Instituto Politécnico Nacional

ESCOM

Sistemas Operativos

David Araujo Diaz

Islas Osorio Enrique
Numero de lista: 20
Boleta:2021630409

4SCM1

Tarea 2: Administración de procesos

Tarea 2: Administración de procesos

1. ¿Qué es el seudoparalelismo?

Un proceso es un programa en ejecución. En un sistema en tiempo compartido cada proceso es ejecutado unos cuantos milisegundos, luego se pasa a otro, y así sucesivamente. A esto se le llama pseudoparalelismo. El mismo procesador cambia de un programa a otro. El sistema operativo es el que simula el paralelismo. Al cambio entre un proceso y otro se le llama multiprogramación.

2. ¿Qué es un proceso?

Un proceso es una instancia de ejecución de un programa, caracterizado por su contador de programa, su palabra de estado, sus registros del procesador, su segmento de texto, pila y datos, etc.

3. Describa en que consiste la jerarquía de procesos.

Desde el punto de vista macro, los procesos son las actividades claves que se requieren para manjar y, o dirigir una organización, es necesario mostrar la jerarquía de proceso en la siguiente grafica.

Esta jerarquía muestra cinco niveles: nivel macroproceso, nivel proceso, nivel subproceso, nivel actividades y nivel de tareas específicas a realizar en un proceso concreto.

Procesos de Dirección

Necesarios para cumplir con la misión y visión de la organización. Proporcionan directrices y lineamientos a los procesos claves.

Procesos misionales

Dan cumplimiento a la razón de ser de la organización. Generan un impacto al cliente creando valor para este.

Procesos de soporte

Aportan elementos de apoyo requeridos para que se puedan desempeñar los procesos Estratégicos y Operativos.

4. Defina qué es el bloque de control del proceso o BCP (*Process Control Block*).

El BCP de informática o Bloque de control del proceso, en ingles PCB (Process Control Block), se refiere a un registro especial donde el sistema operativo agrupa toda la información que necesita conocer respecto a algún proceso en particular. Por tanto, cada vez que un proceso se crea, el sistema operativo crea el BCP, que va a servir como descripción en tiempo de ejecución durante toda la vida del proceso.

Una vez que el BCP es creado, se llena con los atributos definidos como parámetros encontrados en la plantilla del proceso o especificados como parámetros de la llamada al sistema operativo (crear proceso). Después de esto, el sistema operativo por lo general asigna valores a otros campos.

Al terminar el proceso, el BCP creado para este, se borra y el registro se puede usar para otros procesos. Cuando hay un BCP asociado a algún proceso, entonces este será conocido para el sistema operativo y elegible para competir por los recursos del sistema. Este BCP corresponde a una estructura de datos con campos destinados a registrar los diferentes aspectos de la ejecución del proceso y el uso de recursos.

5. ¿Qué incluye el bloque de control del proceso?

La información almacenada en un BCP incluye típicamente algunos o todos los campos siguientes:

Identificador del proceso (Process Identificador -PID-, de sus siglas en inglés).
Estado del proceso. Por ej. listo, en espera, bloqueado.
Contador de Programa: Dirección de la próxima instrucción a ejecutar.
Valores de registro de CPU. Se utilizan también en el cambio de contexto.
Espacio de direcciones de memoria.
Prioridad en caso de utilizarse dicho algoritmo para planificación de CPU.
Lista de recursos asignados (incluyendo descriptores de archivos y sockets abiertos).
Estadísticas del proceso.
Datos del propietario (owner).
Permisos asignados.
Signals pendientes de ser servidos. (Almacenados en un mapa de bits)

