EXAMEN SISTEMAS OPERATIVOS AVANZADOS

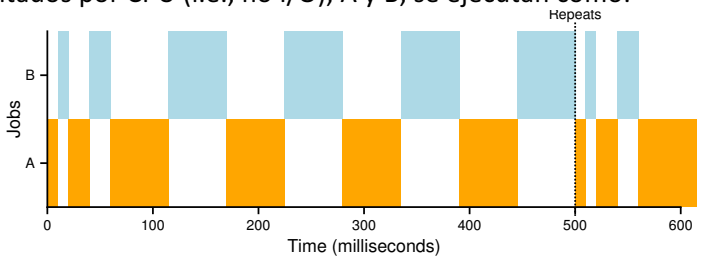
JUEVES 11 NOVIEMBRE 2016

NOMBRE Y FIRMA:

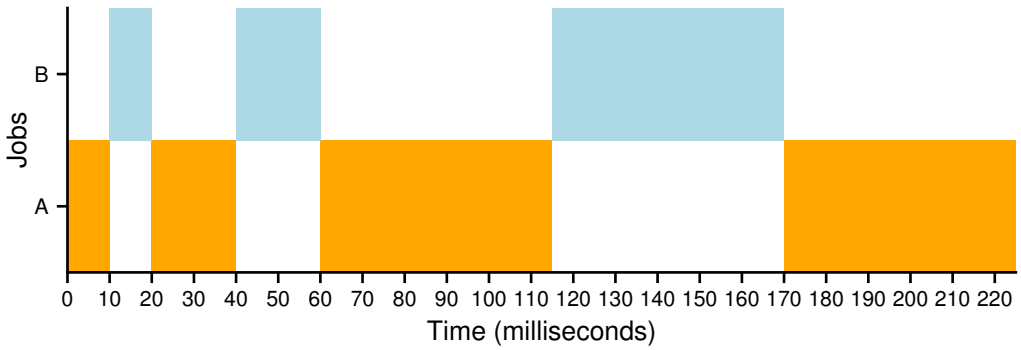
P1:	/2	P3:	/1	P5:	/1	LAB1:	/2
P2	/2	P4:	/1	P6:	/1	LAB2:	/2

PROBLEMA 1 (2 PUNTOS)

Se dispone de un planificador MLFQ del que, desafortunadamente, se desconocen todos sus parámetros. Solo disponemos de dos trazas de ejecución de las que debemos derivar como son estos. En la primera traza dos Jobs limitados por CPU (i.e., no I/O), A y B, se ejecutan como:



El proceso se repite cada 500ms. Para aclarar la situación se provee de la siguiente ampliación de la traza de ejecución:



A. ¿Cuantas colas crees que tiene el planificador?
B. ¿Cuánto es el quantum de tiempo del planificador en la cola más prioritaria?
C. ¿Cuánto es en la menos prioritaria?
D. ¿Cuánto tarda en regresar cada job a la cola más prioritaria?
E. ¿Por qué se hace?

PROBLEMA 2 (2 PUNTOS)

Esta pregunta breve es acerca de la **segmentación**. La segmentación es una aproximación que usa un conjunto de registros para ayudar en la virtualización de la memoria.

a) ¿Para que se usa el registro base?

b) ¿Qué tipo de dirección contiene dicho registro? (virtual o física)

c) ¿Para que se usa el registro limite o *bounds*?

d) ¿Por qué se necesitan más de una pareja de registros base/limite por proceso?

e) ¿Cuantas parejas de registros debe soportar el hardware (y por que)?

f) Dibujar un esquema de cómo debería estar constituida ser la MMU (dos elementos hardware y el interconexionado "aproximado")

Ahora, supón que tenemos un Sistema con la siguiente configuración. Solamente hay dos segmentos soportados por el hardware. El espacio de direcciones es pequeño (1KB), y la cantidad de memoria física disponible es 16KB. Supón que el registro base del Segmento-0 tiene un valor de 1KB y que el tamaño de dicho segmento es 300bytes y que crece en direcciones ascendentes. Supón que el segmento 1 tuene un registro base con valor 5KB, tiene 300bytes y crece en direcciones descendentes. Supongamos que queremos ejecutar el siguiente programa:

```
void *ptr = 20;
while (ptr <= 1024) {
        int x = (int *) *ptr;       // LINEA 1
         ptr = ptr + 20;            // LINEA 2
```

g) ¿Que direcciones virtuales genera la LINEA1 al de-referenciar ptr? (suponer que el programa se ejecuta hasta el final. Simplemente proporciona una lista de las direcciones generadas. No te preocupes de los accesos a x o los *fetchs* de instrucciones).

h) ¿Cuantas iteraciones del programa se ejecutarán antes llegar a un fallo de segmentación?

i) ¿Qué direcciones físicas serán generadas en la de-referenciación de *ptr* antes de que el programa se pare?

j) ¿Qué direcciones físicas legales podría haber generado la de-referenciación de *ptr*(si el programador hubiese sido más cuidadoso).

