Algoritmos y Estructuras de Datos III
Trabajo Práctico 1

Martín Arjovsky 683/12
Ezequiel Darío Gambaccini
Silvio Vileriño

Abril 2014

1 Ejercicio 2

A continuación se muestran los gráficos del lote 2 para 1, 2 y 3 núcleos.

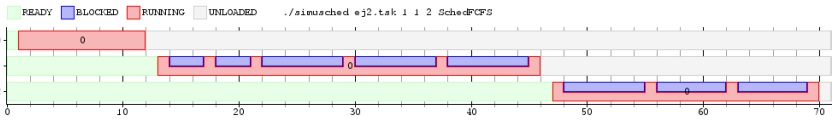

Figure 1: Diagrama de Gantt para el lote 2 en FCFS con un núcleo

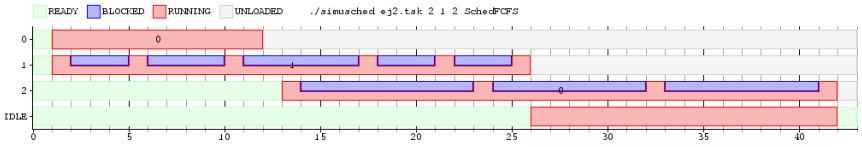

Figure 2: Diagrama de Gantt para el lote 2 en FCFS con dos núcleos

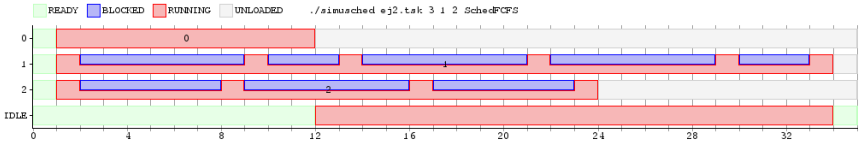

Figure 3: Diagrama de Gantt para el lote 2 en FCFS con tres núcleos

2 Ejercicio 4

A continuación se muestran los gráficos del lote 4 ejecutado en el scheduler Round-Robin.

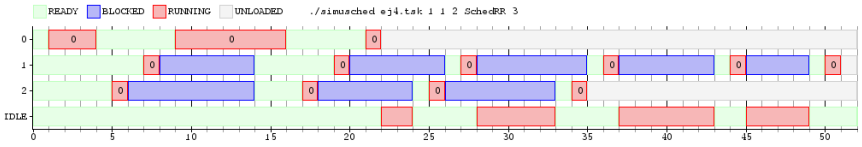

Figure 4: Diagrama de Gantt para el lote 4 en RR con un núcleo

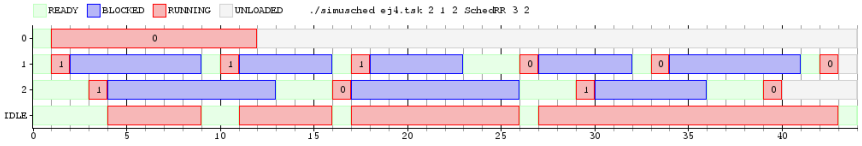

Figure 5: Diagrama de Gantt para el lote 4 en RR con dos núcleos

En ambos gráficos se observa el comportamiento esperado de un scheduler de tipo Round-Robin. En particular, en la figura 4 se ve como la tarea 0 corre en el único núcleo hasta que en el tiempo 4 se le acaba el quantum disponible y el scheduler busca otro proceso disponible en la cola global. También se muestra repetidamente como cuando un proceso realiza una llamada bloqueante, el CPU deja esa tarea para ir a ejecutar otra. En la figura 5 se observa el mismo comportamiento, pero además podemos observar la migración de procesos entre núcleos, por ejemplo a tiempo 16 en la tarea 2.

