Sistemas Operativos

I. T. Informática de Sistemas

Córdoba, 31 de febrero de 2001

Cuestiones

1. Proponga un modelo de planificación multinivel. Indique que características tendría y que tipo de procesos serían favorecidos y cuales perjudicados.

2. ¿Por qué el algoritmo de reemplazo de página NRU utiliza un bit que indica si una página ha sido modificada y el resto de algoritmos no utilizan este bit?

3. Explique las diferencias que encuentra entre las primitivas `wait()` y `signal()` sobre semáforos y las mismas primitivas sobre variables condición de monitores.

4. ¿Es posible la inanición de un proceso en el algoritmo de planificación HRRN?

5. Algunos sistemas localizan las tablas de segmentos y páginas de cada proceso en sus espacio virtual de direcciones. Indique las ventajas e inconvenientes de este método.

Problemas

1. Supóngase un proceso en UNIX que realiza las operaciones que se indican a continuación:

   (a) El proceso es creado en un instante en el cual la carga del sistema es muy alta. Posteriormente la carga del sistema vuelve a ser normal.

   (b) El proceso es planificado para ejecutar por primera vez.

   (c) El proceso ejecuta instrucciones que no necesitan llamadas al sistema y es planificado en varias ocasiones agotando siempre el cuanto de tiempo que le asigna el planificador.

   (d) El proceso realiza una escritura en el disco.

   (e) El proceso realiza una lectura de una posición de memoria que no se encuentra en memoria real y se produce un fallo de página.

   (f) El proceso realiza un intento de escritura en una posición de memoria sobre la que no tiene acceso.

   (a) Indique por qué estados pasaría el proceso a medida que va realizando las acciones anteriores.

   (b) Indique cómo se verían afectadas las estructuras de información del sistema, especialmente el PCB, a medida que se realizan las acciones siguientes.

2. Para la gestión del acceso a los usuarios de un sistema informático compuesto por tres computadoras se utiliza el siguiente método. La computadora principal almacena una el fichero `passwd` donde se encuentran los usuarios y sus claves de acceso. Las otras dos computadoras tienen una **copia** de este fichero que **no pueden modificar**. Cada computadora ejecuta un proceso de validación del acceso que lee las claves introducidas por el usuario y comprueba su validez.

Para la modificación de la clave de un usuario existe un **único proceso** que se ejecuta en la computadora principal y que es el encargado de escribir la nueva clave en el fichero `passwd` de la computadora principal. Una vez realizada esta modificación sobreescribe las copias de este fichero que se hayan en los otros dos computadores. Dado que este proceso modifica las claves, anulando la clave antigua, es necesario que se ejecute siempre con prioridad respecto a los procesos que comprueban la validez de una clave.

(a) Identifique los problemas de programación concurrente que pueden aparecer en el sistema anterior.

(b) Implemente un esquema general de solución del problema anterior indicando básicamente los esquemas de gestión de la concurrencia y la sincronización entre procesos.

(c) Implemente la solución anterior en el sistema operativo Unix mediante el uso de proceso o hilos.

3. Supóngase en un sistema de cómputo la siguiente secuencia de procesos con los instantes de llegada y tiempos de espera que se indican:

Proceso	Instante de llegada	Tiempo de ráfaga
1	0	10 ms
2	4	13 ms
3	9	20 ms
4	12	5 ms
5	18	16 ms

realice los dos ejercicios siguientes utilizando esta tabla de procesos.

(a) Considere que el planificador realiza una planificación Round Robin. Cada vez que el planificador ha de seleccionar un nuevo proceso para su ejecución invierte 2 ms en realizar esta tarea. Además cada cambio de contesto necesita 3 ms. Realice el diagrama de Gant y calcule el tiempo medio de espera si el cuanto asignado a cada proceso es de $q = 6ms$.