6. Detalle las operaciones que se pueden realizar con los procesos

Crear: puede hacerse desde un proceso ya existente o a través del intérprete de comandos del S.O. en cualquier caso se considera hijo del proceso creador. En cualquier caso la creación de procesos se hace mediante una llamada al sistema (fork en UNIX y execve).
Destruir: eliminar la entrada en la cola de PCB. Puede haber problemas en la gestión de las propiedades heredadas del proceso padre o, si tiene procesos hijo, tener que esperar a que finalicen estos o los finaliza forzosamente.
Terminación normal. Es la forma más normal (exit en UNIX)
Terminación por error. Por ejemplo gcc uno.c y el fichero uno.c no existe.
Error fatal. Acceso a una posición no permitida, división por cero … etc.
Terminado por otro proceso. En UNIX es KILL.
Cambiar la prioridad del proceso.
Dormir o bloquear la ejecución de un proceso. Dormir un proceso un tiempo.
Despertar un proceso. Una forma de desbloquear un proceso de forma artificial. Se suele emplear para procesos dormidos artificialmente.
Suspender un proceso. Suele hacerse en situaciones de sobrecarga del S.O.
Reanudar un proceso. Activar un proceso suspendido

7. ¿Cuáles son las opciones de finalización de un proceso?

Terminación normal. Es la forma más normal (exit en UNIX)
Terminación por error. Por ejemplo gcc uno.c y el fichero uno.c no existe.
Error fatal. Acceso a una posición no permitida, división por cero …
etc.
Terminado por otro proceso. En UNIX es KILL.

8. ¿Cuáles son los tres estados comunes en los que se puede encontrar un proceso y cuáles son sustransiciones?

Los posibles estados que puede tener un proceso son ejecución, bloqueado y listo:

Ejecución, es un proceso que está haciendo uso del procesador.
Bloqueado, No puede ejecutarse hasta que un evento externo sea llevado a cabo.
Listo, ha dejado disponible al procesador para que otro proceso pueda ocuparlo.
Las posibles transiciones son 4. La primera se realiza cuando el sistema operativo determina que el proceso no puede continuar justo en ese momento, en algunos sistemas se puede hacer una llamada al sistema "pause" para pasar al estado bloqueado, en Unix cuando el proceso está leyendo datos provenientes de una canalización o de un archivo especial (terminal) y no hay entrada disponible, el proceso se bloquea de forma automática.

Las transiciones 2 y 3 son llevadas a cabo por el planificador de procesos, siendo que el proceso no tiene conocimiento de este. La transición 2 se da cuando el planificador de procesos decide que el proceso ya estuvo el tiempo suficiente en ejecución y debe dar paso a la ejecución de otros procesos (adquieran tiempo del procesador). La transición 3 se realiza cuando todos los procesos han ocupado tiempo del procesador y debe retomarse el primer proceso.

La transición 4 ocurre cuando se produce un evento externo por el que un proceso estaba en espera, por ejemplo, introducir datos desde la terminal. Si no hay otro proceso en ejecución en ese instante, la transición 3 se activa y el proceso comienza a ejecutarse; también podría pasar al estado de "listo" y esperar un momento para iniciar la ejecución.

9. Describa como se realiza la implementación de procesos, mediante una tabla de procesos y vectoresde interrupción.

El vector de interrupciones es un vector que contiene el valor que apunta a la dirección en memoria del gestor de una interrupción. En muchas arquitecturas de computación típicas, los vectores de interrupción se almacenan en una tabla en una zona de memoria, la llamada tabla de vectores de interrupción, de modo que cuando se atiende una petición de interrupción de número n, el sistema, tras realizar eventualmente algunas tareas previas (tales como salvar el valor de ciertos registros) transfiere el control a la dirección indicada por el elemento n-ésimo de dicha tabla.

Int. Num.	Address in I.V.T.	Description
0	00-03	CPU divide by zero
1	04-07	Debug single step
2	08-0B	Non Maskable Interrupt (NMI input on processor)
3	0C-0F	Debug breakpoints
4	10-13	Arithmetic overflow
5	14-17	BIOS provided Print Screen routine
6	18-1B	Reserved
7	1C-1F	Reserved
8	20-23	IRQ0, Time of day hardware services
9	24-27	IRQ1, Keyboard Interface
A	28-2B	IRQ2, ISA Bus cascade services for second 8259
B	2C-2F	IRQ3, Com 2 hardware
C	30-33	IRQ4, Com1 hardware

10. ¿Cuál es la diferencia entre un hilo y un proceso?

