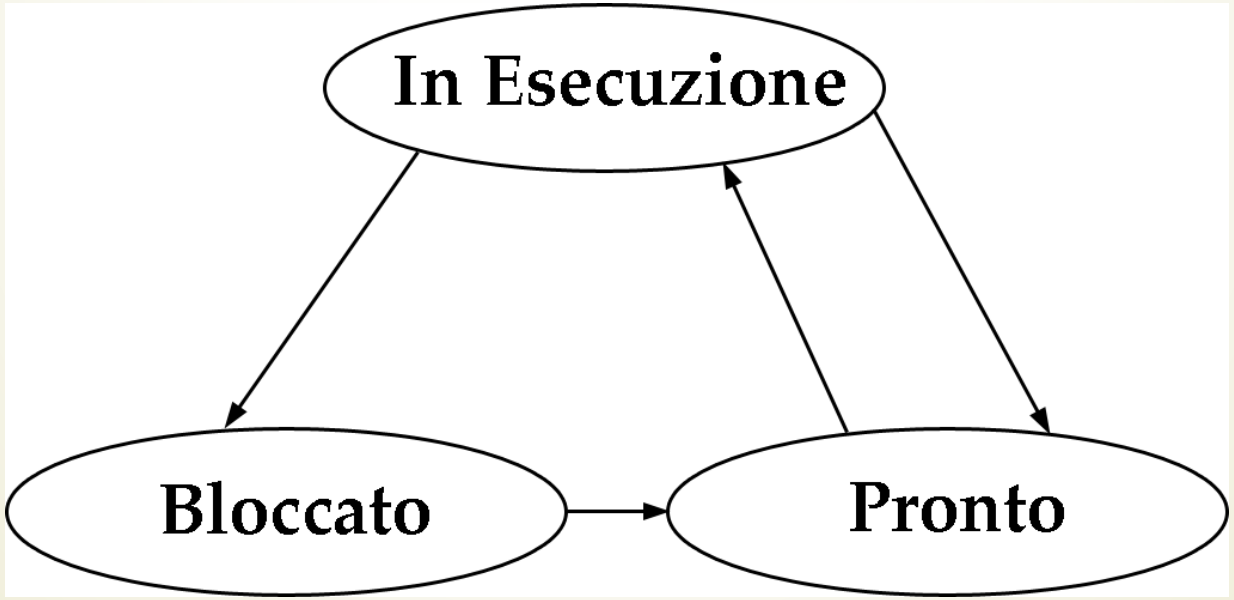
ESERCITAZIONE 6

Processi e thread

Processi e Thread

Calcolatori e processi (1)

1) In figura è mostrato lo stato di tre processi. In teoria, con 3 stati potrebbero verificarsi 6 transizioni, 2 per uscire da ogni stato. Tuttavia, sono mostrate solo 4 transizioni. Esistono circostanze in cui possono verificarsi una o entrambe le transizioni mancanti?



Calcolatori e processi (2)

2) Confrontare la lettura di un file usando un file server a thread singolo e uno multi-thread. Occorrono 15 ms per ottenere una richiesta di lavoro, smistarla ed elaborala, nel caso in cui essa faccia riferimento a dati presenti in un blocco di cache. Nel caso in cui sia necessario effettuare un'operazione su disco (che avviene in un terzo dei casi) sono richiesti ulteriori 75 ms, durante i quali il processo è sospeso. Quante richieste può gestire il server se è a thread singolo? E se è multi-thread?

Calcolatori e processi (2) Idea (1)

- Thread singolo: un unico thread può essere svolto contemporaneamente. Nel caso di chiamata bloccante, deve essere attesa la sua terminazione prima di riprendere l'esecuzione.
- Thread multiplo: più thread svolgibili con l'illusione del parallelismo anche in situazioni a processore singolo. Quando un thread invoca una chiamata bloccante, viene schedulato il thread successivo.

Calcolatori e processi (3)

3) In un computer con 1 GB di memoria, il sistema operativo occupa 512 MB e i processi occupano mediamente 64 MB, se l'attesa media dell'I/O è del 60%, qual è l'utilizzo della CPU? Aggiungendo 256 MB di RAM, quale sarà il nuovo utilizzo della CPU?

Calcolatori e processi (3) Idea (1)

- Il tempo di attesa del 60% è riferito all'attesa di completamento di operazioni di I/O.
- Se abbiamo *n* processi in memoria e il tempo di attesa medio è *p*, allora la probabilità che tutti i processi siano in attesa (e quindi la CPU sia intattiva) è p^n
- L'utilizzo della CPU è, dunque, 1- p^n
- Il numero *n* di processi è ottenibile dividendo la dimensione di RAM libera per lo spazio occupato naturalmente da un solo processo.

Calcolatori e processi (4)

4) In un sistema a thread a livello utente, c'è uno stack per thread oppure ce ne è una per processo? E in un sistema a thread a livello utente? Spiegare.

