DEPARTAMENTO
DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico I

Scheduling

Sistemas Operativos
Segundo Cuatrimestre de 2015

Integrante	LU	Correo electrónico
Arístides Catalano	279/10	`dilbert_cata@hotmail.com`
Laura Muiño	399/11	`mmuino@dc.uba.ar`
Jorge Quintana	344/11	`jorge.quintana.81@gmail.com`

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

0.1. Ejercicio 1

El ejercicio consiste en programar una tarea que simule n llamadas bloqueantes (con n pasado por parámetro) donde la duración de cada llamada bloqueante está determinada por los parámetros bmin y bmax.

Para generar las duraciones al azar dentro del rango [bmin, bmax], se utilizó la funcion rand de la libreria stdlib y se realizó lo siguiente:

```
    int cant = bmax - bmin + 1;
int cant_ciclos_bloqueada;
cant_ciclos_bloqueada = bmin + (rand() % cant);
```

De esta forma se garantiza que *cant_ciclos_bloqueada* esté entre el rango pedido. Para realizar la simulación de las llamadas bloqueantes se ultilizó la función uso_IO provista por la cátedra.

Para probar la tarea se crearon los siguientes lotes:

- TaskConsola 5 10 20
- TaskConsola 15 2 20

En la primera tarea se eligieron esos parámetros para comprobar que la cantidad de tiempo bloqueado respetara los límites requeridos, dado que la primera simulación no permitía apreciar gráficamente a simple vista la variabilidad dentro del rango, se creo la segunda tarea, con un rango más amplio para que el diagrama de Gantt manifieste dicha variabilidad en los tiempos de bloqueo random de la tarea.

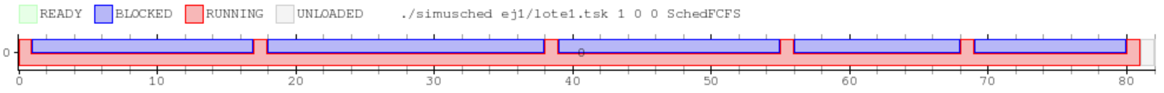

Figura 1:

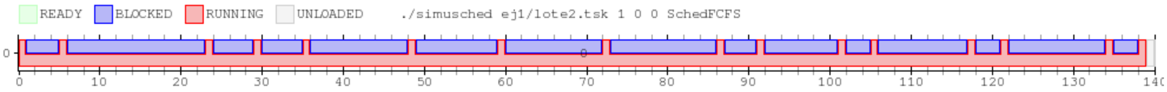

Figura 2: Figura 2.

0.2. Ejercicio 2

Para representar el uso de 100 ciclos de CPU y dos tareas bloqueantes definimos el siguiente lote de tareas:

TaskCPU
TaskConsola 20 2 4
TaskConsola 25 2 4

TaskConsola realizará 20 y 25 llamadas bloqueantes respectivamente, con cada bloqueo de duración variable entre 2 y 4 ciclos. El cambio de contexto se fijo a 4 ciclos. En este ejercicio, el enunciado no aclaraba nada de costos de migración, con lo cual lo mantuvimos en cero. Estos fueron los gráficos obtenidos :

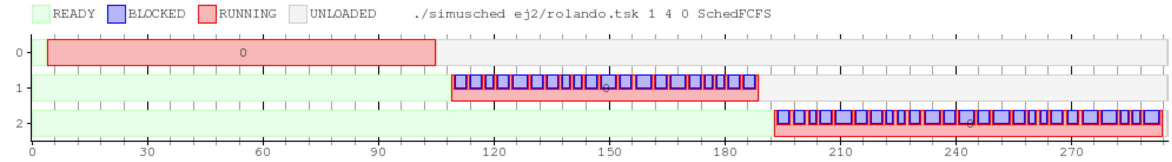

Figura 3:

La latencia es el tiempo que un proceso tarda en empezar a ejecutarse. Con un núcleo, la latencia para las tareas será la suma entre el costo de cambio de contexto y la duración de la tarea anterior. En el caso de la primer tarea (la cero) su latencia es 4. La segunda tarea (la uno) tendrá latencia $4 + 100 + 4 = 108$. Por último, la tarea 2 tendra latencia $108 + duracion\_tarea1 + 4$.

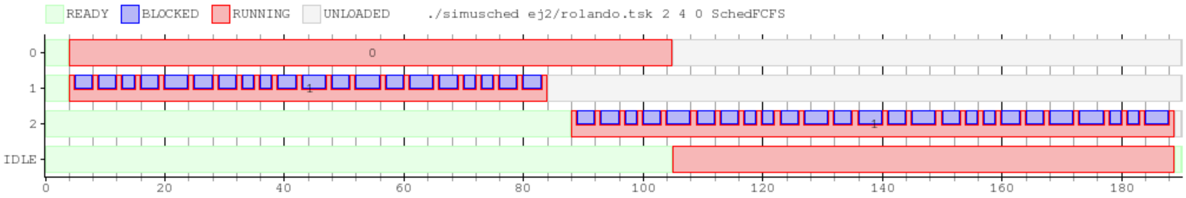

Figura 4:

En cambio, con dos núcleos, la tarea cero tiene latencia 4, y las tareas uno y dos tienen 4 y duracion_tarea1 + 4 respectivamente.

0.3. Ejercicio 3

Para realizar la tarea TaskBatch se generaron cant_bloqueos números distintos aleatoriamente dentro del rango [0, total_cpu -1]. Al hacerlo se tuvo en cuenta el hecho de que se pueden obtener números repetidos. Luego, si el momento coincidia con el número generado al azar, se llamaba a la funcion que simula los bloqueos (uso_IO).

La tarea generada es la siguiente

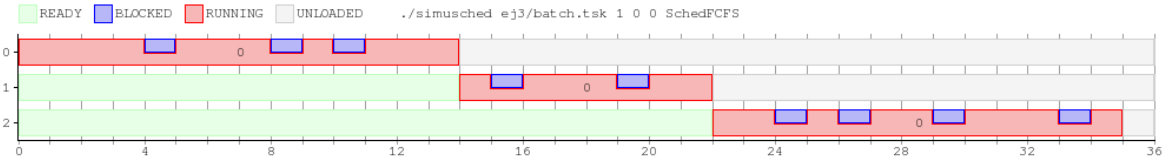

Figura 5: