Consegna S3/L1 - Scheduling CPU

L'esercizio di oggi verte sui meccanismi di pianificazione dell'utilizzo della CPU.

Traccia dell'esercizio

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. 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 secondi	-	-
P4	4 secondi	1 secondo	-

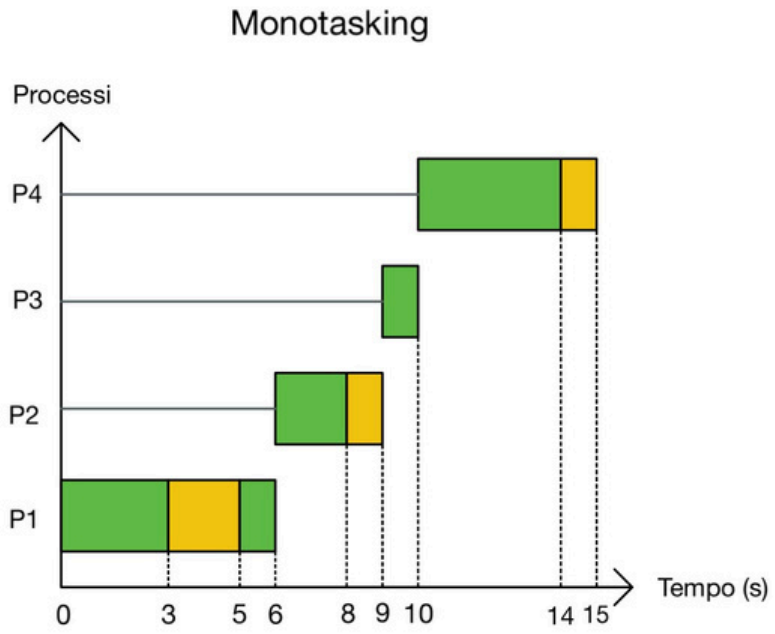
Svolgimento

Per classificare le velocità di esecuzione dei processi ho preso come riferimento tre sistemi: ·Sistemi **monotasking** ·Sistemi **multitasking** ·Sistemi **time-sharing**.

Ho creato un grafico per ogni tipologia di sistema preso in esami in modo tale da mostrare le differenze dei tre diversi sistemi.

Caso sistema monotasking

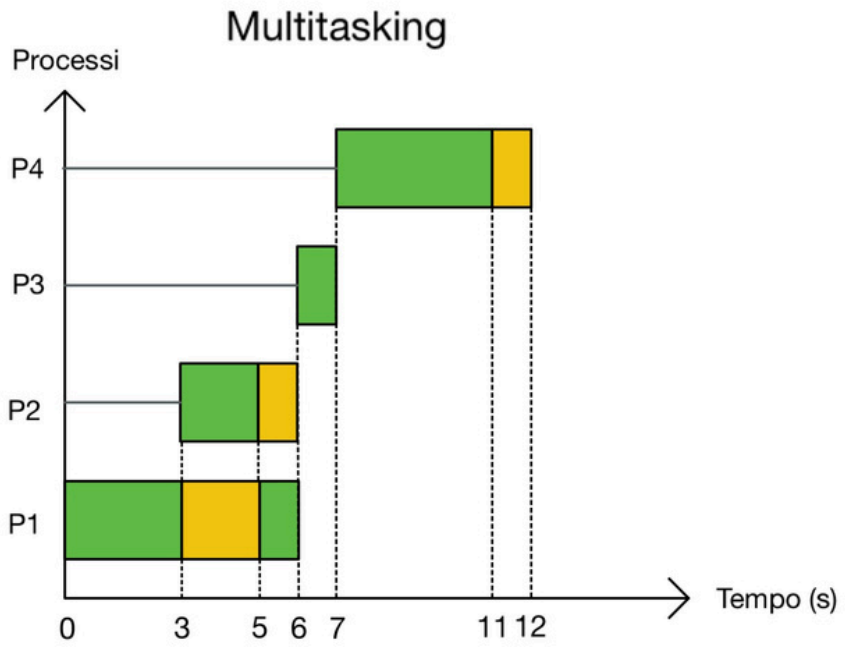
Il primo caso che ho analizzato è quello di un sistema monotasking. In questa tipologia di scheduling dei processi, la CPU esegue ogni programma fino al suo completamento, senza poterlo mettere in pausa. Nel caso pratico di studio, il processore terminerà l'esecuzione dei 4 processi dopo 15 secondi.



