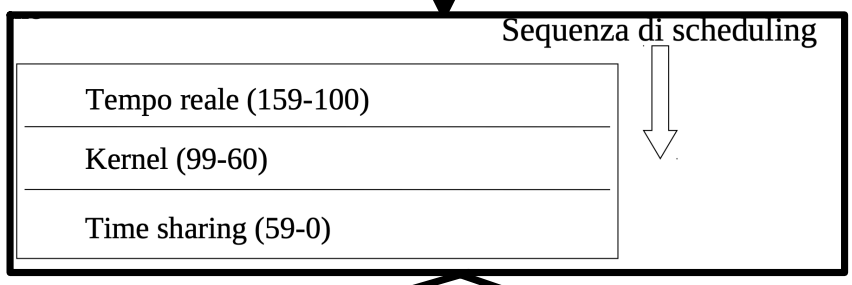
Scheduling UNIX SVR4

giovedì 13 aprile 2023 14:12

Scheduling UNIX SVR4

Avevamo uno schema di gestione delle priorità dei thread attivi con 160 livelli di priorità e 3 classi di priorità differenti: Tempo reale, Kernel e Time-Sharing.



Avevamo l'idea di mantenere più stabili i thread all'interno dell'organizzazione delle priorità che avevamo assegnato, in modo tale da poter organizzare in maniera più coerente l'assegnazione della CPU alle varie attività che dovessero essere eseguite.

Una caratteristica fondamentale è il kernel preemptabile: abbiamo un thread che esegue, entra in modo kernel, e mentre siamo in modo kernel la CPU viene tolta a questo kernel. Quindi i thread che sono in esercizio possono perdere il controllo della CPU non soltanto quando sono USER o quando hanno chiamato un servizio bloccante, ma anche mentre eseguono a livello kernel un servizio non bloccante.

Il software del kernel è organizzato all'interno in modo da avere un Safe Places (SP) all'interno del blocco di codice dove è avvenuta a livello kernel un'interruzione, che possa permettere al kernel stesso, lungo quel thread, di cedere la CPU a qualche altro.
Questo è un modo per permetterci di essere in grado di far sì che se c'è un thread che è entrato ad eseguire in modo kernel ma questo ha una priorità bassa, nel momento in cui c'è la necessità di lasciare la CPU a qualcosa di molto più alto, di poter lasciare la CPU prima che il thread esca dalla modalità kernel, quindi prima che il thread completi le sue attività.

- **bitmap** per determinare i livelli non vuoti
- quanto di tempo variabile in funzione della classe e, in alcune classi, del livello