Proyecto semestral
Sistema Operativos

Integrantes: Rodolfo Cuevas

Profesor: Carlos Faúndez

Fecha entrega: 17-08-2020

Índice

Contenido

Introducción

Este proyecto se llevó a cabo utilizando una jerarquía de procesos donde el proceso padre es quien pausa al hijo después de crearlo y lo guarda en una Cola circular para después terminar ejecutando una función distinta dependiendo de lo que se le haya pasado por parámetro (-F -R -P) explicados a continuación para realizar un orden de ejecución distinto para el procesamiento de los hijos que cuya misión será limitada a ejecutar un archivo loop que genera un sleep() aleatorio entre 1~10.

-F, representa a FIFO que procesa en el orden en el que han sido ingresado los datos. El primer proceso en entrar es el primero en salir.

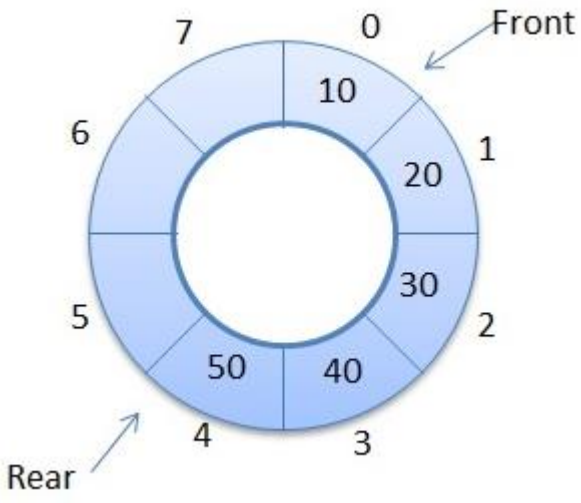
-R, representa a RoundRobin el cual basado en un Quantum(Q) ingresado por parámetros de consola turna los tiempos de ejecución de cada proceso y los va finalizando cuando termina la ejecución de loop.

-P, representa a planificación por prioridades no expulsivo el cual requiere de un valor de prioridad para cada proceso, para así poder ejecutarlos en un orden de prioridad.

Estructura de datos utilizada

La estructura de datos utilizada para el almacenamiento de los procesos elegida fue la Cola circular por su facilidad para la ejecución en RR ya que al terminar de ser recorrida vuelve al principio. Esto presentó una dificultad para trabajar con los espacios en el A. de planificación no expulsivo debido a que dependiendo de la prioridad ingresada para cada proceso puede darse la condición en que requiera eliminar un proceso que se encuentre entremedio de la Cola.

Eso se solucionó guardando los pid() de los procesos en un arreglo para luego vaciar la cola y volverla a llenar omitiendo el pid() del proceso que se está eliminando.



Uso de señales

En un principio se usa la señal (SIGSTOP) para detener la ejecución de los procesos hijos inmediatamente después han sido creados. Luego, se va ocupando (SIGCONT) de una forma distinta en cada función según corresponda siendo siempre el fin continuar el proceso hijo pausado.

Se creó una alarma(SIGALRM) para el manejo de los tiempos en Round Robin a la cual se le asigna una función a través de signal, la función que se le asigna cambia el estado de una variable booleana a true. Cuando se define la alarma con un tiempo como parámetro llamará a la función que se asignó a la función anterior, al llamarla cambia la variable booleana a true. De este modo se pudo controlar los tiempos de ejecución en CPU y crear una señal que se mande cuando sea momento de detener el proceso y pasar a la ejecución del siguiente.

Especificación de A. de planificación

-F: Ejecutará el algoritmo de planificación FIFO que hace que cada proceso se ejecute en el orden que fueron ingresados, es decir, el primero que sale es el primero en salir. Luego, pasará al segundo proceso y así hasta completar la cantidad ingresada por argumento de consola después de l -F.

-R: Ejecutará el algoritmo de planificación Round Robin que hace que mediante un Quantum (Q) ingresado por parámetros de consola [-R 5 -Q 2] ejecutará el proceso en CPU por la cantidad de segundos que tenga Q, siguiente a esto se detendrá para dar paso al siguiente proceso el cual pasa a estar activo por un periodo de tiempo de la misma manera que lo hace el primer proceso. Al terminar cada proceso hijo de ejecutar el programa loop se detienen y dan paso a los procesos restante para que puedan terminar su ejecución.

