# Sistemi Operativi
# 06 luglio 2010
### Compito

Si risponda ai seguenti quesiti, giustificando le risposte.

1. 1. Che cosa significa che il kernel è prelazionabile? Che cosa sono i punti di prelazionabilità?
   2. In quali situazioni è importante che il kernel sia prelazionabile?
   3. Elencare vantaggi e svantaggi di un kernel prelazionabile.

   **Risposta:**

   1. Un kernel si dice prelazionabile quando consente la prelazione di un processo anche quando operi in kernel-mode (e.g., quando esegue il codice di una chiamata di sistema). I punti di prelazionabilità individuano delle posizioni nel codice delle chiamate di sistema in cui è possibile che il processo venga prelazionato: solitamente i punti di prelazionabilità vengono individuati in posizioni sicure del kernel, ovvero, laddove non vi siano in corso operazioni di modifica delle strutture dati di quest'ultimo (per non rischiare di creare delle inconsistenze).
   2. Nei sistemi operativi per applicazioni tradizionali il kernel non è prelazionabile in quanto non vi sono esigenze di risposte in tempi così rapidi come quelli tipici di questi sistemi. Invece nei sistemi real-time avere un kernel prelazionabile è fondamentale per evitare che una chiamata di sistema troppo lunga porti al fallimento dei vincoli temporali dei vari task.
   3. I vantaggi di un kernel prelazionabile consistono nell'avere dei tempi di risposta molto più rapidi, raggiungendo un grado di parallelismo molto più alto rispetto ai kernel tradizionali. Gli svantaggi sono rappresentati dalla maggior complessità del kernel stesso e dal rischio di deadlock/inconsistenze in caso di errori di progettazione dei punti di prelazione o dell'utilizzo delle primitive di sincronizzazione.

2. 1. In quali situazioni può essere attivato lo scheduling della CPU?
   2. Quando un algoritmo di scheduling è preemptive? Quali sono i vantaggi e gli svantaggi di un algoritmo preemptive?
   3. Gli scheduler Round Robin mantengono una lista di tutti i processi pronti, in cui ogni processo compare una sola volta. Che cosa potrebbe accadere se un processo apparisse due volte nella lista? Ci sono situazioni in cui questo potrebbe essere permesso?

   **Risposta:**

   1. Lo scheduling della CPU può essere attivato nelle seguenti circostanze:
      (a) un processo viene creato ed entra nella cosa dei pronti;
      (b) un processo passa dallo stato di esecuzione allo stato di attesa;
      (c) un processo passa dallo stato di esecuzione allo stato pronto;
      (d) un processo passa dallo stato di attesa allo stato pronto;
      (e) un processo termina.
   2. Un algoritmo di scheduling si dice preemptive se può interrompere l'esecuzione di un processo a favore di un altro processo e può quindi essere attivato ogni volta che un processo passa nella coda dei pronti (coda ready), oltre che ovviamente anche in altri casi. I vantaggi di un algoritmo preemptive sono essenzialmente dei tempi di risposta migliori e la garanzia che nessun processo riesca a monopolizzare la CPU senza rilasciarla. Gli svantaggi sono relativi alla condivisione dei dati fra processi; infatti, se un processo sta manipolando dei dati utilizzati anche da altri processi e viene prelazionato c'è il rischio che questi rimangano in uno stato inconsistente e generino così degli errori. Per evitare tutto ciò è necessario un attento uso di primitive come mutex e semafori per garantire un accesso esclusivo e corretto alle risorse condivise.
   3. Inserendo due puntatori allo stesso processo, quest'ultimo, nel caso in cui ad esempio i due puntatori siano consecutivi, si vedrebbe assegnare un tempo di CPU doppio rispetto agli altri processi. Infatti, scaduto il primo quanto di tempo, il primo puntatore al PCB verrebbe spostato in fondo alla coda, ma il secondo puntatore conferirebbe un ulteriore quanto di tempo al processo in questione. In generale quindi il processo beneficerebbe di un tempo di CPU doppio nel corso di ogni singola scansione della coda dei processi pronti. Un vantaggio derivante dall'utilizzo di questo sistema è quello di poter assegnare quanti di tempo diversi a seconda delle necessità di ogni singolo processo. Uno svantaggio potrebbe essere il rallentamento di alcuni processi a

causa dell'inserimento "non accorto" dei puntatori "doppi" nella coda. Un ulteriore aspetto negativo è costituito dal fatto che nel caso di puntatori consecutivi allo stesso PCB ci sarebbe comunque uno spreco di tempo per l'esecuzione delle system call dovute agli interrupt del timer di sistema per gestire l'assegnazione dei quanti di tempo (tali chiamate di sistema potrebbero essere evitate in quanto il processo che va in esecuzione rimane sempre lo stesso).

