Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Nome: _____ Cognome: _____ Matricola:_____ fila:___ posto: ___

Esercizio 1 (5 punti)

Si consideri un sistema dove la memoria è gestita con paginazione a domanda. La pagine logiche e i blocchi fisici hanno un'ampiezza di 2^{11} byte (= 2 kByte).

Nel sistema è realizzato un File System FAT-32 di 2Gbyte dove:

- i blocchi del disco hanno un'ampiezza di 2^{11} byte (= 2 kByte);
- gli elementi della FAT codificano con 4 byte gli indirizzi dei blocchi del disco;

La FAT è mappata in una regione *R* del kernel paginata e caricata a domanda come gli spazi virtuali dei processi, a partire dalla pagina 32 dello spazio virtuale di questa regione. Al tempo t, il contenuto (parziale) della tabella delle pagine della regione *R* è mostrato in tabella. Il campo P è il bit di presenza; sono omessi gli altri bit di controllo.

	…	94	95	96	97	98	99	100	101	103	104	105	106	107	108	109	110	111	112	113	114	…
Blocco		8140	--	8201	9100	9101	9102	9103	--	--	7300	7301	7302	7303	--	--	7310	7312	7320	--	--	
P		1	0	1	1	1	1	1	0	0	1	1	1	1	0	0	1	1	1	0	0	

Tabella delle pagine della regione *R*

Nel file system è presente il file *saturno*, che ha una lunghezza di 19.000 byte distribuiti in 10 blocchi logici, numerati da 0 a 9 e memorizzati ordinatamente nei seguenti blocchi fisici del disco: 33.000, 33.001, 33.002, 42.110, 35.560, 35.000, 35.001, 37.900, 36.000, 36.001.

Il file è stato aperto dal processo A e la posizione corrente del puntatore di lettura è 14149 (= $6*2^{11} + 1861$). La directory del file è caricata in memoria.

Al tempo *t* il processo A esegue una *read*, che legge 150 caratteri dal file *saturno* (a partire dalla posizione attuale del puntatore di lettura) Si chiede:

1. Il numero di blocchi del disco (che ospita il file system);
2. La dimensione della FAT, in numero di byte e di pagine;
3. Il numero di elementi della FAT contenuti in una pagina;
4. Quali sono le pagine logiche che contengono la FAT;
5. Quanti e quali blocchi logici del file *saturno* si devono leggere per leggere 150 byte dal file *saturno* (a partire dalla posizione attuale del puntatore di lettura)
6. Quali elementi della FAT devono essere letti a questo scopo;
7. A quali pagine logiche si deve accedere per leggere questi elementi;
8. Quanti *page fault* vengono provocati da questi accessi.

Soluzione

1. Numero di blocchi del disco: _____
2. La FAT ha _____ elementi, ciascuno di 4 byte;
   dimensione della FAT in byte: _____ = _____ Mbyte;
   dimensione della FAT in pagine: _____ pagine logiche
3. numero di elementi della FAT contenuti in una pagina logica: _____ elementi
4. pagine logiche occupate dalla FAT: _____.
5. Si leggono i blocchi logici: _____
6. Si leggono gli elementi della FAT di indice: _____
7. Si accede alle pagine logiche: _____
8. L'operazione provoca _____ page faults

Soluzione

1. Numero di blocchi del disco: 2Gbyte / 2Kbyte = 2^{20} blocchi
2. La FAT ha 2^{20} elementi, ciascuno di 4 byte;
   dimensione in byte $2^{20}* 4= 2^{22}$ byte = 4 Mbyte; dimensione in pagine: $2^{22}/2^{11} = 2^{11}$ pagine logiche
3. Il numero di elementi della FAT contenuti in una pagina logica: $2^{11}/ 4= 2^9$ elementi
4. Le pagine logiche occupate dalla FAT: sono quella di indice 32 e le successive 2^{11}- 1 pagine.
5. Per eseguire l'operazione si deve leggere unicamente il blocco fisico corrispondente al blocco logico di indice 6. Infatti il puntatore di lettura, che ha valore $6*2^{11} + 1861$, è posizionato sul carattere 1861 del blocco logico 6; l'ultimo dei 150 caratteri da leggere è il carattere 2011 del medesimo blocco (2011 < 2048):
6. Devono essere letti gli elementi gli elementi di indici 33.000 (individuato dalla directory), 33.001, 33.002, 42.110, 35.560, e 35.000, infatti è necessario raggiungere l'elemento di indice 35.000 per conoscere l'indirizzo 35.001 del blocco fisico corrispondente al blocco logico 6.
7. Tenendo conto del fatto che:
   - L'indirizzo 33.000 è contenuto nella directory del file
   - L'indirizzo 33.001 è contenuto nell'elemento della FAT 33.000, che appartiene alla pagina logica 33.000 div 2^9=64 della FAT corrispondente alla pagina 32+64=96 della regione R
   - L'indirizzo 33.002 è contenuto nell'elemento della FAT 33.001, che appartiene alla pagina logica 33.001 div 2^9=64 della FAT corrispondente alla pagina 32+64=96 della regione R

