First Come First Served

martedì 11 aprile 2023 12:00

Scheduling FCFS (First Come First Served)

Storicamente il problema dello scheduling di CPU ha riguardato sistemi a processi ed e' nato nell'ambito di macchine uniprocessore ...

i processi nello stato Ready vengono mandati in esecuzione secondo l'ordine di inserimento nella “Ready Queue”

non vi è prelazione, quindi ogni processo rimane in esecuzione fino al suo completamento, oppure fino a che esso non rilascia la CPU spontaneamente

L'algoritmo FCFS (First come, first served) è un tipo di algoritmo FIFO: esegue i processi nello stesso ordine in cui essi vengono sottomessi al sistema.
Il primo processo ad essere eseguito è esattamente quello che per primo richiede l'uso della CPU.
Quelli successivi vengono serviti non appena questo ha terminato la propria esecuzione, e così avviene successivamente per tutti gli altri posti in coda.
Questo tipo di algoritmo è molto semplice da implementare ma solitamente è anche poco efficiente, almeno considerando il tempo medio d'attesa.

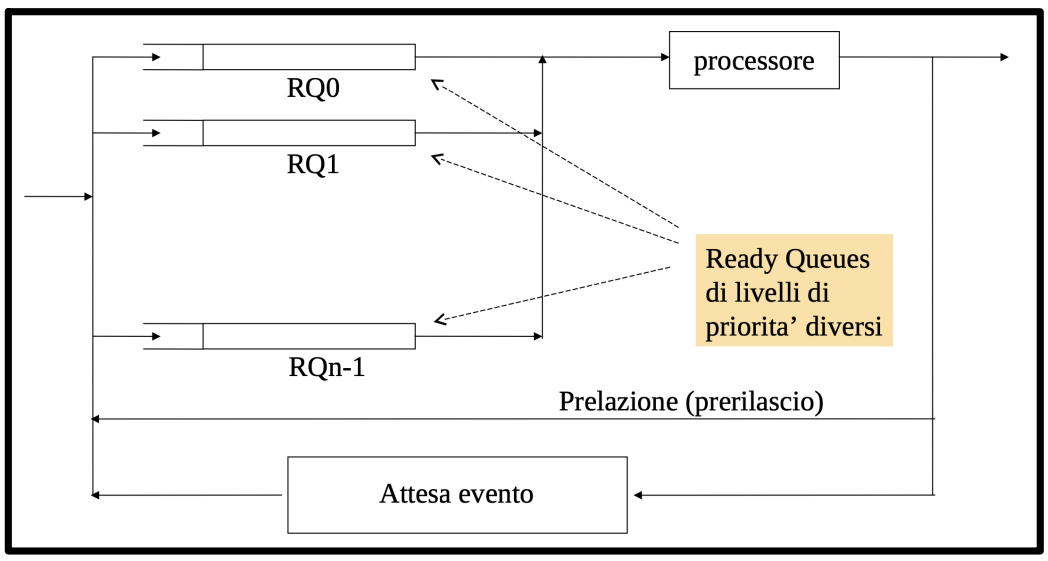
Infatti, prendiamo ad esempio la sottomissione nell'ordine dei seguenti processi con la seguente durata espressa in millisecondi:

- **p1: 10**
- **p2: 4**
- **p3: 2**

Verranno eseguiti nello stesso ordine: p1 → p2 → p3

Il processo p1 non attende nulla, perché entra immediatamente in esecuzione. Il processo p2 attende 10 millisecondi, e p3 14. Il tempo medio d'attesa quindi è (0+10+14)/3=8 millisecondi.

È il PCB (Process Control Block) che viene inserito nella coda delle priorità, come nella seguente immagine:



In questo caso abbiamo solo una coda, in cui si va ad inserire le applicazioni che sono "Ready". Quindi quando un'applicazione entra nello stato ready il suo PCB viene inserito nella coda. Ogni processo rimane in esecuzione fino al suo completamento.
Non vi è prelazione

E da questa coda andiamo man mano a pescare il primo processo con una politica FIFO.

Svantaggi

- non minimizza il tempo di attesa, e di conseguenza neanche il tempo di turnaround
- inadeguato per la gestione di processi interattivi
- può causare sottoutilizzo dei dispositivi di I/O a causa del fatto che i processi interattivi non necessariamente vengono favoriti

Se noi consideriamo P1 e P2 e nella Ready Queue c’è prima P1 e dopo P2, quest’ultimo potrebbe essere un processo interattivo e quindi usare

la CPU per pochissimo (1 min) (mentre P1 la vuole utilizzare per 1 ora) per andare a rilasciare la CPU spontaneamente chiamando il Sistema Operativo per interagire con qualcos'altro, PER FAR Sì CHE IL SOFTWARE DI SISTEMA ATTIVI UN DISPOSITIVO.
Questa cosa avverrà solo tra un'ora.
In questo scenario, questo scheduler è inefficiente,

È importante la distinzione tra scheduling con diritto di prelazione (scheduling *preemptive*) e scheduling senza diritto di prelazione (scheduling *non-preemptive* o *cooperative*).

Nel primo caso lo scheduler può sottrarre il possesso del processore al processo anche quando questo potrebbe proseguire nella propria esecuzione.

Nel secondo caso, invece, lo scheduler deve attendere che il processo termini o che cambi il suo stato da quello di esecuzione a quello di attesa.

È uno scheduler scomparso dai sistemi di CPU moderni.