(b) Considere un planificador Round Robin cuyo tiempo de planificación y cambio de contexto es despreciable. El planificador implementa un algoritmo RR con un cuanto $q = 8ms$ basado en prioridades dinámicas. Cuando un proceso llega al sistema se le asigna una prioridad 0, cada ms incrementa su prioridad en un valor $\alpha = 2.0$. La prioridad mayor es la numéricamente mayor. Cuando un proceso agota su cuanto el planificador elige el proceso con mayor prioridad. Mientras este proceso está en ejecución su prioridad aumenta cada milisegundo en un valor $\beta = 1.0$. Cuando un proceso retorna a la cola de preparados tras agotar su cuanto mantiene el valor de prioridad.

Realice el diagrama de Gant y calcule el tiempo medio de espera según este algoritmo.

4. Dada la siguiente secuencia de referencias a memoria, de lectura y escritura, y una memoria real de 4 marcos de página y una memoria virtual de 8 páginas:

```
2 1 3 2 4 5 2 3 4 7 6 1 1 6 5 4 0 1 2 7 1 1 3 4 0 2 4
E E L L L E L E E L L E L L E E L E L E L L E E L E L
```

Indique el número de fallos de página si se utiliza como algoritmo de reemplazo:

(a) Indique la secuencia de reemplazo que produciría el número mínimo de fallos de página y calcule este número de fallos.

(b) Indique la secuencia de reemplazo utilizando el algoritmo NRU (puesta a 0 del bit cada 6 accesos) y calcule el número de fallos de página que se producirían.

(c) Indique la secuencia de reemplazo utilizando el algoritmo LRU y calcule el número de fallos de página que se producirían.

(d) Suponiendo que cada fallo de página tiene un tiempo se servicio de 70 ms si la página a reemplazar no ha sido modificada y de 100 ms si la página a reemplazar has sido modificada, calcule el tiempo invertido por cada algoritmo en servir los fallos de página producidos en los dos aparatados anteriores.

5. Un sistema informático organiza su memoria virtual mediante segmentación paginada y la política de obtención de página es la paginación anticipada. Para la traducción de las direcciones virtuales a reales utiliza un esquema mixto con traducción directa mediante tablas de segmento y página y una TLB que traduce directamente $(s, p) \rightarrow (m_p)$ en 0.05 ms. El tiempo medio de acceso a memoria es de 1.1 ms. La traducción se inicia comprobando la TLB, que tiene una tasa de acierto del 85%. Si no se encuentra la traducción en la TLB se inicia el acceso a la tabla de segmentos de donde se toma la dirección base de la tabla de página. En caso de fallo se segmento se tardan 40 ms en crear la tabla de páginas correspondiente al segmento y 60 ms más en cargar la página pedida desde la memoria secundaria. Si se produce una fallo de página es necesario un tiempo de 90 ms hasta que la página que produjo el fallo se encuentre disponible en memoria real. La tasa de fallos de segmento es del 3%, y la tasa de fallos de página, cuando no se ha producido un fallo de segmento, es del 8%.

   (a) Calcule el tiempo medio de acceso a la memoria según el esquema indicado.

   (b) Supóngase en el esquema anterior que la tabla de páginas se encuentra en la memoria virtual del proceso. En ese caso cuando se accede a la tabla de páginas se puede producir un fallo de página con una probabilidad del 5%. Calcule el tiempo medio de acceso a la memoria según este esquema.

NOTA: Justifique siempre todas las respuestas.

Tiempo de realización: **5 horas**.

Cabeceras de las funciones de creación de procesos, memoria compartida, semáforos System V, semáforos Posix e hilos:

```
pid_t fork (void);

key_t ftok (char *pathname, char proj);

int shmget(key_t key, int size, int shmflg);
int shmctl(int shmid, int cmd, struct shmid_ds *buf);
void *shmat(int shmid, const void *shmaddr, int shmflg);
int shmdt(const void *shmaddr);

int semget(key_t key, int nsems, int semflg);
int semop(int semid, struct sembuf *sops, unsigned nsops);
int semctl(int  semid, int semnum, int cmd, union semun arg);

int sem_wait (sem_t *sem);
int sem_post (sem_t *sem);
int sem_init (sem_t *sem, 0, int value);

int pthread_create (pthread_t *id, 0, void * (*f)(void *), void *arg);
void pthread_exit (void *retval);
int pthread_join (pthread_t id, void **retval);
```