1 Lezione del 08-10-25

Continuiamo la discussione degli algoritmi di scheduling.

1.0.1 Algoritmo RR

L'algoritmo **RR** (*Round Robin*) è preemptive e non prioritario, e si basa su un meccanismo molto semplice: ad ogni processo viene dato un quanto temporale prefissato, e via preemption si passa al processo successivo quando tale quanto viene esaurito.

Questo lo rende molto efficiente: l'overhead O_v è minimo, quasi al pari di FCFS (leggermente più alto in quanto i cambi di contesto sono più frequenti, uno ogni quanto temporale).

Dal punto di vista implementativo, organizzaremo i quanti temporali sfruttando una componente hardware detta **timer**: questo è considerato a tutti gli effetti una periferiche, ed invia (dopo un'opportuna configurazione) interruzioni esterne periodiche. Ogni volta che il processore riceve tale interruzione, mette in esecuzione lo scheduler, che provvede a cambiare il contesto al prossimo processo. Chiaramente, lo scheduler può comunque essere messo in esecuzione da eventi comuni come la terminazione di processi.

Per realizzare l'esecuzione ciclico si usa una semplice coda pronti dove lo scheduler estreae sempre dalla testa e inserisce sempre in fondo alla coda.

Il RR rende molto semplice stimare il tempo di attesa: se ci sono N processi in esecuzione, $N-1$ saranno in coda pronti in qualsiasi momento, quindi un dato processo aspetterà:

$$T_a = (N-1)q$$

dove q è il quanto di tempo.

Chiaramente il T_a diventa troppo grande se ci sono troppi processi.