MOD22-MLFQ

TRABALHO PRÁTICO

O programa mlfq.py simula a excução do MLFQ. Veja o ficheiro *README-mlfq-pt.txt* para mais detalhes.

1- Corra alguns problemas gerados aleatoriamente apenas com duas tarefas e duas filas. Antes de correr com o *trace (-c)* preveja o resultado. Para tornar mais fácil esta questão, limite a dimensão de cada tarefa e elimine as E/S.
Em particular: Corra dois trabalhos em duas fila com tempo máximo de execução de 10 e com o quantum 2.
(a) Em que fila terminam os dois trabalhos a execução?
(b) Que parâmetros devemos alterar para que um dos trabalhos termine a execução na fila com maior prioridade (sem recorrer ao *boost*)?

2- Pretende-se simular alguns dos exemplos apresentados nos slides das sessões teóricas. Em particular:
(a) Uma tarefa pesada – sem I/O, confrontada com tarefas que são muito interativas. Experimente diferentes valores de *boost* e considere o tempo de arranque mais tardio para as tarefas interativas **(slide 13)**
(b) Uma tarefa 0 que utiliza o facto de não se contabilizar o tempo total (com a flag -S ligada) versus uma tarefa 1 pesada. A tarefa 1 inicia mais cedo e vai gastando o seu "time slice" saltando sucessivamente para filas com menor prioridade. A tarefa 0, esperta, renuncia ao CPU num curto intervalo de tempo (I/O fingida) e volta a ter novamente a prioridade máxima **(side 16)**

3- Como configuraria o escalonador de forma a que este se comportasse como um escalonador *round-robin*, isto é, um escalonador que executa à vez diferentes trabalhos durante um determinada fatia de tempo ou *quantum*.

4- Desenhe uma carga (*workload*) com dois trabalhos, atribuindo parâmetros ao escalonador de forma que uma tarefa aproveite as regras **4a** e **4b** (ativas com a flag -S) para jogar com o escalonador de forma a garantir para si 99% do CPU.

5- Dado o comprimento do *quantum* igual a 10 ms na fila com maior prioridade (prioridade mais alta) de quanto em quanto tempo dever-se-ia lançar o *boost* (utilizando a flag -B) de forma a garantir que uma tarefa longa (que poderá entrar em inanição - *starvation*) obtenha pelo menos 5% do CPU?

6- Uma questão que surge na tarefa de escalonamento é qual os lados da fila à qual se adiciona uma tarefa que terminou uma E/S. A flag -I modifica este comportamento no simulador. Procura perceber o efeito desta *flag* no resultado do escalonamento para diferentes cargas.