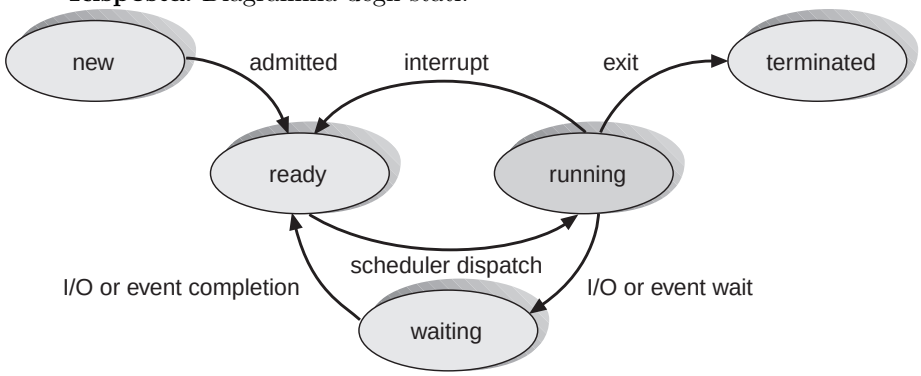
- Che cos'è un sistema operativo?

  Risposta: A questa domanda non c'è una definizione completa ed esauriente; infatti, a seconda del contesto, un sistema operativo può dover svolgere una o più delle seguenti funzioni:

  - deve agire da intermediario tra l'utente/programmatore e l'hardware del calcolatore, astraendo dalle peculiarità e dai dettagli di basso livello di quest'ultimo.
  - Svolge il ruolo di *assegnatore di risorse*, ovvero, gestisce ed alloca efficientemente le risorse finite della macchina.
  - Funge da *programma di controllo*, ovvero, controlla l'esecuzione dei programmi e le operazioni sulle risorse del sistema di calcolo, garantendo una condivisione corretta rispetto al tempo e rispetto allo spazio.

- Si tracci il diagramma degli stati dei processi e si diano esempi che causano transizioni da uno stato all'altro.

  Risposta: Diagramma degli stati:



  Esempi di transizioni da uno stato all'altro:

  - da new a ready: il processo viene inserito nella ready queue;
  - da ready a running: la CPU viene assegnata al processo che quindi esegue le proprie istruzioni;
  - da running a ready: il processo viene prelazionato e quindi costretto ad interrompere la propria esecuzione (ad esempio perché ha terminato il proprio quanto di tempo) per finire nella ready queue;
  - da ready a running: il processo viene scelto dallo short term scheduler per l'esecuzione;
  - da running a waiting: il processo attende il completamento di un evento o di un'operazione di I/O;
  - da waiting a ready: il completamento dell'evento/operazione di I/O è avvenuto e quindi il processo termina la propria attesa, venendo reinserito nella ready queue;
  - da running a terminated: il processo termina la propria esecuzione.

- Si consideri un sistema con scheduling SRTF, ove $\alpha = 0,5$ e $\tau_0 = 20\ msec$. 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 20, 40 msec per P_1 e 25, 30 msec per P_2; mentre P_3 è andato in esecuzione una volta con CPU burst di 50 msec.

  (a) Quale processo viene selezionato dallo scheduler all'istante 0?

  (b) All'istante 8 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?

  (c) Che cosa succede quando il processo in esecuzione termina il suo burst?

  Risposta:

  (a) Viene selezionato P_2 perché il suo prossimo CPU burst previsto (26,25 msec) è più basso di quelli di P_1 (30 msec) e P_3 (35 msec). La formula per calcolare i CPU burst previsti è $\tau_{n+1} = \alpha t_n + (1-\alpha)\tau_n$; quindi, quando $\alpha = 0,5$, si ha $\tau_{n+1} = \frac{t_n+\tau_n}{2}$.

  (b) Continua l'esecuzione P_2 in quanto il suo CPU burst previsto è 26,25 msec ed ha eseguito per 8 msec; quindi il tempo rimanente stimato è 18,25 msec che è inferiore al CPU burst previsto per P_4 (20 msec).

  (c) Viene selezionato di nuovo P_2 se il suo prossimo CPU burst previsto è minore o uguale a 20 msec (burst previsto per P_4, che è quello minimo fra gli altri processi), altrimenti viene selezionato P_4.

- (*) Si consideri un sistema con memoria paginata ad un livello, la cui page table sia mantenuta in memoria principale. Il tempo di accesso alla memoria principale sia $t = 50ns$.

  1. Qual è il tempo effettivo di accesso alla memoria (EAT: Effective Access Time)?
  2. Aggiungendo un TLB (Translation Look-aside Buffer), con tempo di accesso $\epsilon = 1ns$, quale hit rate (α) dobbiamo avere per un degrado delle prestazioni del 10% rispetto a t?

