SISTEMI OPERATIVI
26 settembre 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)

In un sistema operativo che adotta uno scheduling con diritto di prelazione, quattro processi arrivano al tempo indicato e consumano la quantità di CPU indicata nella tabella sottostante)

Processo	T. di arrivo	Burst
P1	0	13
P2	3	8
P3	6	4
P4	8	1

a)
Qual è il waiting time medio migliore (ossia ottimale) che potrebbe essere ottenuto per lo scheduling dei quattro processi della tabella? RIPORTATE IL DIAGRAMMA DI GANTT USATO PER IL CALCOLO. (lasciate pure i risultati numerici sotto forma di frazione, e indicate quali assunzioni fate)

Diagramma di GANT, assumendo come algoritmo di scheduling SJF preemptive:
(0)….P1 …(3) …. P2….(6)….P3….(8)….P4….(9)….P3…(11)….P2…(16)…P1….(26)

Waiting time medio:
P1 = (26 – 0) – 13 = 13;
P2 = (16 – 3) – 8 = 5;
P3 = (11 – 6) – 4 = 1;
P4 = (9 – 8) – 1 = 0;
waiting time medio = 19/4

b)
All'interno di un sistema operativo, un certo processo P è correntemente in stato di "Ready to Run", e si sa che, una volta acquisita la CPU, non dovrà più rilasciarla volontariamente prima di aver terminato la propria esecuzione (in altre parole, non dovrà più eseguire operazioni di I/O, di sincronizzazione o di comunicazione con altri processi).
Quale/quali, tra gli algoritmi di scheduling **FCFS**, **SJF preemptive, SJF non-preemptive**, **round robin** garantisce/garantiscono che il processo P riuscirà a portare a termine la propria computazione? (motivate la vostra risposta, assumendo che SJF possa effettivamente essere implementato)

FCFS, e **round robin**. Infatti, nel caso di SJF (preemptive e non) potrebbe sempre arrivare in coda di ready un processo che deve usare la CPU per un tempo minore di quanto rimane da eseguire a P.

c)
Riportate lo pseudocodice che descrive la corretta implementazione dell'operazione di WAIT. Quale significato pratico ha sull'uso del semaforo, il valore corrente della sua variabile semaforica S?

Per il codice si vedano i lucidi della sezione 6.5.2.

Se S > 0, indica quanti processi possono superare il semaforo senza addormentarsi nella sua coda di wait.
Se S <=0, |S| indica quanti processi sono addormentati nella coda di wait del semaforo.

ESERCIZIO 2 (5 punti)

Si consideri un sistema in cui in una tabella delle pagine di un processo l'indice più grande usabile nella tabella delle pagine di quel processo può essere 7FFF. Un indirizzo fisico del sistema è scritto su 25 bit, e la RAM è suddivisa in 4000 (esadecimale) frame.

a)
Quanto è grande, in megabyte, lo spazio di indirizzamento logico del sistema (esplicitate i calcoli che fate)?

4000(esadecimale) = 2^{14}, per cui un numero di frame è scritto su 14 bit, e la dimensione di un frame, e quindi di una pagina, è di $2^{\wedge 11}$ byte (25 - 14 = 11). Poiché il numero più grande di una pagina è 7FFF, ci possono essere al massimo $2^{\wedge 15}$ pagine, e lo spazio di indirizzamento logico è di $2^{\wedge 15}$ x $2^{\wedge 11}$ byte (pari a circa 64 megabyte).

b)
Indicate tutte e sole le informazioni contenute in una entry di una tabella delle pagine di questo sistema, se il sistema usa l'algoritmo di rimpiazzamento della seconda chance.

Il numero del frame che contiene la pagina corrispondente, il bit di validità della pagina, il reference bit.

c)
Nel caso non si verifichino mai page fault, qual è, in nanosecondi, il tempo medio di accesso in RAM del sistema se viene usato un TLB con un tempo di accesso di 5 nanosecondi, un hit-ratio del 90% e un tempo di accesso in RAM di 0,1 microsecondi? (è sufficiente riportare l'espressione aritmentica che fornisce il risultato finale)

Tmedio = 0,90 * (100+5) + 0,1 * (2*100 + 5) nanosecondi

d)
Elencate tre svantaggi dell'uso delle librerie statiche

Non possono essere condivise tra più processi, per cui occupano più spazio in RAM.
Vengono caricate in RAM comunque, anche se non verrà usata nessuna delle loro funzioni, per cui i processi partono più lentamente.
Se vengono aggiornate occorre ricompilare i programmi che le usano

e)
Commentate la seguente affermazione: in un qualsiasi sistema operativo che implementi la memoria virtuale, l'effettivo tempo di esecuzione di un programma su quel sistema può dipendere pesantemente dal modo con cui il programma accede ai propri dati.

L'affermazione è vera. Ad esempio, l'accesso per colonne ad array memorizzati per riga può produrre un elevatissimo numero di page fault, e quindi un forte rallentamento nell'esecuzione del programma.

ESERCIZIO 3 (4 punti)

a) Descrivete (se preferite usando un disegno) il metodo di allocazione dello spazio su disco adottato dal sistema operativo Unix.

Si vedano i lucidi della sezione 11.4.3.

b) Scegliete una dimensione per i blocchi dell'hard disk e il numero di bit usati per scrivere il numero di un blocco, e calcolate di conseguenza la dimensione massima che può avere un file di quel sistema (è sufficiente riportare l'espressione che permette di calcolare la dimensione del file).

Blocco: 1024 byte, puntatore a blocco: 4 byte.
Dimesione massima: $10 \times 1k + 256 \times 1k + 256^2 \times 1k + 256^3 \times 1k$

c) come è fatta la struttura interna di una qualsiasi cartella unix, e quali entry sono sempre presenti in qualsiasi cartella unix? (se preferite usate un disegno)

È un array in cui ciascuna entry è formata dal nome di un file e dal numero di un index-node che contiene tutte le informazioni del file stesso. Qualsiasi cartella unix contiene sempre almeno le due entry "." E ".."

d) E' più veloce l'accesso ad un file attraverso un hard link o attraverso un symbolic link a quel file? Motivate la vostra risposta.

Attraverso un hard link, perché l'uso di un symbolic link richiede l'accesso a due i-node anziché ad uno solo.