FIFO, se reemplazará la primera en cargarse (5) y el resto quedarán con bit U = 0 (salvo la pág 2 que se acaba de cargar). El puntero de próxima víctima se avanzará a 9.

2 1
9 0 ←
1 0

Luego necesita obtener el valor de c, que como es una variable local, se encuentra en el stack. El mismo se encuentra en la pág que acabamos de reemplazar, por lo que hay que volver a traerla a memoria. (2do PF). Nuestra víctima apunta a 9, cuyo bit de uso está en 0, por lo que lo reemplazamos

2 1
5 1
1 0 ←

Finalmente, la última página que necesitamos es la que contiene el código de la función factorial, 3, (3er PF)

2 1 ←
5 1
3 1

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

26/05/2016

Nota:

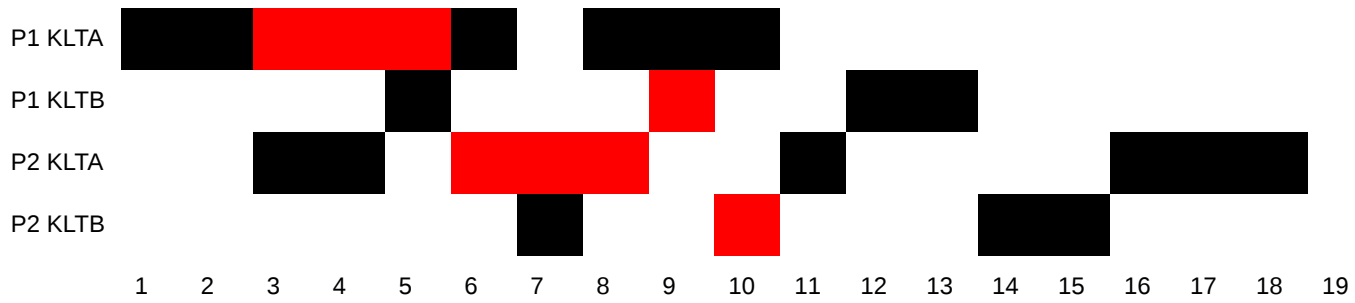
El resto de las referencias a memoria que se generen para ejecutar la función van a requerir la pág 3 y 5 (código de factorial y stack) que ya se encuentran cargadas, por lo que no generarán nuevos PFs.

Por lo tanto, se requiere 3 PFs.

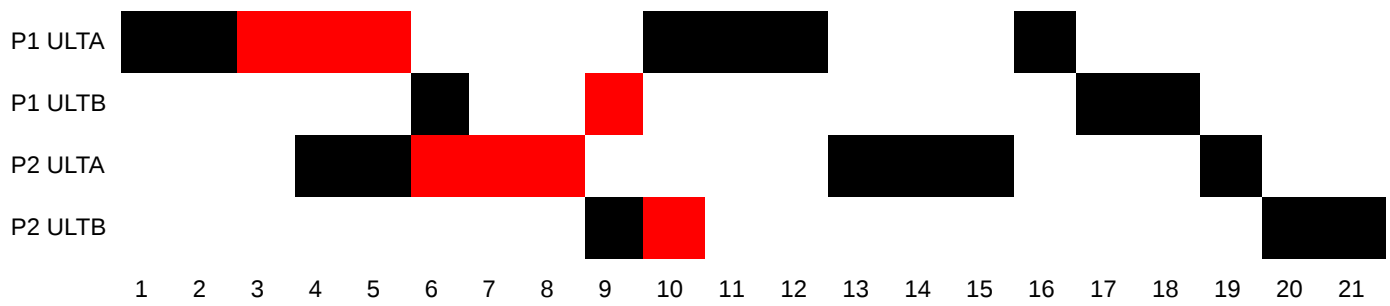
1.b)

	Stack
Se llama factorial por primera vez. “main” y ”c” se encuentran cargados previamente	factorial -> num = 3 main -> c = 3
Como num != 1 se vuelve a llamar factorial	factorial -> num = 2 factorial -> num = 3 main -> c = 3
Como num != 1 se vuelve a llamar factorial	factorial -> num = 1 factorial -> num = 2 factorial -> num = 3 main -> c = 3
Como num == 1 , se devuelve 1 (saliendo de la última función) que se guarda en resultado	resultado = 1 factorial -> num = 2 factorial -> num = 3 main -> c = 3
Se realiza la multiplicación entre 1 (resultado) y 2 (num) y se retorna (saliendo de la segunda llamada a factorial) que se guarda en resultado	resultado = 2 * 1 factorial -> num = 3 main -> c = 3
Se realiza la multiplicación entre 2 (resultado) y 3 (num) y se retorna (saliendo de la primer llamada a factorial) que se guarda en resultado_final	main -> c = 3, resultado_final = 3 * 2 * 1

2) a)



2) b)



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.