Todos los hilos de un programa están contenidos lógicamente dentro de un proceso.

Un proceso es pesado, pero un hilo es ligero.

Un programa es una unidad de ejecución aislada mientras que el hilo no está aislado y comparte memoria.

Un hilo no puede tener una existencia individual; Se adjunta a un proceso. Por otro lado, un proceso puede existir individualmente.

En el momento de la expiración de un hilo, su pila asociada podría recuperarse, ya que cada hilo tiene su propia pila. En contraste, si un proceso muere, todos los hilos mueren, incluido el proceso.

Bases para la comparación	Proceso	Hilo
BASIC	Programa en ejecución.	Proceso ligero o parte de él.
Compartir la memoria	Completamente aislado y no compartir memoria.	Comparte la memoria entre sí.
Consumo de recursos	Más	Menos
Eficiencia	Menos eficiente en comparación con el proceso en el contexto de la comunicación.	Mejora la eficiencia en el contexto de la comunicación.
Tiempo requerido para la creación.	Más	Menos
Tiempo de cambio de contexto	Toma mas tiempo	Consume menos tiempo.
Terminación incierta	Resultados en la pérdida del proceso.	Un hilo puede ser reclamado.
Tiempo requerido para la terminación	Más	Menos

11. ¿Qué son los hilos?

El hilo es una ejecución de programa que utiliza recursos de proceso para realizar la tarea. Todos los hilos dentro de un solo programa están contenidos lógicamente dentro de un proceso. El núcleo asigna una pila y un bloque de control de hilos (TCB) a cada hilo. El sistema operativo solo guarda el puntero de pila y el estado de la CPU en el momento de cambiar entre los subprocesos del mismo proceso.

12. Liste los estados en los que se puede encontrar un hilo.

Un hilo puede ser en alguno de los siguientes estados:

Listo: el hilo puede ser elegido para su ejecución.
Standby: el hilo ha sido elegido para ser el siguiente en ejecutarse en el procesador.
Ejecución: el hilo está siendo ejecutado.
Espera: un hilo pasa a este estado cuando se bloquea por un suceso
(E/S): se realiza una espera voluntaria de sincronización o alguien suspende al hilo.
Transición: después de una espera el hilo pasa a este estado si está listo para ejecutar pero alguno de sus recursos no está disponible aún.
Terminado: un hilo llega a este estado cuando termina normalmente cuando su proceso padre ha terminado.

13. ¿Por qué es necesaria la comunicación entre procesos?

La comunicación entre procesos es una función básica de los sistemas operativos que provee un mecanismo que permite a los procesos comunicarse y sincronizarse entre sí, normalmente a través de un sistema de bajo nivel de paso de mensajes que ofrece la red subyacente.

La comunicación entre procesos puede estar motivada por la competencia o el uso de recursos compartidos o porque varios procesos deban ejecutarse sincronizadamente para completar una tarea.

14. Defina la memoria compartida.

La memoria compartida es aquel tipo de memoria que puede ser accedida por múltiples programas, ya sea para comunicarse entre ellos o para evitar copias redundantes. La memoria compartida es un modo eficaz de pasar datos entre aplicaciones. Dependiendo del contexto, los programas pueden ejecutarse en un mismo procesador o en procesadores separados. La memoria usada entre dos hilos de ejecución dentro de un mismo programa se conoce también como memoria compartida.

15. ¿Qué es el paso de mensajes?

El paso de mensajes es un término que se utiliza para identificar un tipo de proceso de comunicación que se utiliza en varios procesos diferentes, incluida la programación orientada a objetos, la comunicación entre procesos y la computación en paralelo.

16. Diga que es una tubería.

Una tubería (pipe, cauce o '|') consiste en una cadena de procesos conectados de forma tal que la salida de cada elemento de la cadena es la entrada del próximo. Permiten la comunicación y sincronización entre procesos. Es común el uso de buffer de datos entre elementos consecutivos.
En informática, una tubería (pipeline o cauce) consiste en una cadena de procesos conectados de forma tal que la salida de cada elemento de la cadena es la entrada del próximo. Permiten la comunicación y sincronización entre procesos. Es común el uso de búfer de datos entre elementos consecutivos.

