Presentación del Taller de Scheduling

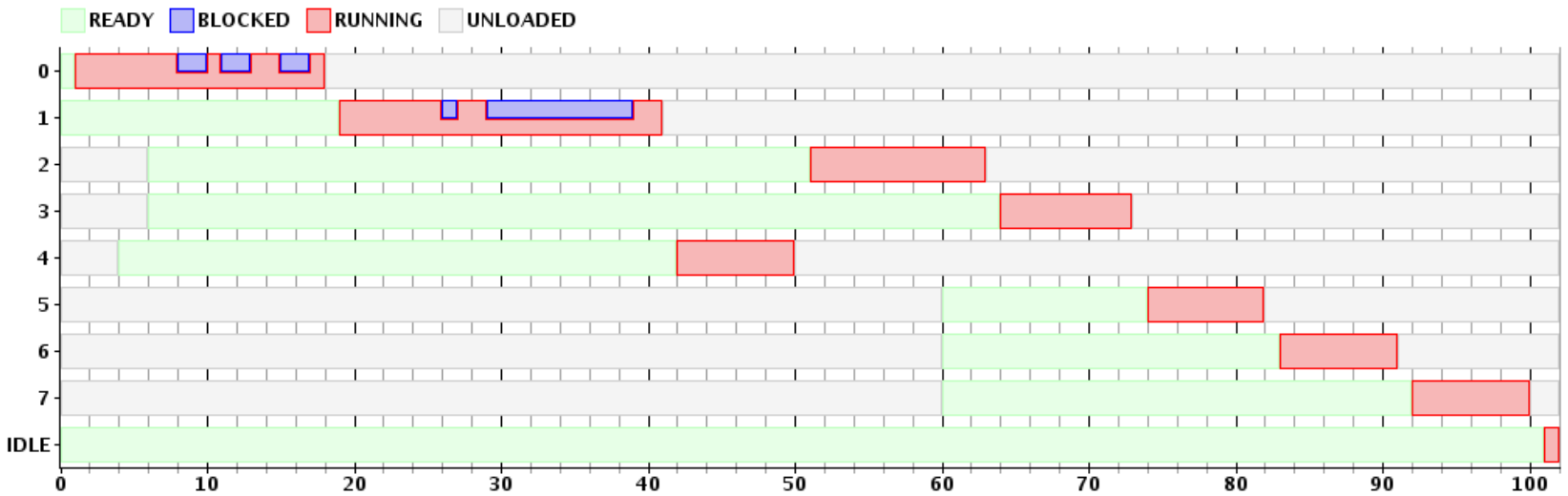
Matías Barbeito (Cani)

DC - FCEyN - UBA

Sistemas Operativos

Diagrama de GANTT

- Un diagrama de **GANTT** es una herramienta gráfica cuyo objetivo es mostrar el **estado** de cada uno de los **procesos** existentes en un sistema durante un período de tiempo determinado.



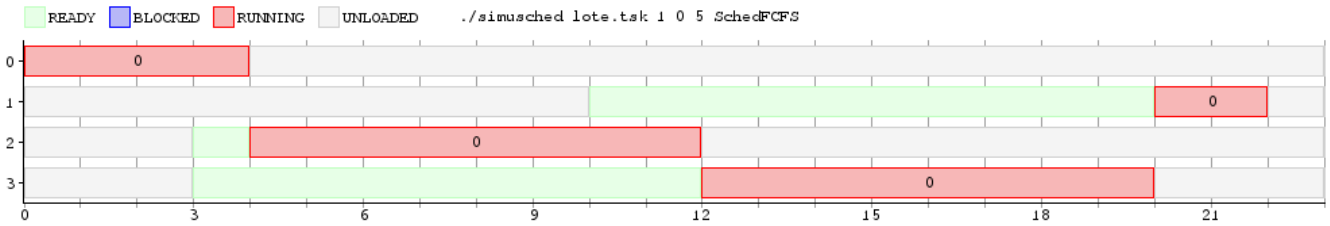
Entendiendo el simulador

Tareas

- Tipo (TaskCPU, TaskIO, ...)
- Parámetros
- Release Time

Lotes

```
TaskCPU 3
@10:
TaskCPU 1
@3:
*2 TaskCPU
7
```



Entendiendo el simulador

Definir un nuevo tipo de tarea

- Función dentro de *tasks.cpp*
- *uso_CPU*(t)
- *uso_IO*(t)
- *return*

```
void TaskIO(int pid, vector<int> params) {
    // Uso el CPU x ciclos.
    uso_CPU(pid, params[0]);

    // Uso IO y+1 ciclos.
    uso_IO(pid, params[1]);

    // Uso 1 ciclo
    return;
}
```

Entendiendo el simulador

```
./simusched <lote.tsk> <num_cores> <costo_cs> <costo_mi> <sched> [<params_sched>]
```

- <lote.tsk> es el archivo que especifica el lote de tareas a simular.
- <num cores> es la cantidad de núcleos de procesamiento.
- <costo cs> es el costo de cambiar de contexto.
- <costo mi> es el costo de cambiar un proceso de núcleo
- <sched> es el nombre de la clase de scheduler a utilizar

