La schedulazione dei processi

(schedulatore a breve termine)

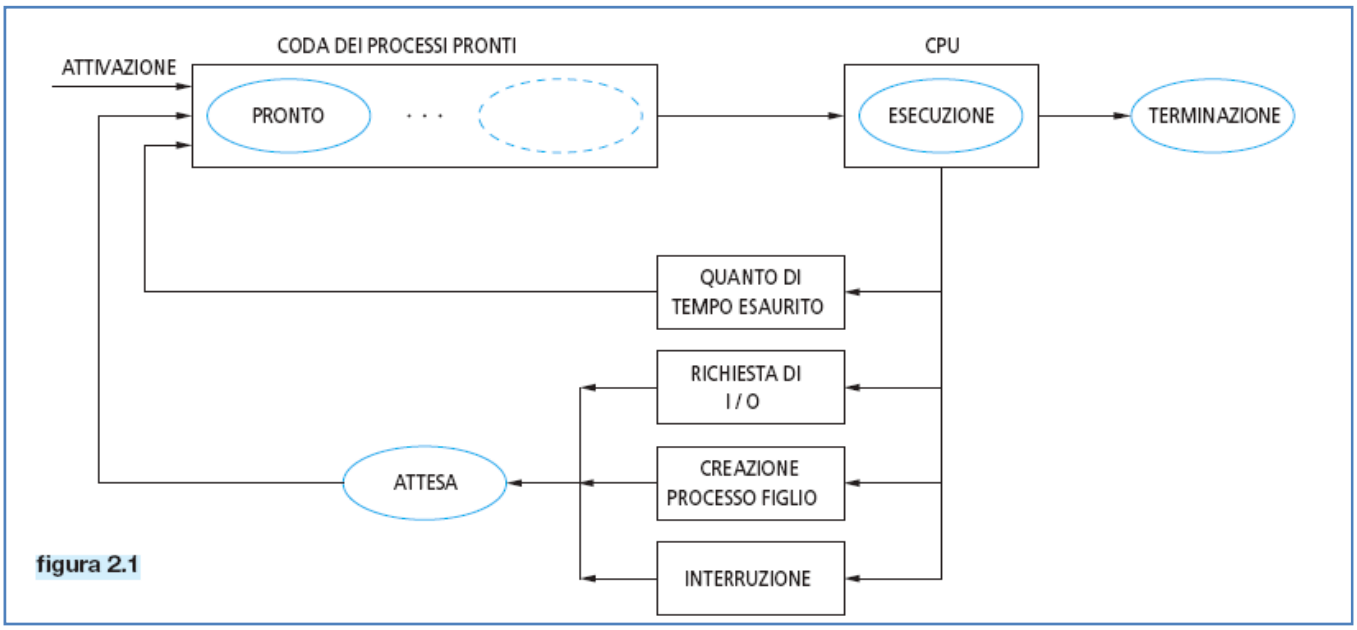


figura 2.1

Come già detto in precedenza l'algoritmo di rotazione tra i **processi in esecuzione** deve essere il più equo ed efficiente possibile. Per questo motivo è opportuno avere sia processi che usano molto la CPU, sia processi che usano molto le periferiche (I/O). In questo modo si evitano tempi di risposta inaccettabili per i programmi interattivi, in un caso, e inattività del processore nell'altro.

In particolare lo schedulatore dei processi usa precisi criteri per operare la scelta di quale processo far avanzare dallo stato di "pronto" (ready) allo stato di "esecuzione" (run).

Questi criteri si dividono in due grandi categorie:

- **Politiche di scheduling non preemptive (senza prerilascio) :** una volta che la CPU è stata assegnata ad un processo, essa non può essergli tolta fino a quando il processo non è terminato oppure se ha eseguito un'operazione di I/O. Questo tipo di politica è adatta per sistemi real time, dove un'interruzione del processo in esecuzione potrebbe avere conseguenze catastrofiche.