La comunicación por medio de tuberías se basa en la interacción productor/consumidor, los procesos productores (aquellos que envían datos) se comunican con los procesos consumidores (que reciben datos) siguiendo un orden FIFO. Una vez que el proceso consumidor recibe un dato, este se elimina de la tubería.

17. ¿Cuáles son las características de una tunería?

Las tuberías están implementadas en forma muy eficiente en los sistemas operativos multitarea, iniciando todos los procesos al mismo tiempo, y atendiendo automáticamente los requerimientos de lectura de datos para cada proceso cuando los datos son escritos por el proceso anterior. De esta manera el planificador de corto plazo va a dar el uso de la CPU a cada proceso a medida que pueda ejecutarse minimizando los tiempos muertos.

Para mejorar el rendimiento, la mayoría de los sistemas operativos implementan las tuberías usando búferes, lo que permite al proceso proveedor generar más datos que lo que el proceso consumidor puede atender inmediatamente.

Tubería sin nombre

Las tuberías sin nombre tienen asociado un fichero en memoria principal, por lo tanto, son temporales y se eliminan cuando no están siendo usados ni por productores ni por consumidores. Permiten la comunicación entre el proceso que crea un cauce y procesos hijos tras la creación de la tubería.

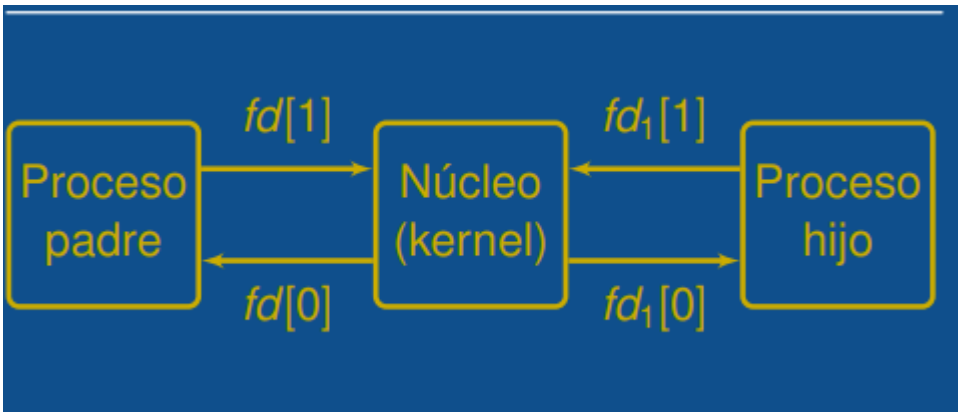
Tubería con nombre

Su diferencia respecto a las tuberías sin nombre radica en que el cauce se crea en el sistema de archivos, y por lo tanto no tienen carácter temporal. Se manejan mediante llamadas al sistema (open, close, read y write) como el resto de los ficheros del sistema. Permiten la comunicación entre los procesos que usen dicha tubería, aunque no exista una conexión jerárquica entre ellos.

18. Mencione un ejemplo de tubería.

Tubería sin nombre

```
#include <unistd.h>
int pipe(int fildes [2]);
```



Solo el proceso que hace la llamada y sus descendientes pueden utilizarla

19. Mencione algunas de las razones para sincronizar procesos/hilos.

La razón más común para la sincronización es cuando dos o más hilos necesitan acceso a un recurso compartido que solo puede ser utilizado por un hilo a la vez.

20. ¿Qué es una condición de carrera?

Una condición de carrera es una situación indeseable que ocurre cuando un dispositivo o sistema intenta realizar dos o más operaciones al mismo tiempo, pero debido a la naturaleza del dispositivo o sistema, las operaciones deben realizarse en la secuencia adecuada para que se realicen correctamente.