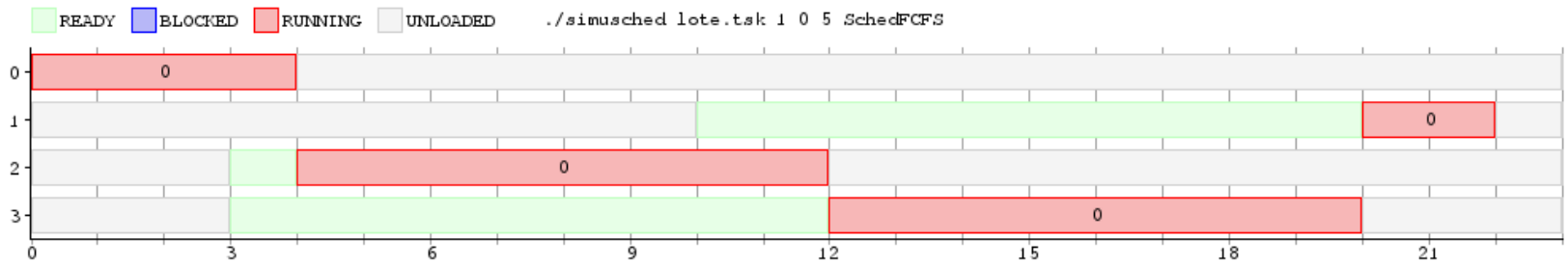
Entendiendo el simulador

```
./simusched <lote.tsk> <num_cores> <costo_cs> <costo_mi> <sched> [<params_sched>]
```

- LOAD 0 1. En el ciclo 0, la tarea 1 está ready
- CPU 33 1 0. En el ciclo 33, se está ejecutando la tarea 1 en el core 0
- BLOCK 44 1. En el ciclo 44, la tarea 1 está bloqueada
- UNBLOCK 65 1. En el ciclo 65, se desbloqueó la tarea 1
- EXIT 66 1 0. En el ciclo 66, la tarea 1 terminó en el core 0
- CPU 112 -1 0. En el ciclo 112, el core 0 está idle.
- CONTEXT CPU 0 32. Cambio de contexto en el core 0, en el ciclo 32

Graficación de simulaciones

```
./simusched lote.tsk 1 0 5 SchedFCFS | ./graphsched.py > imagen.png
```



Pueden tener que instalar las siguientes librerías

sudo apt-get install python-matplotlib libfreetype6-dev ttf-freefont
sudo pip install Pillow

Veamos el enunciado

Scheduler FCFS

- ¿Cómo era el scheduler **First Come, First Served**?

Scheduler FCFS

- ¿Cómo era el scheduler **First Come, First Served**?
- Va asignando el procesador según el orden de llegada de los procesos

Scheduler FCFS

- ¿Cómo era el scheduler **First Come, First Served**?
- Va asignando el procesador según el orden de llegada de los procesos
- NO tiene desalojo, mantiene asignado el procesador aunque se bloquee
- ¿Qué pasa si hay varios cores?

Scheduler FCFS

- ¿Cómo era el scheduler **First Come, First Served**?
- Va asignando el procesador según el orden de llegada de los procesos
- NO tiene desalojo, mantiene asignado el procesador aunque se bloquee
- ¿Qué pasa si hay varios cores?
- En su forma clásica tiene una cola general y asigna al core que se libera
- ¿Hay migración entre cores?

Scheduler FCFS

- ¿Cómo era el scheduler **First Come, First Served**?
- Va asignando el procesador según el orden de llegada de los procesos
- NO tiene desalojo, mantiene asignado el procesador aunque se bloquee
- ¿Qué pasa si hay varios cores?
- En su forma clásica tiene una cola general y asigna al core que se libera
- ¿Hay migración entre cores?
- No, no tiene sentido porque no hay desalojo

Scheduler Round-Robin

- ¿Cómo era el scheduler **Round-Robin**?

Scheduler Round-Robin

- ¿Cómo era el scheduler **Round-Robin**?
- Asigna el procesador por un lapso de tiempo (quantum) y luego desaloja

Scheduler Round-Robin

- ¿Cómo era el scheduler **Round-Robin**?
- Asigna el procesador por un lapso de tiempo (quantum) y luego desaloja
- También desaloja a un proceso si se bloquea
- ¿Qué pasa si hay varios cores? ¿Hay migración entre cores?

Scheduler Round-Robin

- ¿Cómo era el scheduler **Round-Robin**?
- Asigna el procesador por un lapso de tiempo (quantum) y luego desaloja
- También desaloja a un proceso si se bloquea
- ¿Qué pasa si hay varios cores? ¿Hay migración entre cores?
- Hay muchas posibilidades, en este taller tomamos dos:
  - Una cola global con migración entre núcleos
  - Cola por núcleo con asignación fija

Métricas

Existen diversas métricas para evaluar el comportamiento de los schedulers. Ninguna es absoluta, sino que nos dan una idea de cierto aspecto del scheduler que estamos evaluando. Incluso un mismo scheduler puede empeorar o mejorar mucho su rendimiento para una misma métrica dependiendo del lote y del tipo de tareas. Algunas de las métricas que utilizamos son:

- Latencia: tiempo hasta que el proceso se ejecuta por primera vez
- Throughput: cantidad de procesos que terminan por unidad de tiempo
- Waiting time: tiempo que el proceso está en estado **ready**
- Turnarround: tiempo total que le toma a un proceso terminar de ejecutarse

Notar que algunas métricas son por proceso. En esos casos suele tomarse el promedio entre todos los procesos del lote para evaluar al scheduler en ese caso.

Manos a la obra

- Trabajen en grupos de a 3 (tres).
- Tengan la teórica como referencia.
- El taller está en la página para descargar.