  First Come First Served (FCFS): il processore viene assegnato ai processi in stato di pronto a seconda dell'ordine di arrivo.

  Difetti: tempo medio di attesa abbastanza lungo e vengono penalizzati i processi brevi.

  Shortest Job First (SJF): il processore viene assegnato al processo, tra tutti quelli in stato di pronto, che prevede l'intervallo di tempo più breve di utilizzo della CPU prima di un I/O.

  Difetti: spesso non si hanno sufficienti informazioni per stabilire quale processo richiederà per primo un'operazione di I/O, perché normalmente questo dato è disponibile solo a run time.

  Highest Response Ratio Next (HRRN): ideato per correggere in parte i problemi di SJF e in particolare l'eccessiva tendenza a sfavorire i processi più lunghi e l'esagerata preferenza per quelli più brevi; il processore viene assegnato ad un processo secondo una priorità dinamica che

dipende dal tempo di esecuzione stimato e dal tempo in cui il processo è rimasto in attesa in stato di pronto.

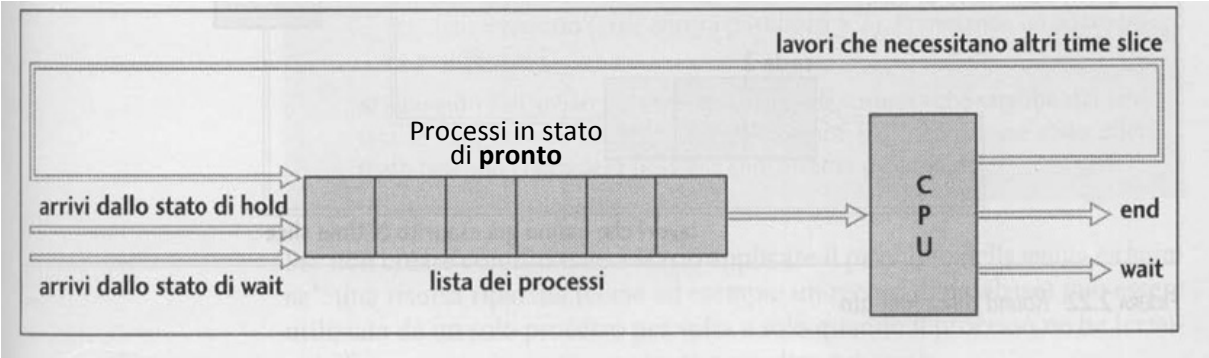
$$\textbf{Priorità} = \frac{\text{T. di attesa + T. stimato di esecuzione}}{\text{T. stimato di esecuzione}} = 1 + \frac{\text{T. di attesa}}{\text{T. stimato di esecuzione}}$$

In questo caso a parità di tempo di attesa, vengono privilegiati i processi più corti, ma se il tempo di attesa aumenta vengono privilegiati anche i processi più lunghi.
Questo evita allo schedulatore di posporre un processo indefinitamente (starvation).