21. Defina qué es una sección crítica.

Se denomina sección crítica, en programación concurrente, a la porción de código de un programa de ordenador en la que se accede a un recurso compartido (estructura de datos o dispositivo) que no debe ser accedido por más de un proceso o hilo en ejecución. La sección crítica por lo general termina en un tiempo determinado y el hilo, proceso o tarea sólo tendrá que esperar un período determinado de tiempo para entrar. Se necesita un mecanismo de sincronización en la entrada y salida de la sección crítica para asegurar la utilización en exclusiva del recurso, por ejemplo un semáforo.

22. ¿Qué es la exclusión mutua?

Consiste en que un solo proceso excluye temporalmente a todos los demás para usar un recurso compartido de forma que garantice la integridad del sistema.

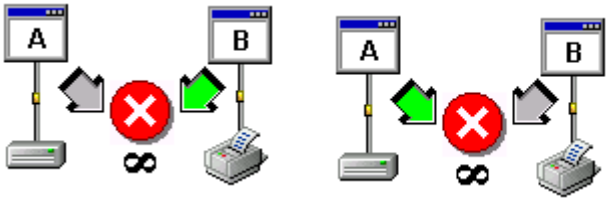
23. ¿Cómo define la atomicidad?

La atomicidad es una característica de los sistemas de bases de datos que dicta dónde una transacción debe ser todo o nada. Es decir, la transacción debe suceder completamente o no suceder en absoluto. No

debe completarse parcialmente.

24. Describa un ejemplo de abrazo mortal.

Los abrazos mortales pueden también involucrar diferentes tipos de recursos. Por ejemplo, considere un sistema con una impresora y una unidad de disco. Suponga que el proceso A tiene asignada la unidad de disco y que el proceso B tiene asignada la impresora. Ahora, si A pide la impresora y B pide la unidad de disco, ocurre un abrazo mortal.



25. ¿Qué es la exclusión mutual con espera activa?

La espera activa es una técnica en la cual un proceso comprueba continuamente si una condición se cumple o si se produce un evento

26. ¿Qué es la exclusión mutual con espera pasiva?

Los procesos efectúan un ciclo continúo hasta que pueden continuar, lo que representa un problema en sistemas multiprogramados. Espera pasiva La operación de bloqueo coloca un proceso en una cola de espera asociada al mecanismo de sincronización y transfiere el control al planificador.

27. Mencione las cuatro condiciones para evitar condiciones de competencia, empleando secciones críticas.

1. Dos procesos no deben encontrarse al mismo tiempo dentro de sus secciones críticas .

2. No se deben hacer hipótesis sobre la velocidad o el número de UCP.

3. Ninguno de los procesos que estén en ejecución fuera de su sección crítica puede bloquear a otros procesos.

4. Ningún proceso debe esperar eternamente para entrar a su sección crítica

28. ¿Qué son los semáforos?

Los semáforos son una herramienta de sincronización que ofrece una solución al problema de la sección crítica (porción de código de un programa de computador en la cual se accede a un recurso compartido que no debe ser accedido por más de un proceso o hilo en ejecución). Un semáforo provee una simple pero útil abstracción para controlar el acceso de múltiples procesos a un recurso común en programación paralela, o entornos multiusuarios. El concepto de semáforo fue inventando por el holandés Esdger W. Dijkstra.

29. ¿Cómo se resuelve el problema del productor consumidor con semáforos?

El uso de semáforos resuelve el problema de las llamadas perdidas. En la solución que se muestra a continuación se usan dos semáforos fillCount y emptyCount. El primero es el número de objetos que existen en el buffer mientras que emptyCount es el número de espacios disponibles donde se puede insertar objetos. fillCount es incrementado y emptyCount es decrementado cuando se inserta un objeto en el buffer. Si el productor intenta decrementar emptyCount cuando su valor es cero, el productor es puesto a dormir. La próxima vez que se consuma un objeto el productor despierta. El consumidor funciona análogamente.

30. ¿Qué es un monitor y como resuelve el problema del productor consumidor?

Un monitor es una estructura del lenguaje cuyas principales características son:

Los datos son privados.