- L'indirizzo 42.110 è contenuto nell'elemento della FAT 33.002, che appartiene alla pagina logica 33.002 div 2^9=64 della FAT corrispondente alla pagina 32+64=96 della regione R
- L'indirizzo 35.560 è contenuto nell'elemento della FAT 42.110, che appartiene alla pagina logica 42.110 div 2^9=82 della FAT, corrispondente alla pagina 32+82=114 della regione R
- L'indirizzo 35.000è contenuto nell'elemento della FAT 35.560, che appartiene alla pagina logica 35.560 div 2^9=69 della FAT, corrispondente alla pagina 32+69=101 della regione R,

Ne consegue che è necessario leggere le pagine logiche: 96, 114, 101

8. Si verificano 2 page *fault,* per l'accesso alle pagine 114 e 101.

Esercizio 2 (5 punti)

Si consideri un sistema che gestisce la memoria con paginazione a domanda, adottando una politica, ispirata a quella di Windows, che assegna a ogni processo MaxBlocchi blocchi per il caricamento del proprio working set. Questo numero può essere superato transitoriamente per effetto degli errori di pagina del processo e può essere ridefinito dal processo di sistema WorkingSetManager, che interviene periodicamente.

E` definita una tabella delle pagine per ogni processo, con campi *blocco* (indice del blocco assegnato alla pagina; vuoto se la pagina non è caricata) e *Bit R* (bit di uso della pagina). Inoltre è definita una *Pila di Indici di Blocco Liberi*, sulla quale si eseguono operazioni di *pop* e di *push*, rispettivamente per prelevare l'indice di un blocco da assegnare a una pagina o per restituire l'indice di un blocco divenuto libero. L'algoritmo di sostituzione è il *Second Chance locale,* con puntatore che scorre ciclicamente sugli indici della pagine assegnate nella tabella delle pagine del processo interessato.

Al verificarsi di un errore di pagina la pagina riferita viene caricata in un blocco libero, anche se il numero di blocchi assegnati al processo è maggiore o uguale a MaxBlocchi.

Il processo WorkingSetManager, che interviene periodicamente, applica la seguente politica:

a) se tutte le pagine hanno il bit R=1, incrementa di 1 il valore di MaxBlocchi senza eseguire scaricamenti;

b) altrimenti:

b1) se esistono almeno 3 pagine con bit R=0, decrementa di 1 MaxBlocchi

b2) applicando ripetutamente l'algoritmo *Second Chance locale,* scarica pagine fino a quando il numero delle pagine caricate diviene uguale al valore (eventualmente modificato) di MaxBlocchi.

A un certo istante, quando il tempo virtuale del processo A è t=10, la tabella delle pagine del processo A ha il contenuto riportato in tabella, con il puntatore dell'algoritmo *Second Chance* posizionato sull'elemento 8, e si ha MaxBlocchi= 6. Il contenuto della *Pila di Indici di Blocco Liberi*, è 20→25→16→19→31, dove l'elemento 20 corrisponde al *top* della pila.

								↓		
Pagina	0	1	2	3	4	5	6	7	8	9
Blocco	7	5	-	8	9	-	15	-	11	-
Bit R	0	0		1	0		0		0	

A partire da questo istante il processo A avanza ed esegue i seguenti riferimenti alla memoria:

- al tempo 10 riferisce la pagina 3
- al tempo 11 riferisce la pagina 2
- al tempo 12 riferisce la pagina 0
- al tempo 13 riferisce la pagina 9
- al tempo 14 riferisce la pagina 8

Dopo quest'ultimo riferimento interviene il processo WorkingSetManager, che applica la politica sopra definita.

