Sistemi Operativi
17 settembre 2010
Compito

Si risponda ai seguenti quesiti, giustificando le risposte.

1. Si descriva la struttura dei sistemi operativi.

   Risposta: Per quanto riguarda la struttura dei sistemi operativi, l'approccio più semplice è quello relativo all'MS-DOS (pensato per fornire le massime funzionalità nel minore spazio possibile) che non è diviso in moduli e presenta un minimo di struttura senza peraltro separare chiaramente le sue interfacce ed i suoi livelli funzionali. Nel sistema UNIX originale invece si aveva una distinta separazione fra programmi di sistema e kernel. Quest'ultimo era inteso come tutto ciò che stava tra le chiamate di sistema e l'hardware, implementando molte funzionalità in un solo livello: il file system, lo scheduling della CPU, la gestione della memoria ed altre funzioni.

   I sistemi operativi successivi (ad esempio il THE OS, OS/2 e Linux) hanno raffinato l'approccio del sistema UNIX originale, proponendo una struttura stratificata. In pratica il sistema operativo è diviso in un certo numero di strati (livelli): ogni strato è costruito su quelli inferiori. Lo strato di base (livello 0) consiste nell'hardware, mentre il più alto rappresenta l'interfaccia utente. Secondo il principio della modularità, gli strati sono pensati in modo tale che ognuno utilizza funzionalità (operazioni) e servizi solamente degli strati inferiori. Tale approccio viene portato all'estremo dalle macchine virtuali che trattano l'hardware e il sistema operativo come se fossero interamente hardware. In questo caso una macchina virtuale fornisce un'interfaccia uniforme all'hardware sottostante ed il sistema operativo impiega le risorse del calcolatore fisico per creare le macchine virtuali:

   (a) lo scheduling della CPU crea l'illusione che ogni processo abbia il suo processore dedicato;

   (b) la gestione della memoria crea l'illusione di una memoria virtuale per ogni processo;

   (c) lo spooling può implementare delle stampanti virtuali;

   (d) lo spazio disco può essere impiegato per creare "dischi virtuali".

2. (a) Che cosa si intende per processi CPU-bound e processi I/O-bound? Quali effetti si possono avere se viene data la precedenza a processi CPU-bound?

   (b) Si diano esempi di algoritmi di scheduling della CPU che privilegiano i processi I/O-bound.

   Risposta:

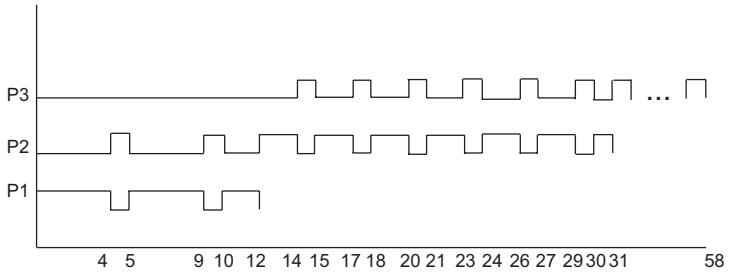
   (a) I processi possono essere classificati come I/O-bound quando presentano lunghi periodi di I/O e brevi periodi di calcolo. Al contrario si parla di processi CPU-bound quando si hanno lunghi periodi di intensiva computazione e pochi (possibilmente lunghi) cicli di I/O. Privilegiando i processi CPU-bound si allungano i tempi di risposta percepiti dagli utenti che stanno eseguendo dei processi interattivi (I/O-bound).

   (b) In generale una classe di algoritmi di scheduling della CPU che privilegiano i processi I/O-bound è costituita dagli scheduling con code multiple e con retroazione (feedback) in cui la coda a maggior priorità viene riservata ai processi interattivi, mentre quelli CPU-bound vengono posizionati nelle code a minor priorità. Fra i sistemi operativi di uso comune l'algoritmo di scheduling di Linux è un esempio di algoritmo che favorisce i processi interattivi (I/O-bound) dato che assegna un bonus di priorità dinamica a questi ultimi. Anche lo scheduling di Windows favorisce i processi I/O-bound dato che sceglie sempre dalla coda a priorità maggiore, ma la priorità di un thread utente può essere temporaneamente maggiore di quella base (per effetto delle "spinte") per thread che attendono dati di I/O (spinte fino a +8) e per dare maggiore reattività a processi interattivi (+2).

3. I processi P_1, P_2, P_3, di priorità decrescente e con richieste di CPU totali di 10, 15, 20 s, sono in coda ready nell'ordine indicato. L'algoritmo di scheduling della CPU è con prelazione (e i processi prelazionati vengono messi in fondo alla coda ready). Ogni processo P_i ha un comportamento ciclico: in ogni ciclo consuma c_i secondi di CPU ed esegue operazioni di I/O per o_i secondi, dove $c_i = 2^{3-i}$ e $o_i = 0,5^{3-i}$. L'algoritmo di scheduling viene eseguito ad intervalli di 1 secondo. Si calcolino i tempi di attesa per i processi, nel caso l'algoritmo di scheduling sia RR con quanto pari a 2s e con prelazione in base alla priorità.

   Risposta: I tempi di attesa Δ_i dei processi P_i sono $\Delta_1 = 2s$, $\Delta_2 = 16s$ e $\Delta_3 = 38s$. Infatti abbiamo la seguente situazione:

	$\text{CPU}_t ot$	c_i	o_i
P_1	10	4	0,25
P_2	15	2	0,5
P_3	20	1	1



4. Cosa si intende per anomalia di Belady? Si citi un algoritmo di rimpiazzamento delle pagine che soffre di questo problema ed uno che invece ne è immune.