Ofrecen una serie de métodos públicos para acceder a dichos datos.

En cada momento sólo puede haber un proceso activo en algún método del monitor, es decir, ejecutando código de esos métodos públicos del monitor. Sería equivalente a decir que el recurso que queremos compartir se declara monitor. Los procesos que usan el monitor son independientes unos de otros y cuando deseen usar el recurso, llamarán a los métodos del monitor que implementen la operación que se desea ejecutar.

Permiten organizar procesos en espera mediante:

Dado que la exclusión mutua está implícita en los monitores, no es necesario ningún esfuerzo adicional para proteger la sección crítica. En otras palabras, la solución que se muestra a continuación funciona con cualquier número de productores y consumidores sin ninguna modificación. También es digno de

mención que es menos probable que un programador escriba código que sufra condiciones de carrera cuando usa monitores que cuando usa semáforos.

31. ¿Qué es la planificación de procesos?

La planificación de procesos es una herramienta para que el sistema operativo determine el orden en que se adecua el procesador a los procesos que lo vayan necesitando y a las políticas que se utilizarán en la eficiencia del tiempo esperado en el sistema.

32. ¿Cuáles son los criterios para tener un buen algoritmo de planificación?

Utilización de CPU (CPU utilization): Es el porcentaje de uso (en cuanto a ejecución de tareas de usuario o del sistema que son consideradas útiles) que tiene un procesador.

– Rendimiento (Throughput): Es el número de procesos que ejecutaron completamente por unidad de tiempo (una hora p.ej.).

– Tiempo de retorno (Turnaround time): Es el intervalo de tiempo desde que un proceso es cargado hasta que este finaliza su ejecución.

– Tiempo de espera (Waiting time): Es la suma de los intervalos de tiempo que un proceso estuvo en la cola de procesos listos (ready queue).

– Tiempo de respuesta (Response time): Es el intervalo de tiempo desde que un proceso es cargado hasta que brinda su primer respuesta. Es útil en sistemas interactivos.

33. ¿Qué es la planificación expropiativa y la no expropiativa?

Sistemas operativos con planificadores no expropiativos (nonpreemptive) son los que asignan el recurso procesador a un proceso y hasta que este no lo libere, ya sea porque finaliza su ejecución o se bloquea, no se vuelve a ejecutar el planificador.

・Sistemas operativos con planificadores expropiativos (preemptive) son que los que pueden expropiar el recurso procesador a un proceso cuando otro proceso entra en estado pronto (ya sea porque es nuevo o porque se desbloqueó) o porque se le impone un límite de tiempo para ejecutar.

34. Describa la planificación Round Robin?

Round-robin es un método para seleccionar todos los abstractos en un grupo de manera equitativa y en un orden racional, normalmente comenzando por el primer elemento de la lista hasta llegar al último y empezando de nuevo desde el primer elemento. El nombre del algoritmo viene del principio de Round-Robin conocido de otros campos, donde cada persona toma una parte de un algo compartido en cantidades, es decir, "toma turnos". En operaciones computacionales, un método para ejecutar diferentes procesos de manera concurrente, para la utilización equitativa de los recursos del equipo, es limitando cada proceso a un pequeño período (quantum), y luego suspendiendo este proceso para dar oportunidad a otro proceso y así sucesivamente. A esto se le denomina comúnmente como Planificación Round-Robin.

Este algoritmo de planificación, conocido por Round robin, está diseñado especialmente para sistemas de tiempo compartido. Se define un intervalo de tiempo denominado "Quantum", cuya duración varía según el sistema. La cola de procesos se estructura como una cola circular. El

planificador la recorre asignando un cuanto de tiempo a cada proceso. La organización de la cola es FIFO. El Quantum se suele implantar mediante un temporizador que genera una interrupción cuando se agota el Quantum de tiempo. Si el proceso agota su ráfaga de CPU antes de finalizar el Quantum, el planificador asigna la CPU inmediatamente a otro proceso. Este algoritmo tiene un tiempo de espera relativamente grande. Sin embargo, garantiza un reparto de la CPU entre todos los usuarios y arroja tiempos de respuesta buenos. Como ejemplo, supongamos los siguientes tres procesos en un instante en el sistema:

35. ¿Qué es la planificación por prioridad?