Calcolatori e processi (4) Idea (1)

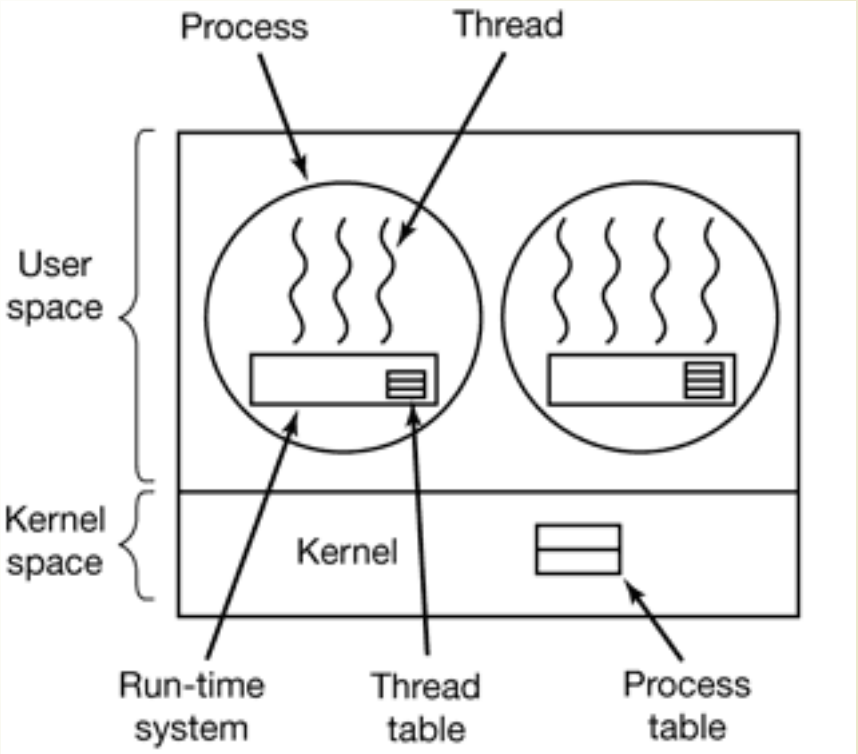
Thread

- unità di base di utilizzo della CPU
- Contiene:
  - Program counter
  - Insieme registri
  - Spazio stack
- Condivide:
  - Spazio indirizzamento (no protezione)
  - Dati globali
  - File aperti

Calcolatori e processi (4) Idea (2)

Thread livello utente

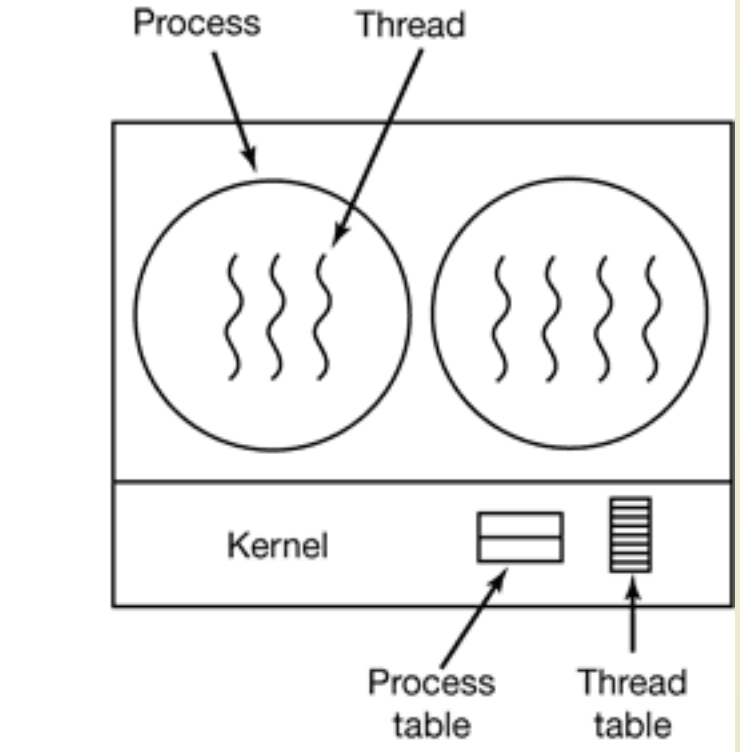
- Sistema ha un solo processo
- Implementabili in sistemi che non supportano thread
- Ogni processo ha tabella thread
- Cambio contesto thread più veloce rispetto processi
- Thread hanno algoritmo schedulazione interno
- Problema: chiamate bloccanti
- Thread possono switchare solo se rilascio CPU volontario



Calcolatori e processi (4) Idea (3)

Thread livello kernel

- Sistema conosce e gestisce i thread
- Tabella thread oltre che processi
- Chiamate bloccanti sono chiamate di sistema
- Costo elevato creazione superato tramite "riciclo" thread terminati.



Calcolatori e processi (5)

4) Supponendo di utilizzare l'algoritmo round – robin per lo scheduling in un sistema interattivo, se la coda è fatta dai processi A-B-C-D-E, nel caso in cui il quanto sia di 100 ms e il un tempo di cambio di contesto di 1 ms, quanto tempo sarà necessario prima che E venga eseguito? Qual è il rapporto tra cambio di contesto e tempo di esecuzione?

Come cambiano I tempi analizzati nel caso in cui il quanto sia, invece, di 4 ms?

Quale delle due soluzioni sembra più favorevole?

Calcolatori e processi (5)
Idea (1)

- Round- robin: a ogni processo viene assegnato un quanto di tempo in cui può andare in esecuzione, i processi vengono assegnati alla CPU tramite una coda circolare.
- Se ha una richiesta bloccante o è terminato prima dello scadere del quanto, la CPU viene immediatamente assegnata al processo successivo.
- Al termine di un quanto di tempo, viene cambiato l'assegnamento alla CPU, e deve intercorrere un certo tempo, il tempo di **cambio di contesto** prima che il nuovo processo possa iniziare la propria esecuzione.