-P: Ejecutará el algoritmo de planificación por prioridades no expulsivo el cual requiere un orden de prioridades ingresado por parámetros de consola igual a la cantidad de procesos requeridos [-P 5 -p 5 4 3 2 1], es decir, un valor de prioridad para cada proceso el cual será ingresado.

Conclusión

En un principio y a bajo número de procesos no se puede apreciar un mayor desempeño en algún algoritmo de planificación respecto a otros. Lo que no significa que en una condición más extrema con un mayor numero de procesos y una función a ejecutar que no sea solamente un loop() que exista una fuerte diferencia entre los tiempos de ejecución totales para cada grupo de procesos.

Pruebas

Algoritmo Round Robin

```
rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$ ./xd -R 2 -Q 7
Proceso(7139) agregado en la cola.(1)
Padre ha pausado el Proceso(7139)
Proceso(7140) agregado en la cola.(2)
Padre ha pausado el Proceso(7140)

Se procederá a usar el algoritmo Round Robin con 2 procesos y con Q = 7.
Se ha activado el proceso(7139)
Loop(10)
Turno en CPU completado.(◔ ʖ ◔)
Se ha dormido el proceso(7139)


Se ha activado el proceso(7140)
Loop(5)
-------------------------Proceso(7140) terminado en 5 segundos.
Turno en CPU completado.(◔ ʖ ◔)
Se ha dormido el proceso(7140)
[PID BORRADO(7139)]


Se ha activado el proceso(7140)
Turno en CPU completado.(◔ ʖ ◔)
Se ha dormido el proceso(7140)
[PID BORRADO(7140)]


rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$
```

Algoritmo FIFO

```
rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$ ./xd -F 3
Proceso(6874) agregado en la cola.(1)
Padre ha pausado el Proceso(6874)
Proceso(6875) agregado en la cola.(2)
Padre ha pausado el Proceso(6875)
Proceso(6876) agregado en la cola.(3)
Padre ha pausado el Proceso(6876)

Se procederá a usar el algoritmo FIFO con 3 procesos.
Se ha activado el proceso(6874)
Loop(1)
-------------------------Proceso(6874) terminado en 1 segundos.
Se ha dormido el proceso(6874)
[PID BORRADO(6874)]


Se ha activado el proceso(6875)
Loop(1)
-------------------------Proceso(6875) terminado en 1 segundos.
Se ha dormido el proceso(6875)
[PID BORRADO(6875)]


Se ha activado el proceso(6876)
Loop(9)
-------------------------Proceso(6876) terminado en 9 segundos.
Se ha dormido el proceso(6876)
[PID BORRADO(6876)]


rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$
```

Algoritmo Prioridades

```
rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$ ./xd -P 3 -p  15 1 7
Proceso(7650) agregado en la cola.(1)
Padre ha pausado el Proceso(7650)
Proceso(7651) agregado en la cola.(2)
Padre ha pausado el Proceso(7651)
Proceso(7652) agregado en la cola.(3)
Padre ha pausado el Proceso(7652)

Se procederá a usar el algoritmo Prioridades con 3 procesos y prioridades de [15][1][7]
Vector de prioridades: [15][1][7]
La prioridad mayor actual es de: Se ha activado el proceso(7650)
Loop(1)
--------------------------Proceso(7650) terminado en 1 segundos.
Se ha dormido el proceso(7650)
[PID BORRADO(7650)]



Vector de prioridades: [-1][1][7]
La prioridad mayor actual es de: Se ha activado el proceso(7652)
Loop(6)
--------------------------Proceso(7652) terminado en 6 segundos.
Se ha dormido el proceso(7652)
[PID BORRADO(7652)]



Vector de prioridades: [-1][1][-1]
La prioridad mayor actual es de: Se ha activado el proceso(7651)
Loop(4)
--------------------------Proceso(7651) terminado en 4 segundos.
Se ha dormido el proceso(7651)
[PID BORRADO(7651)]


rodolfocuevas@pop-os:~/Escritorio/ProyectoSistemasOperativos/Código$
```

Observación: (Cuando se ejecuta muchas veces empieza a arrojar problemas de memoria Ram).