Caso sistemi multitasking

Nei sistemi multitasking, il processore può eseguire più processi contemporaneamente. La CPU infatti può interrompere il programma in esecuzione per eseguire un processo diverso. Questo permette al sistema di impiegare la capacità di calcolo per altri compiti, anziché restare inattiva.

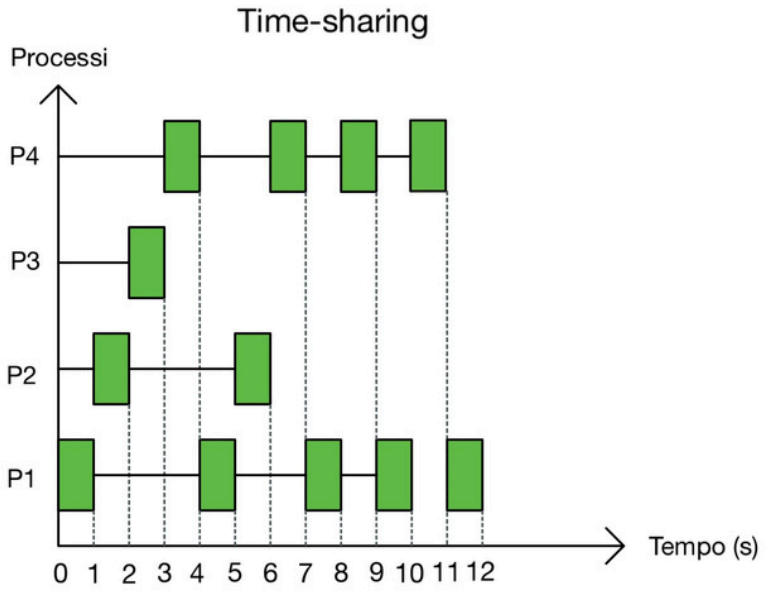
Nel caso pratico, come mostra il diagramma, quando un processo è in stato di attesa, il processore esegue una parte di un altro processo, per poi tornare a lavorare sul processo "iniziale". Questa implementazione permette di risparmiare tempo e ottimizza i processi. Il tempo totale impiegato dal sistema multitasking di questo esempio è di 12 secondi.
Nota: Avrei potuto realizzare una soluzione che permette di completare tutti i processi in 11 secondi, ad esempio anticipando l'esecuzione di P3 e posticipando l'esecuzione dell'ultima parte del processo P1, nel momento di attesa di P4 ma ho preferito creare un'implementazione in cui i processi con indice più basso hanno maggiore "priorità" e quindi richiedono di essere terminati nel più breve tempo possibile.



Caso sistemi time-sharing

Nei sistemi time-sharing ogni processo è in esecuzione per un determinato periodo di tempo (detto **quanto**). Al termine del tempo prestabilito, il processo viene interrotto per passare al processo successivo per un altro quanto. Ogni processo dunque viene eseguito in maniera ciclica per piccole porzioni di tempo, fino al suo completamento.

Nel caso analizzato, ho preso come quanto 1 secondo. Ho scelto di realizzare questa implementazione, prendendo come tempo di riferimento il tempo minimo per completare un processo (P3). Ho considerato inoltre che il periodo di attesa di 2 secondi di P1 sia da considerarsi successivo rispetto ai primi 3 secondi di esecuzione del processo.
Anche questa ottimizzazione riesce a terminare tutti i processi in 12 secondi.



Conclusione

Risulta evidente che i sistemi multitasking e time-sharing siano un’evoluzione dei sistemi monotasking e permettano di ottimizzare i tempi di esecuzione dei processi da parte della CPU, tuttavia non è possibile dire quale dei due sistemi sia migliore poiché dipende da molteplici fattori come: il numero di processi da completare, la priorità dei processi, la scelta dell’implementazione ecc.