3. Si consideri un sistema con scheduling SJF con prelazione (cioè SRTF), ove $\alpha = 0,5$ e $\tau_0 = 30$ msec. All'istante 0 il processore si libera e tre processi, $P_1, P_2, P_3$, sono in coda ready. Finora i processi $P_1$, $P_2$ sono andati in esecuzione due volte con CPU burst $25, 40$ msec per $P_1$ e $30, 20$ msec per $P_2$; mentre $P_3$ è andato in esecuzione una volta con CPU burst di 60 msec.
   Si determini:

   1. Quale processo viene selezionato dallo scheduler all'istante 0?
   2. All'istante 10 msec entra nella coda ready un nuovo processo $P_4$ con CPU burst previsto di 20 msec. Il processo selezionato precedentemente è ancora in esecuzione. Che cosa succede?
   3. Che cosa succede quando il processo in esecuzione termina il suo burst?

   **Risposta:**

   1. Per $P_1$ abbiamo $\tau_1 = \frac{1}{2} \cdot 30 + \frac{1}{2} \cdot 25 = 15 + 12,5 = 27,5$, $\tau_2 = \frac{1}{2} \cdot 27,5 + \frac{1}{2} \cdot 40 = 33,75$, per $P_2$ abbiamo $\tau_1 = \frac{1}{2} \cdot 30 + \frac{1}{2} \cdot 30 = 30$, $\tau_2 = \frac{1}{2} \cdot 30 + \frac{1}{2} \cdot 20 = 15 + 10 = 25$ ed infine per $P_3$ abbiamo $\tau_1 = \frac{1}{2} \cdot 30 + \frac{1}{2} \cdot 60 = 45$. Quindi all'istante 0 SRTF sceglie $P_2$.
   2. All'istante 10 msec quando entra nella coda ready un nuovo processo $P_4$ con CPU burst previsto di 20 msec, continua $P_2$ perché gli mancano meno di 20 ms ($25 - 10 = 15$ ms) che è il burst previsto per $P_4$.
   3. Quando $P_2$ termina, va in esecuzione $P_4$.

4. Si descrivano le operazioni compiute dal driver di un dispositivo di I/O.

   **Risposta:** Le operazioni eseguite dai driver di un dispositivo di I/O sono le seguenti:

   1. Controllare i parametri passati.
   2. Accodare le richieste in una coda di operazioni (soggette a scheduling).
   3. Eseguire le operazioni, accedendo al controller.
   4. Passare il processo in modo wait (I/O interrupt-driven), o attendere la fine dell'operazione in busy-wait.
   5. Controllare lo stato dell'operazione nel controller.
   6. Restituire il risultato.

5. Relativamente all'allocazione dei blocchi nei dischi, si spieghi brevemente come funziona lo schema di allocazione contigua, illustrandone pregi e difetti. Soffre di frammentazione esterna?

   **Risposta:** Lo schema di allocazione contigua prevede che ogni file occupi un insieme di blocchi contigui sul disco. I vantaggi sono la semplicità in quanto basta conoscere il blocco iniziale e la lunghezza per accedere al file; inoltre l'accesso random è facile da implementare. I difetti principali sono l'impossibilità per i file di crescere (a meno di deframmentazione) e la frammentazione esterna.

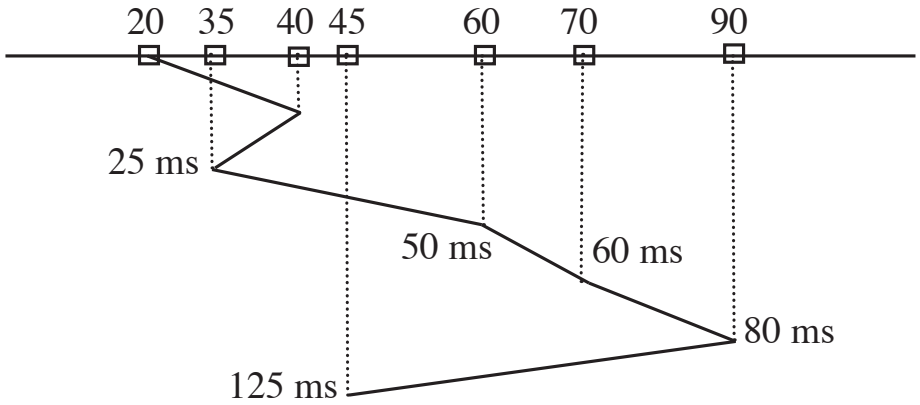
6. Si consideri un disco gestito con politica SSTF. Inizialmente la testina è posizionata sul cilindro 20; lo spostamento ad una traccia adiacente richiede 1 ms. Al driver di tale disco arrivano richieste per i cilindri 70, 60, 35, 45, 90, rispettivamente agli istanti 0 ms, 15 ms, 20 ms, 45 ms, 55 ms. Si trascuri il tempo di latenza.

   1. In quale ordine vengono servite le richieste?
   2. Il tempo di attesa di una richiesta è il tempo che intercorre dal momento in cui è sottoposta al driver a quando viene effettivamente servita. Qual è il tempo di attesa medio per le cinque richieste in oggetto?

   **Risposta:**

# Sistemi Operativi
# 06 luglio 2010
### Compito

1. Le richieste vengono servite nell'ordine 35, 60, 70, 90, 45:




2. Il tempo di attesa medio per le cinque richieste in oggetto è
$\frac{(25-20)+(50-15)+(60-0)+(80-55)+(125-45)}{5} = \frac{5+35+60+25+80}{5} = \frac{205}{5} = 41\ ms.$