S3/L1

L’esercizio di oggi verte sui meccanismi di pianificazione dell’utilizzo della CPU (o processore).

In ottica di ottimizzazione della gestione dei processi, abbiamo visto come lo scheduler si sia evoluto nel tempo per passare da approccio mono-tasking ad approcci multi-tasking.

Traccia: Si considerino 4 processi, che chiameremo P1,P2,P3,P4, con i tempi di esecuzione e di attesa input/output dati in tabella. I processi arrivano alle CPU in ordine P1,P2,P3,P4. Individuare il modo più efficace per la gestione e l’esecuzione dei processi, tra i metodi visti nella lezione teorica. Abbozzare un diagramma che abbia sulle ascisse il tempo passato da un instante «0» e sulle ordinate il nome del Processo.

|  |  |  |  |
| --- | --- | --- | --- |
| Processo | Tempo di esecuzione | Tempo di attesa | Tempo di esecuzione dopo attesa |
| P1 | 3 secondi | 2 secondi | 1 secondo |
| P2 | 2 secondi | 1 secondo |  |
| P3 | 1 secondo |  |  |
| P4 | 4 secondi | 1 secondo |  |

I sistemi operativi che gestiscono l’esecuzione di un solo programma per volta sono detti mono-tasking in quanto non è possibile sospendere l’esecuzione di un programma per assegnare la CPU ad un altro programma. I sistemi operativi mono-tasking risultano piuttosto inefficienti, a causa dei periodi di inutilizzo della CPU, la quale rimane in attesa di eventi esterni.

Tempo di utilizzo CPU

Tempo attesa eventi esterni

MONO-TASKING

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P4 |  |  |  |  |  |  |  |  |  |  |  | | | |  |
| P3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P2 |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| P1 |  | | |  | |  |  |  |  |  |  |  |  |  |  |
| P/T | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

P = processi T = tempo in secondi

I sistemi operativi che permettono l’esecuzione contemporanea di più programmi sono detti, invece multi-tasking. In questi, la pianificazione con prelazione fa’ in modo che quando un processo è in attesa di eventi esterni, la CPU possa essere impiegata per altro processo, piuttosto che attendere inattiva. In questo caso, quando il processo 1 passa dallo stato di esecuzione a quello di attesa, la CPU potrà essere impiegata per eseguire il processo 2, e così via.

MULTI-TASKING

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P4 |  |  |  |  |  |  |  | | | |  |  |  |  |  |
| P3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P2 |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |
| P1 |  | | |  | |  |  |  |  |  |  |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

Infine, un’evoluzione dei sistemi multi-tasking sono i sistemi time-sharing. In un sistema time-sharing ogni processo viene eseguito in maniera ciclica per piccole porzioni di tempo che prendono il nome di “quanti” dati dal server. Grazie l’elevata velocità della CPU, l’esecuzione dei processi risulterà essere quasi in parallelo gli uni con gli altri.

Il processo 1 viene interrotto per passare ad eseguire un altro processo per un “quanto” e cosi via.

TIME-SHARING

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| P1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

In conclusione possiamo dire che, ad oggi il modo più efficace per la gestione e l’esecuzione dei processi, è attraverso il sistema time-sharing.