  Risposta:

  1. Il tempo effettivo di accesso alla memoria è $2t$, ovvero, 100 ns; infatti sono necessari 50 ns per accedere alla page table entry corretta e 50 ns per accedere alla locazione nel frame fisico in memoria.
  2. Un degrado del 10% rispetto a t significa un EAT pari a $1,1 \cdot t$, ovvero, 55 ns. Quindi si ha quanto segue (α rappresenta l'hit rate):

$$\begin{array}{rcl} EAT & = & \epsilon + \alpha t + (1-\alpha)2t \\ 55 & = & 1 + 50\alpha + (1-\alpha)\cdot 100 \\ 55 & = & 101 - 50\alpha \end{array}$$

  da cui si ricava $\alpha = \frac{46}{50} = 0,92$ (92%).

- Spiegare come funziona una tabella delle pagine invertita. Quando è necessario ricorrere a questa tecnica? Quante di queste tabelle è necessario mantenere nel sistema?

  Risposta: L'idea alla base di questa tecnica è di mantenere una tabella con una entry per ogni frame fisico della memoria, non per ogni pagina virtuale. Le informazioni contenute in ogni entry sono il numero della pagina (virtuale) memorizzata in quel frame ed informazioni riguardanti il processo che possiede la pagina. In questo modo diminuisce la memoria necessaria per memorizzare le page table, a discapito del tempo di accesso alla tabella stessa. Quest'ultimo ovviamente aumenta dato che, per tradurre un indirizzo virtuale nel corrispondente indirizzo fisico, è necessario scandire tutta la tabella alla ricerca di un'entry che contenga il numero di pagina virtuale che stiamo cercando di mappare.

  Questa tecnica è necessaria quando si ha a che fare con spazi di indirizzamento molto grandi (tipicamente a 64bit) e una page table ordinaria non potrebbe di conseguenza risiedere in memoria principale a causa del numero di entry troppo elevato.

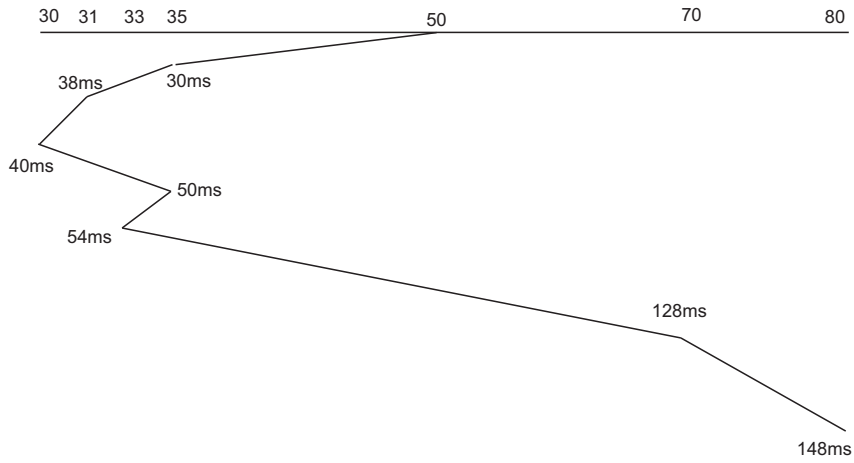
  Siccome in ogni entry, oltre al numero di pagina virtuale memorizzata nel frame, sono presenti anche informazioni che consentono di individuare il processo a cui appartiene la pagina, è sufficiente mantenere una sola tabella nel sistema.

- Descrivere i passi eseguiti durante un trasferimento di dati da un dipositivo di I/O ad un buffer in memoria in modalità DMA.

  Risposta: In questo caso il trasferimento avviene direttamente fra il dispositivo di I/O e la memoria fisica senza l'intervento della CPU (se si escludono il momento iniziale e quello finale dell'operazione). Inizialmente il controller DMA viene inizializzato specificando la periferica da cui prelevare i dati, l'indirizzo fisico X della memoria in cui memorizzare questi ultimi ed il numero C dei byte complessivi da trasferire. A questo punto il controller della periferica inizia il trasferimento, inviando ogni singolo byte al controller DMA. Quest'ultimo acquisisce il controllo del bus della memoria memorizzando il byte all'indirizzo X; in seguito incrementa l'indirizzo per il prossimo byte e decrementa il contatore C. Quando quest'ultimo raggiunge il valore 0, il trasferimento si è concluso e viene inviato un interrupt alla CPU per segnalare l'evento.

- (*) Si consideri un disco gestito con politica SSTF. Inizialmente la testina è posizionata sul cilindro 50; lo spostamento ad una traccia adiacente richiede 2 ms. Al driver di tale disco arrivano richieste per i cilindri 30, 31, 70, 33, 80, rispettivamente agli istanti 0 ms, 30ms, 40 ms, 50 ms, 70 ms. Si trascuri il tempo di latenza. In quale ordine vengono servite le richieste?

  Risposta: Le richieste vengono soddisfatte nell'ordine 31, 30, 33, 70, 80, come evidenziato nel diagramma seguente:



Il punteggio attribuito ai quesiti è il seguente: 3, 3, 2, 2, 2, 3, 3, 4, 4, 6 (totale: 32).