Si chiede:

a) gli eventuali caricamenti effettuati ai tempi 10, 11, 12, 13, 14;

b) la configurazione della tabella delle pagine del processo A e il contenuto della *Pila di Indici di Blocco Liberi* subito prima dell'intervento del processo WorkingSetManager

c) il valore di MaxBlocchi processo A dopo l'intervento del processo WorkingSetManager;

d) la configurazione della tabella delle pagine del processo A e il contenuto della *Pila di Indici di Blocco Liberi* subito dopo dell'intervento del processo WorkingSetManager

Soluzione

a) caricamenti di pagine ai tempi 10, 11, 12, 13, 14:

- tempo 10 (riferita la pagina _____): caricata la pagina _____ nel blocco _____;
- tempo 11 (riferita la pagina _____): caricata la pagina _____ nel blocco _____;
- tempo 12 (riferita la pagina _____): caricata la pagina _____ nel blocco _____;
- tempo 13 (riferita la pagina _____): caricata la pagina _____ nel blocco _____;
- tempo 14 (riferita la pagina _____): caricata la pagina _____ nel blocco _____;

b) Configurazione della tabella delle pagine del processo A e contenuto della *Pila di Indici di Blocco Liberi* prima dell'intervento di Working Set Manager:

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Contenuto della Pila di Indici di Blocco Liberi: _____
Configurazione della tabella delle pagine:

Pagina	0	1	2	3	4	5	6	7	8	9
Blocco										
Bit R										

c) Valore di MaxBlocchi processo A dopo l'intervento del processo WorkingSetManager: _____

d) Configurazione della tabella delle pagine del processo A e contenuto della *Pila di Indici di Blocco Liberi* dopo l'intervento di Working Set Manager:

Contenuto della Pila di Indici di Blocco Liberi: _____
Configurazione della tabella delle pagine:

Pagina	0	1	2	3	4	5	6	7	8	9
Blocco										
Bit R										

Soluzione

a) caricamenti di pagine ai tempi 10, 11, 12, 13, 14:

- tempo 10 (riferita la pagina 3): caricata la pagina - nel blocco -;
- tempo 11 (riferita la pagina 2): caricata la pagina 2 nel blocco 20;
- tempo 12 (riferita la pagina 0): caricata la pagina - nel blocco -;
- tempo 13 (riferita la pagina 9): caricata la pagina 9 nel blocco 25;
- tempo 14: (riferita la pagina 8): caricata la pagina - nel blocco -;

Configurazione della tabella delle pagine del processo A e contenuto della *Pila di Indici di Blocco Liberi*
b) Prima dell'intervento di *Working Set Manager*

Contenuto della *Pila di Indici di Blocco Liberi*: 16→19→31
Configurazione della tabella delle pagine:

									↓	
Pagina	0	1	2	3	4	5	6	7	8	9
Blocco	7	5	20	8	9	-	15	-	11	25
Bit R	1	0	1	1	0		0		1	1

c) Dopo l'intervento di *Working Set Manager*
Valore di MaxBlocchi: 5
Contenuto della *Pila di Indici di Blocco Liberi*: 15→9→5→16→19→31
d) Configurazione della tabella delle pagine:

									↓	
Pagina	0	1	2	3	4	5	6	7	8	9
Blocco	7		20	8					11	25
Bit R	0		0	0					0	0

Esercizio 3 (5 punti)

In un file system UNIX i blocchi del disco hanno ampiezza di 1Kbyte e gli i-node contengono 10 indirizzi diretti e 3 indirizzi indiretti. Tutti gli indirizzi hanno una lunghezza di 32 bit.
Si chiede di calcolare:

1) il numero di puntatori che possono essere contenuti in ogni blocco indice,
2) il numero di blocchi indirizzabili con indirizzamento indiretto semplice,
3) il numero di blocchi indirizzabili con indirizzamento indiretto doppio
4) il numero di blocchi indirizzabili con indirizzamento indiretto triplo.

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

5) Calcolare la minima dimensione di un file affinché nella sua rappresentazione sia necessario ricorrere all'indirizzamento indiretto triplo.
6) Calcolare il numero di blocchi indice di primo, secondo e terzo livello necessari per rappresentare un file di dimensione 97.000 byte.
7) Calcolare il numero di blocchi indice di primo, secondo e terzo livello necessari per rappresentare un file di dimensione 16 Mbyte.