Esta planificación se caracteriza porque a cada proceso se le asigna una prioridad y se continúan con un criterio determinado. Los procesos serán atendidos de acuerdo con la prioridad determinada.

36. ¿Qué es la planificación por colas múltiples?

Derivado de MQS (Multilevel Queue Scheduling). Es un algoritmo donde la cola de procesos en estado de listos se divide en varias colas más pequeñas. Los procesos se clasifican a partir de un criterio que determina en qué cola se ubicará el proceso cuando se encuentre en estado de listo.

37. ¿Cómo funciona la planificación por el trabajo más corto?

Conocido como SJF (Shortest Job First). Este proceso se distingue porque cuando un proceso se encuentra en ejecución, voluntariamente cambia de estado, es decir que el tiempo de ejecución del proceso no es determinado. Por lo cual cada proceso tiene una asignación de tiempo cuando vuelve a ser ejecutado y va ejecutando el proceso con la menor cantidad de tiempo asignada. Al encontrarse que dos algoritmos poseen la misma cantidad de tiempo, se utilizará el algoritmo FCFS.

38. ¿Qué es la planificación garantizada?

En esta planificación el sistema se enfoca en la cantidad de usuarios que debe atender. Donde en un número n de usuarios se asignará a cada usuario 1/n de tiempo de ejecución.

39. Describa la planificación por lotería.

Es de carácter aleatorio.
Cada vez que aparezca un proceso nuevo, se le conceden boletos, con lo que ya tendrá una probabilidad de ganar proporcional al número de boletos recibidos.
Este esquema de planificación es de resultados comparativamente rápidos en relación a lo pretendido con las reparticiones de boletos, pues tarda solamente hasta el próximo sorteo.
Se pueden dar más boletos a los procesos más importantes, a fin de aumentar sus posibilidades de ganar.
En procesos cooperativos pueden intercambiar boletos entre sí como en servicios cliente-servidor donde el cliente puede prestarle temporalmente al servidor sus boletos para una mayor posibilidad de ganar tiempos.
En este tipo de planificación, la prioridad queda determinada por la cantidad de boletos asignados a un proceso, que influirán estadísticamente de acuerdo con dicha proporción.

40. Mencione en qué consiste la planificación en tiempo real.

La principal característica que distingue a los STR de otros sistemas es el tiempo, para ello es importante saber que: La palabra tiempo significa que el correcto funcionamiento de un sistema depende no sólo

del resultado lógico de la computadora, también depende del tiempo en que se produce ese resultado. La palabra real quiere decir que la reacción de externos un sistema a eventos debe ocurrir durante su evolución. Como consecuencia, el tiempo del sistema (interno) debe ser medido usando la misma escala con que se mide el tiempo del ambiente controlado (externo).

En este sentido un STR cumple tres condiciones primordiales que se representan en la figura 1. [MGC07]

1. Contacto con el mundo físico,
2. Emisión de respuestas correctas,
3. Obtención de respuestas dentro de intervalos de tiempo establecidos.

41. ¿Cómo se planifican los procesos en UNIX?

El planificador siempre selecciona al proceso que encontrándose en el estado preparado en memoria principal para ser ejecutado o en el estado expropiado tiene la mayor prioridad. En el caso de los procesos de igual prioridad (se encuentran en la misma cola) lo que hace es ceder el uso de la CPU a uno de ellos durante un cuanto, cuando finaliza dicho cuanto le expropia la CPU y se lo cede a otro proceso. El planificador varía dinámicamente la prioridad de los procesos basándose en su tiempo de uso de la CPU. Si un proceso de mayor prioridad alcanza el estado preparado en memoria principal para ser ejecutado, el planificador expropia el uso de la CPU al proceso actual incluso aunque éste no haya completado su cuanto.