PROBLEMA 3 (1 PUNTOS)

Un sistema operativo emplea páginas multinivel en el sistema de memoria virtual. Es decir, es un sistema con un directorio de páginas que apuntan a páginas de la tabla de páginas. El directorio tiene capacidad para 32 entradas. Cada PDE tiene un bit de validez y el PFN donde está alojado la página de la tabla de páginas. A continuación, se muestra el contenido del directorio en un (proceso) caso particular:

7f fe 7f 7f 7f d4 7f 7f 7f 9e 7f ad 7f 7f 7f d6
7f 7f 7f 7f 7f 7f 7f e9 a1 e8 7f 7f 7f 7f 7f 7f

¿Cuánto espacio salva esta estructura con respecto a una tabla de páginas lineal?

(1 punto) Problema 4 (fall-mid-11)

¿El algoritmo de remplazo es una política o un mecanismo? ¿Por qué razón LRU puede resultar difícil de implementar de manera precisa? ¿Qué aproximación te permitiría hacerlo de forma imprecisa pero poco costosa?

(2 puntos) Problema 5 (spring-16)

Algunos pasos de la gestión con LDE son llevados a cabo por software del sistema operativo (SO) bien, otros gestionados por el Hardware (HW), y algunos en el propio programa de usuario (USR). A continuación, se listan los pasos llevados a cabo para ejecutar un programa que se sale después de hacer una llamada al sistema. ¿Cuál es el responsable en cada caso?

- Crear una entrada en la lista de procesos (OS) (HW) (USR)
- Reservar la memoria para ejecutar el programa (OS) (HW) (USR)
- Cargar el programa en memoria (OS) (HW) (USR)

- Preparar el stack de usuario con argv (OS) (HW) (USR)
- Preparar el stack de kernel y completarlo con reg/PC (OS) (HW) (USR)
- Ejecutar la instrucción de retorno de trap (OS) (HW) (USR)
- Conmutar a modo usuario (OS) (HW) (USR)
- Colocar el PC apuntando a main() (OS) (HW) (USR)
- Empezar a ejecutar main() (OS) (HW) (USR)
- Llamar a una llamada al sistema (OS) (HW) (USR)
- Ejecutar la instrucción trap (OS) (HW) (USR)
- Salvar los registros al stack de kernel (OS) (HW) (USR)
- Conmutar a modo kernel (OS) (HW) (USR)
- Manejar la trap (OS) (HW) (USR)
- Realizar el trabajo de la syscall (OS) (HW) (USR)
- Ejecutar la instrucción de retorno de trap (OS) (HW) (USR)
- Restaurar los registros desde el stack de kernel (OS) (HW) (USR)
- Conmutar a modo usuario (OS) (HW) (USR)
- Colocar el pc apuntando a la instrucción posterior al trap(OS) (HW) (USR)
- Llamar a la llamada al sistema exit() (OS) (HW) (USR)

(1 punto) Problema 6 (spring-16)

Los siguientes diagramas temporales el tiempo de llegada a la cola de planificación y el tiempo en que el planificador les cede el uso de la CPU (única).¿Cuál es el tiempo de turna-round del job A? Suponer que A llega al comienzo del tiempo. Empieza a ejecutarse cuando "*" aparece en la segunda línea. Finaliza su ejecución cuando "x" aparece en la segunda línea.

Ejemplo ABABA * x	A llega en t=0, empieza a ejecutarse inmediatamente y finaliza en t=5. B se ejecuta entre t=1-2 y t=3-4

ABABABABAB * x	BBABABABABA * x	ABCBABCABCA * x
AAAAABBBBB * x	BCABBBBBABA * x	BBBBBBAAAAA * x

Repetir el mismo proceso, pero para el tiempo de respuesta (del Job A)

ABABABABAB * x	BBBBBBAAAAAA * x	ABCABCCCBA * x	BBABABABABA * x	AAAAABBBBBA * X

Problema práctico:

Escribir un programa que permita determinar el segmento de texto, código y stack en xv6.

Escribir una llamada al sistema que permita obtener el PID del proceso padre.

Modificar el planificador de xv6 de tal modo que todos los procesos en ejecución obtengan el mismo