Soluzione

1) Numero di puntatori contenuti in un blocco indice: _____
2) Numero di blocchi indirizzabili con indirizzamento indiretto semplice: _____
3) Numero di blocchi indirizzabili con indirizzamento indiretto doppio: _____
4) Numero di blocchi indirizzabili con indirizzamento indiretto triplo: _____
5) Minima dimensione di un file affinché nella sua rappresentazione sia necessario ricorrere all'indirizzamento indiretto triplo: _____
6) Il file occupa _____ blocchi dati, dei quali:
   - i blocchi da _____ a _____ sono indirizzati dai puntatori diretti
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto semplice
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto doppio
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto triplo

   Pertanto per la rappresentazione del file sono necessari _____ blocchi indice.
7) Il file occupa _____ blocchi dati, dei quali:
   - i blocchi da _____ a _____ sono indirizzati dai puntatori diretti
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto semplice
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto doppio
   - i blocchi da _____ a _____ sono indirizzati attraverso indirizzamento indiretto triplo

   Pertanto per la rappresentazione del file sono necessari _____ blocchi indice.

Soluzione

1) Dato che ogni indirizzo occupa 4 byte, il numero di puntatori contenuti in un blocco indice è 2^10/4 = 2^8.
2) Il numero di blocchi indirizzabili con indirizzamento indiretto semplice è: 2^8= 256
3) Il numero di blocchi indirizzabili con indirizzamento indiretto doppio è: 2^16= 65.536
4) Il numero di blocchi indirizzabili con indirizzamento indiretto triplo è: 2^24= 16.777.216
5) Minima dimensione di un file affinché nella sua rappresentazione sia necessario ricorrere all'indirizzamento indiretto triplo: (10 + 256 + 65536) Kbyte + 1 Byte.

6) Si nota che non si deve ricorrere all'indirizzamento indiretto triplo. Il file occupa $\lceil 97.000 \textbf{ div } 1024 \rceil = 94$ blocchi, dei quali:
   - 10 sono indirizzati dai puntatori diretti (indici 0 .. 9 nello i-node),
   - 84 sono indirizzati attraverso un unico blocco indice (di primo livello), a sua volta indirizzato dal puntatore indiretto semplice (indice 10 nello i-node).

7) Si nota che non si deve ricorrere all'indirizzamento indiretto triplo. Il file occupa $\lceil 16 \text{ MByte } \textbf{div } 1024\text{Byte} \rceil = 16384$ blocchi, dei quali:
   - 10 sono indirizzati dai puntatori diretti (indici 0 .. 9 nello i-node),
   - 256 sono indirizzati attraverso un unico blocco indice (di primo livello), a sua volta indirizzato dal puntatore indiretto semplice (indice 10 nello i-node).
   - 16818 sono indirizzati attraverso $\lceil 16811 \textbf{ div } 256 \rceil = 63$ blocchi indice di primo livello, che sono indirizzati dai puntatori di indice 0,…,62 di un unico blocco indice di secondo livello, a sua volta indirizzato dal puntatore indiretto doppio (indice 11 nello i-node).

   Pertanto per la rappresentazione del file sono necessari in totale 65 blocchi indice (4 di primo livello e 1 di secondo livello)

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Esercizio 4 (3 punti)

In un sistema vengono generati i processi A,B,C,D, con i tempi di generazione e le durate (tempo di utilizzo del processore necessario per arrivare alla terminazione) sotto specificate:

Processo	Durata	Tempo di generazione
A	25	0
B	40	1
C	5	2
D	15	3
E	10	4

Tutti i tempi sono espressi in millisecondi

La politica di gestione del processore è la Round-Robin, con quanto di tempo pari a 10 msec e con tempo di commutazione di contesto trascurabile. Se un processo si sospende, alla riattivazione ottiene un intero quanto di tempo, indipendentemente dalla frazione consumata prima della sospensione.