El núcleo tradicional de UNIX es estrictamente no expropiable. Es decir, si el proceso actual se encuentra en modo núcleo (debido a una llamada al sistema o a una interrupción), no puede ser forzado a ceder la CPU a un proceso de mayor prioridad.

Dicho proceso cederá voluntariamente la CPU cuando entre en el estado dormido. En caso contrario sólo se le podrá expropiar la CPU cuando retorne a modo usuario.

p_pri. Contiene la prioridad de planificación actual.

p_usrpri. Contiene la prioridad de planificación actual en modo usuario.

p_cpu. Contiene el tiempo transcurrido desde que el proceso utilizó por última vez la CPU, también denominado uso reciente de la CPU.

p_nice. Contiene el factor de amabilidad, que es controlable por el usuario.

42. ¿Cómo se planifican los hilos?

La planificación de ejecución de threads se basa en el modelo de prioridades y no utiliza el modelo de segmentación por segmentos de tiempo. Un thread continuará ejecutandose en la CPU hasta pasar a un estado que no le permita seguir en ejecución.

43. ¿Cuáles son las políticas de desalojo de proceso/hilos en un sistema operativo?

Las políticas de planificación de procesos surgen como necesidad dada por los objetivos perseguidos por el SO: multiprogramación y tiempo compartido.

Objetivo de la multiprogramación:

Tener algún proceso en ejecución en todo momento (maximizar el aprovechamiento de la CPU).

Objetivo del tiempo compartido:

Conmutar la CPU entre procesos con tal frecuencia que los usuarios puedan interactuar con cada programa durante su ejecución.

44. ¿Qué algoritmos se usan para planificar hilos?

Los algoritmos de planificación no hacen distinción entre procesos ligeros y pesados, ya que para ellos solo existen los procesos en su concepción general, de ahí que en la cola de listos se puedan encontrar procesos de ambos tipos, y el planificador de periodo corto escogerá alguno sin hacer distinciones entre uno u otro.

Las políticas de planificación de los SO pueden ser:

* Sin desalojo. En este tipo de política, a un proceso que está en el estado ejecutando no se le retira la CPU mientras esté usándola. El planificador de periodo corto solo entra en acción cuando el proceso que tiene el procesador necesita hacer una entrada/salida o cuando termina. Los SO Windows 3.x usaban esta forma de planificar.

*Con desalojo. Este segundo tipo de política permite que el SO le retire el procesador al proceso que lo tenga. De tal forma, a los dos momentos anteriores para planificar un nuevo proceso (terminó, tuvo que hacer una entrada/salida), se le agrega una tercera que es cuando se le quita el procesador al proceso que lo tiene, a pesar de que necesitaba más tiempo. Los motivos para quitarle el procesador pueden ser varios, por ejemplo: se le había asignado el procesador por un tiempo y ese tiempo terminó, llegó un proceso de mayor prioridad, etc.\

Referencias

S. (2022). Fundamentos De Sistemas Operativos (7.a ed.). MCGRAW HILL EDDUCATION.

ENGINEERING THE WORLD FROM PARAGUAY. (2011, 31 marzo). Sobrevolando los Vectores de Interrupciones. Rama Estudiantil del IEEE de la UCSA. https://ramaucsa.wordpress.com/2011/03/31/2304/

ARQUITECTURA DE COMPUTADORAS II Interrupciones en la IBM PC Carlos Canto Q. (s. f.). Slidetoc. Recuperado 28 de febrero de 2022, de https://slidetodoc.com/arquitectura-de-computadoras-ii-interrupciones-en-la-ibm/

colaboradores de Wikipedia. (2020, 7 septiembre). Proceso (informática). Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Proceso_(inform%C3%A1tica)#:%7E:text=El%20proceso%20es%20la%20actividad,programa%20en%20ejecuci%C3%B3n%202%20veces.

Planificación de procesos de tiempo real - Wiki de Sistemas Operativos. (s. f.). wiki. https://1984.lsi.us.es/wiki-ssoo/index.php/Planificaci%C3%B3n_de_procesos_de_tiempo_real