3 Ejercicio 7

Para estudiar la performance del Round-Robin variando los cuantos creamos el lote7 y realizamos varias simulaciones. Variamos los quantums entre 1 y 10 inclusive y evaluamos el throughput y INSERTAR SEGUNDA MÉTRICA con 2 y 4 núcleos. Como las tareas TaskConsola son pseudoaleatorias, para cada seleccioón de parámetros corrimos 10 simulaciones y graficamos el promedio con la variación estandard. En la figura 6 se muestran los resultados del throughput.

ASD

4 Ejercicio 8

A continuación se muestran los gráficos del lote 8 ejecutado sobre los schedulers Round-Robin con y sin migración entre procesos.

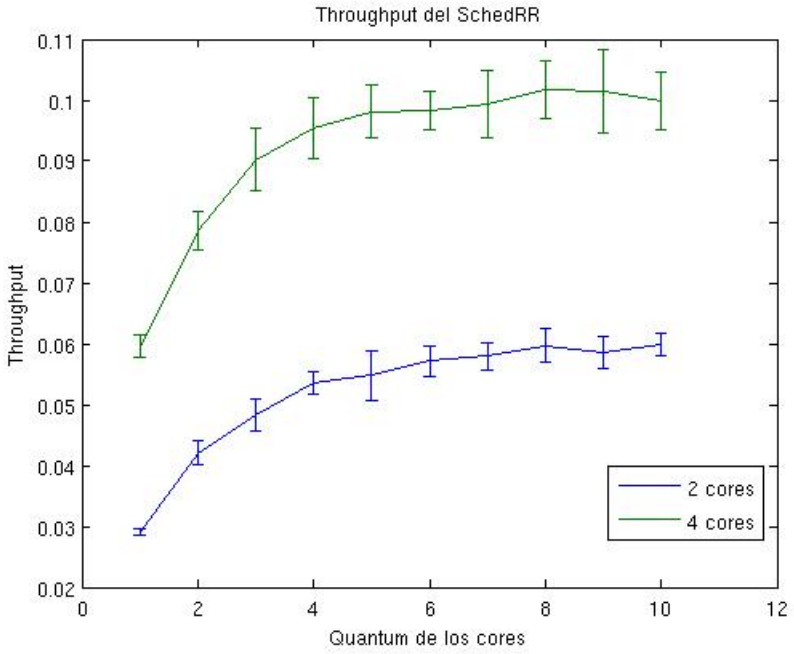

Figure 6: Throughput para SchedRR con dos y cuatro núcleos

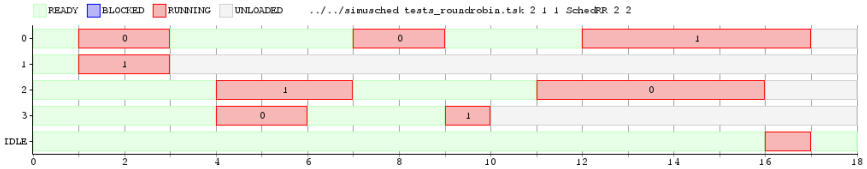

Figure 7: Diagrama de Gantt para el lote 8 en RR con dos núcleos

Figure 8: Diagrama de Gantt para el lote 8 en RR2 con dos núcleos

Se puede observar en el caso donde hay tareas con considerable diferencia de duracion, una mejor performance del scheduler RR1, dado que realiza una correccion permanente del balanceo de carga usando los recursos que tiene a su alcance en cada tick. Mientras que en el RR2, al quedar fija la afinidad al momento del dispatch, si ocurre un caso donde ambas tareas cortas quedan en un core y las tareas largas en otro core, al finalizar las tareas cortas, se desperdiciara un core durante toda la ejecucion de las tareas largas. Esta diferencia se ve claramente en los gráficos 7 y 8 si notamos que cuando se usa RR1 se termina de ejecutar a tiempo 18 y con RR2 a tiempo 27, lo que es una diferencia abismal. Cabe destacar que para que esto pase se necesita que haya una gran diferencia entre el uso de CPU de las tareas y que las largas se encuentren concentradas en un núcleo mientras las cortas en otro, lo que no suele ser un caso promedio.