SISTEMI OPERATIVI
16 febbraio 2012

Cognome:_____ **Nome:**_____
Matricola:_____

1. Ricordate che non potete usare calcolatrici o materiale didattico.
2. Ricordate che potete consegnare al massimo tre prove scritte per anno accademico.

ESERCIZI RELATIVI ALLA PARTE DI TEORIA DEL CORSO

ESERCIZIO 1 (5 punti)

a) Si consideri il problema dei lettori e scrittori visto a lezione, dove i codici del generico scrittore e del generico lettore sono riportati qui di seguito.

Inserite le operazioni di wait e signal mancanti necessarie per il funzionamento del sistema secondo la soluzione vista a lezione, indicando anche il semaforo mancante ed il suo valore di inizializzazione.

semafori e variabili condivise necessarie con relativo valore di inizializzazione:
semaphore mutex = 1;
semaphore scrivi = 1;
int numlettori = 0;

"scrittore"
{
wait(scrivi);
Esegui la scrittura del file
signal(scrivi)
}

"lettore"
{
wait(mutex);

numlettori++;

if numlettori == 1 **wait(scrivi);**

signal(mutex);

... leggi il file ...

wait(mutex);

numlettori--;

if numlettori == 0 **signal(scrivi);**

signal(mutex);

b) La soluzione del problema dei lettori e scrittori vista a lezione garantisce l'assenza di starvation? (motivate la vostra risposta)

No, infatti un qualsiasi processo scrittore potrebbe dover attendere all'infinito senza riuscire a ad entrare in sezione critica. Al contrario i processi lettori sono liberi da starvation (in altre parole, non è garantita l'attesa limitata

c) Riportate il diagramma di stato della vita di un processo e rispondete alla seguente domanda: in quali transizioni da uno stato all'altro deve intervenire il sistema operativo?

Si veda il lucido della sezione 3.1.2. In tutte, in quanto è il SO che sposta i processi (i loro PCB) da uno stato all'altro.

d) Descrivete brevemente un pregio ed un difetto dei semafori, in quanto strumenti di sincronizzazione, implementati come visto a lezione.

Pregio: evitano il busy waiting
Difetto: non sono primitive di sincronizzazione strutturate, per cui un loro uso scorretto può portare alla violazione della mutua esclusione, a starvation o deadlock.

e) In quali casi specifici interviene uno *scheduling con diritto di prelazione* (*preemptive scheduling*)?

1) Quando scade il quanto di tempo assegnato al processo in esecuzione e 2) quando arriva in coda di ready un processo più importante di quello correntemente in esecuzione.

ESERCIZIO 2 (5 punti)

In un sistema la memoria fisica è divisa in 2^{21} frame, un indirizzo logico è scritto su 32 bit, e all'interno di una pagina, l'offset massimo è 3FF.

a) Quanti byte occupa la la page table più grande del sistema? (motivate numericamente la vostra risposta)

Un frame/pagina è grande 2^{10} = 1024 byte, e quindi la page table più grande può avere $2^{(32-10)} = 2^{22}$ entry. Nel sistema vi sono 2^{21} frame, per cui sono necessari tre byte per scrivere il numero di un frame, e quindi la page table più grande occupa ($2^{22} \cdot$ 3) byte = 12 Mbyte

b) Quale dimensione minima dovrebbero avere le pagine di questo sistema per essere certi di non dover ricorrere ad una paginazione a più livelli? (motivate la vostra risposta, e per questa domanda assumete di usare 4 byte per scrivere il numero di un frame all'interno di una page table)

Poniamo 32 = m + n (m = bit usati per scrivere un numero di pagina, n = bit usati per scrivere l'offset). Allora il numero di entry della PT più grande, moltiplicato per la dimensione di una entry deve poter essere contenuto in una pagina/frame, ossia: $2^m \cdot 2^2 \leq 2^n$
Da cui: $m + 2 \leq n$. Poiché $m = 32 - n$; risolvendo il semplice sistema si ha $n \geq 17$, ossia le pagine devono almeno essere grandi 2^{17} = 128 Kbyte.

c) In questo sistema è necessario usare il bit di validità delle pagine? (motivate la vostra risposta).

Si perché essendo lo spazio di indirizzamento logico più grande di quello fisico, è necessario implementare la memoria virtuale.

ESERCIZIO 3 (4 punti)

Un hard disk ha la capienza di 2^{38} byte, ed è formattato in blocchi da 2048 byte.

a) Quanti accessi al disco sono necessari per leggere l'ultimo blocco di un file A della dimensione di 8192 byte, assumendo che sia già in RAM il numero del primo blocco del file stesso e che venga adottata una allocazione concatenata dello spazio su disco? (motivate la vostra risposta)

5. Ogni blocco infatti memorizza 2044 byte di dati più 4 byte di puntatore al blocco successivo (infatti, $2^{38}/2^{11} = 2^{27}$), per cui sono necessari 5 blocchi per memorizzare l'intero file.

b) Qual è lo spreco di memoria dovuto alla frammentazione interna nella memorizzazione di A (motivate la risposta)?

L'hard disk è suddiviso in $2^{38}/2^{11} = 2^{27}$ blocchi, sono necessari 4 byte per memorizzare un puntatore al blocco successivo, e ogni blocco contiene 2044 byte di dati. Il quinto blocco memorizzerà quindi 16 byte del file, e la frammentazione interna corrisponde a 2048 – 16 = 2032 byte (2028 se si considerano non sprecati i 4 byte del quinto blocco che contengono il puntatore, non utilizzato, al blocco successivo)

c) Quanto sarebbe grande, in megabyte, la FAT di questo sistema? (motivate numericamente la vostra risposta)

La FAT è un array con una entry per ciascun blocco dell'hard disk e che contiene il numero di un blocco, per cui: 2^{27} x 2^2 byte = 512 megabyte

d) In una cartella A di un sistema Unix viene creato un nuovo file di nome *pippo*. Assumendo che la working directory sia A, indicate i comandi necessari per eseguire le seguenti operazioni:

1) creare un nuovo hard link di nome *pluto* a *pippo*: ln pippo pluto
2) creare un nuovo symbolic link di nome *topolino* a *pluto*: ln –s pluto topolino
3) visualizzare i permessi, la dimensione e il numero dell'i-node di *pippo*: ls –li pippo
4) copiare *pippo* su *minnie (minnie* prima non esisteva*)*: cp pippo minnie
5) rimuovere *pluto*: rm pluto
6) qual è a questo punto il valore del link-counter di pippo? 1