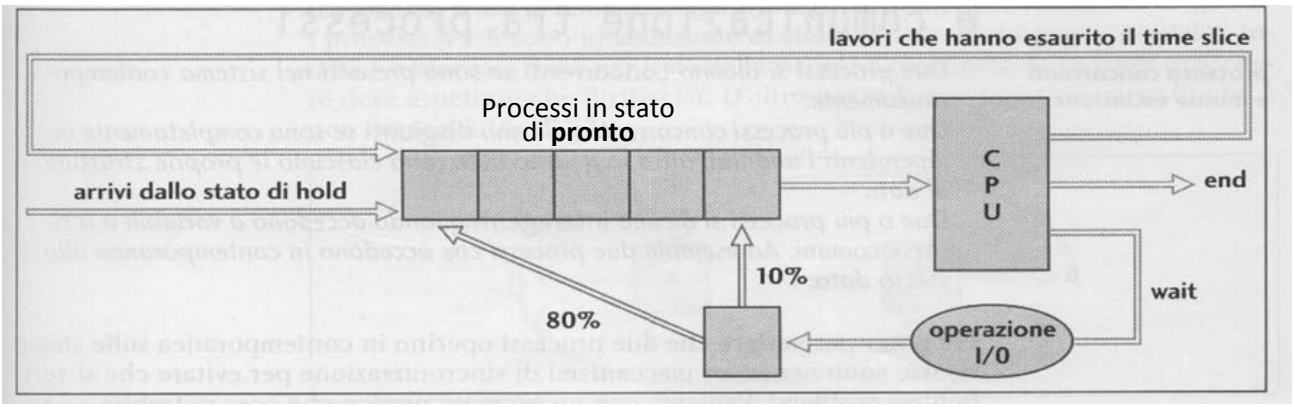
Il fenomeno della **starvation** si verifica quando uno o più processi, magari a causa di una priorità bassa, rimangono nella coda dei processi in stato di "ready" per **un tempo indefinito** in quanto sopraggiungono continuamente processi in stato di "ready" con priorità più alta.

- **Politiche di scheduling preemptive (con prerilascio) :** viene usato nei sistemi operativi multitasking, con il significato che il S.O. che può interrompere un processo in esecuzione contro la volontà di questo in favore di un altro processo più prioritario oppure se il processo ha utilizzato la CPU per il massimo tempo consentito (fine del quanto di tempo).

round robin: il processore viene assegnato a turno, per un intervallo di tempo stabilito (**time slice o quanto di tempo**), ai processi nell'ordine in cui questi ne hanno fatto richiesta.
La **coda** è gestita con il metodo FIFO e quindi tutti gli inserimenti provenienti dalla coda di hold, dalla coda di wait o quelli causati dall'esaurimento di un time slice precedente avvengono al fondo della coda



round robin a percentuale di tempo: con la tecnica precedente i processi che richiedono molte operazioni di I/O sono penalizzati rispetto a quelli che ne richiedono di meno perché il rientro dallo stato di wait significa l'inserimento del processo al fondo della coda.
La tecnica a percentuale di tempo è una variante del round robin: il processo non verrà inserito necessariamente al fondo, ma la posizione di rientro nella coda dipende dalla percentuale di tempo di CPU utilizzata. Questo significa che se un processo lascia il processore perché richiede un I/O, e quindi non ha ancora esaurito il proprio time slice, il suo rientro nella coda non sarà più al fondo ma tanto più avanti quanto minore è la percentuale di tempo di CPU utilizzata prima dell'interruzione.



round robin con priorità statica: a ogni processo viene assegnata una priorità in fase di generazione, ad esempio in base al tempo presunto di esecuzione. Lo schedulatore dei processi terrà conto di questo valore per inserire il processo nella coda.

round robin con priorità dinamica e code a diversi livelli di priorità: il S.O. modifica la priorità in base al tempo di utilizzo della CPU; ai processi che rilasciano la CPU prima dello scadere del quanto temporale viene aumentato il valore della priorità.

I processi vengono organizzati in attesa su diverse code, corrispondenti alle diverse priorità assegnate a ciascuno, e al momento di scegliere il processo da mandare in esecuzione vengono favoriti quelli a priorità più elevata. Quando le code con priorità superiore sono vuote vengono scelti i processi delle code con priorità più bassa. L'esecuzione di un processo a priorità inferiore inoltre può essere interrotta per provvedere all'esecuzione di processi a priorità più alta che nel frattempo sono passati nello stato di "pronto".

Per evitare che i processi con priorità più bassa non ricevano mai il processore o siano interrotti frequentemente senza riuscire ad arrivare dallo stato di "esecuzione" allo stato di "terminazione", il S.O. aumenta la loro priorità tenendo conto del tempo di permanenza in stato di "pronto".