   Risposta: Per anomalia di Belady si intende il fenomeno per cui, nonostante si incrementi la memoria fisica disponibile e quindi il numero di frame totali, non è detto che i page fault diminuiscano. Un algoritmo di rimpiazzamento delle pagine che soffre di questo problema è l'algoritmo FIFO (First-In First-Out), mentre LRU (Least Recently Used) e tutti gli algoritmi di stack ne sono immuni.

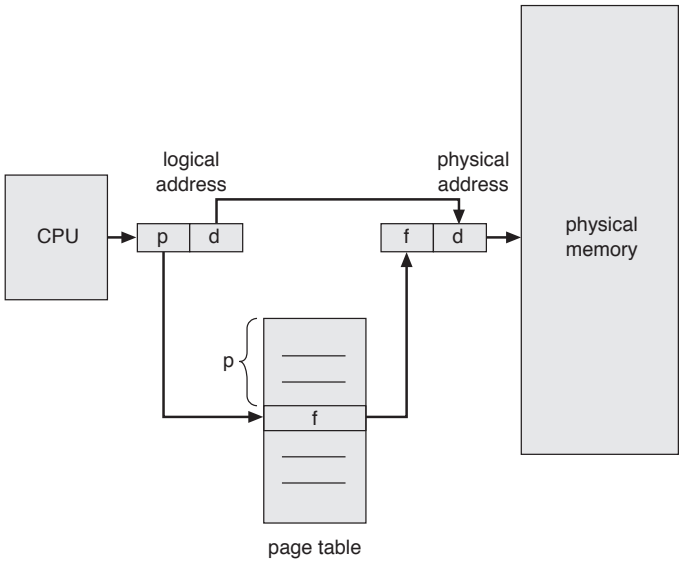
5. Si illustri brevemente il meccanismo di paginazione della memoria con il relativo schema di traduzione dell'indirizzo virtuale in indirizzo fisico.

   Risposta: La paginazione è una tecnica di allocazione della memoria non contigua che prevede:

   - la suddivisione della memoria fisica in *frame* (blocchi di dimensione fissa, una potenza di 2, tra 512 e 8192 byte);
   - la suddivisione della memoria logica in *pagine* (della stessa dimensione dei frame).

   Quindi, per eseguire un programma di n pagine, servono n frame liberi in cui caricare il programma. È compito del sistema operativo tenere traccia dei frame liberi. Non esiste frammentazione esterna e la frammentazione interna è ridotta all'ultima pagina allocata al processo.

   Per quanto riguarda la traduzione degli indirizzi logici in indirizzi fisici, ogni indirizzo generato dalla CPU si suddivide in un numero di pagina p e uno spiazzamento (offset) d all'interno della pagina. Il numero di pagina viene utilizzato come indice nella *page table* del processo correntemente in esecuzione: ogni entry della page table contiene l'indirizzo di base del frame fisico f che contiene la pagina in questione. Tale indirizzo di base f viene combinato (giustapposto come prefisso) con lo spiazzamento d per definire l'indirizzo fisico da inviare all'unità di memoria:



6. Si spieghi brevemente come funzionano i modi di accesso (permessi) e quali sono le classi di utenti per i file in UNIX.

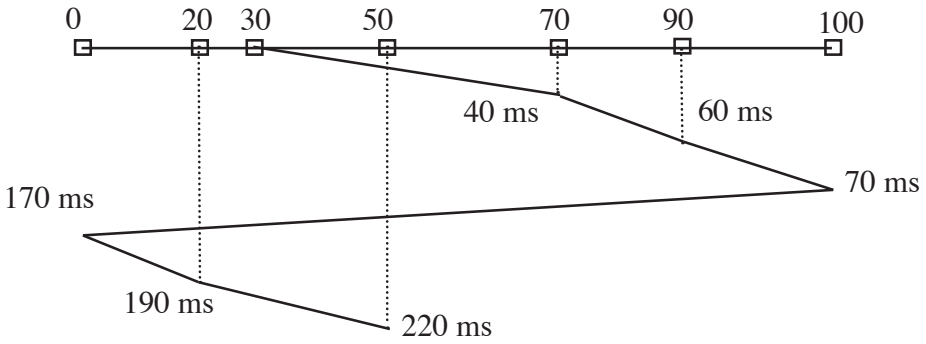
   Risposta: UNIX implementa una versione semplificata di ACL con tre modi di accesso (permessi): lettura (read), scrittura (write), esecuzione (execute) e tre classi di utenti, per ogni file, ovvero, proprietario (owner), gruppo (group) e resto degli utenti (other). Se chi tenta di accedere al file è il proprietario, vengono applicati i permessi ad esso relativi per determinare la liceità dell'operazione. Altrimenti, se chi tenta di accedere appartiene al gruppo assegnato al file, allora si applicano i permessi relativi al gruppo. Se nemmeno questo è il caso, si applicano i permessi relativi al resto degli utenti.

7. Si consideri un disco (con i cilindri numerati da 0 a 100) gestito con politica C-SCAN. Inizialmente la testina è posizionata sul cilindro 30 con direzione di servizio ascendente; lo spostamento ad una traccia adiacente richiede 1 ms. Al driver di tale disco arrivano richieste per i cilindri 90, 20, 70, 50, rispettivamente agli istanti 0 ms, 10 ms, 30 ms, 45 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 quattro richieste in oggetto?

   Risposta:

   1. Le richieste vengono servite nell'ordine 70, 90, 20, 50:



   2. Il tempo di attesa medio per le quattro richieste in oggetto è $\frac{(40-30)+(60-0)+(190-10)+(220-45)}{4} = \frac{10+60+180+175}{4} = \frac{425}{4} = 106,25\ ms.$