E' inoltre definito il semaforo *sem*, con valore iniziale *0* e coda inizialmente vuota, sul quale vengono eseguite le seguenti operazioni:

- al tempo *12* il processo in esecuzione esegue *P*(*sem*) (***wait(sem)*** per gli A.A. antecedenti l'A.A. 2013/14)
- al tempo *28* il processo in esecuzione esegue *P*(*sem*) (***wait(sem)*** per gli A.A. antecedenti l'A.A. 2013/14)
- al tempo *36* il processo in esecuzione esegue *V*(*sem*) (***signal(sem)*** per gli A.A. antecedenti l'A.A. 2013/14)
- al tempo *40* il processo in esecuzione esegue *V*(*sem*) (***signal(sem)*** per gli A.A. antecedenti l'A.A. 2013/14)
- al tempo *50* il processo in esecuzione esegue *V*(*sem*) (***signal(sem)*** per gli A.A. antecedenti l'A.A. 2013/14)

Si chiede di compilare la tabella riportata nello schema di soluzione fino al tempo 50, riportando in ogni riga lo stato raggiunto dal sistema (processo in esecuzione, contenuto della Coda Pronti, valore di *sem*, contenuto della coda di *sem*) subito dopo aver gestito l'evento che si verifica al tempo indicato nella prima colonna.

Soluzione

(indicare tra parentesi tonde il tempo residuo di esecuzione del processo)

Tempo	Evento	Proc. in esecuzione	Coda Pronti	Valore di *sem*	Coda di *sem*
0	Generato A	A(25)	∅	0	∅

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Soluzione

(indicare tra parentesi tonde il tempo residuo di esecuzione del processo)

Tempo	Evento	Proc. in esecuzione	Coda Pronti	Valore di *sem*	Coda di *sem*
0	Generato A	A(25)	∅	0	∅
10	Scade QdiT	B(40)	C(5)→D(15)→E(10)→A(15)	0	∅
12	P(sem) / wait(sem)	C(5)	D(15)→E(10)→A(15)	0	B(38)
17	Ternina C	D(15)	E(10)→A(15)	0	B(38)
27	Scade QdiT	E(10)	A(15)→D(5)	0	B(38)
28	P(sem) / wait(sem)	A(15)	D(5)	0	B(38)→E(9)
36	V(sem) / signal(sem)	A(7)	D(5)→B(38)	0	E(9)
38	Scade QdiT	D(5)	B(38)→A(5)	0	E(9)
40	V(sem) / signal(sem)	D(3)	B(38)→A(5)→E(9)	0	∅
43	Termina D	B(38)	A(5)→E(9)	0	∅
50	V(sem) / signal(sem)	B(31)	A(5)→E(9)	1	∅

Esercizio 5 (3 punti)

Un sistema con processi A, B, C, D, E e risorse dei tipi R1, R2, R3, R4 ha raggiunto lo stato mostrato nelle tabelle seguenti:

Assegnazione attuale				
	R1	R2	R3	R4
A	2	1		1
B	1	2	1	
C		2		2
D			1	1
E	1		3	2

Esigenza residua (considerata la molteplicità e l'assegnazione attuale				
	R1	R2	R3	R4
A	1	0	1	2
B	2	1	1	3
C	2	1	2	1
D	1	0	1	1
E	0	1	0	1

Molteplicità			
R1	R2	R3	R4
4	5	5	6

Disponibilità			
0	0	0	0

Successivamente il processo A richiede un'istanza di R1, il processo B richiede un'istanza di R2, il processo C richiede un'istanza di R2, il processo D richiede un'istanza di R3 e il processo E richiede un'istanza di R4: di conseguenza tutti i processi si sospendono e il sistema è in stallo.
Per eliminare lo stallo si sopprimono i pocessi D ed E.
Si chiede:

1. lo stato raggiunto dopo l'eliminazione di D ed E,
2. lo stato raggiunto dopo aver soddisfatto (compatibilmente con la disponibilità) le richieste pendenti dei processi rimanenti;
3. a questo punto lo stallo è definitivamente eliminato? (ovvero, è sempre garantita la terminazione dei processi A, B e C ?)

Soluzione

1. Stato raggiunto dopo la soppressione dei processi D ed E:

Assegnazione attuale				
	R1	R2	R3	R4
A				
B				
C				
D				
E				

Esigenza residua (considerata la molteplicità e l'assegnazione attuale				
	R1	R2	R3	R4
A				
B				
C				
D				
E				

Molteplicità			
R1	R2	R3	R4
4	5	5	6

Disponibilità			

2. lo stato raggiunto dopo aver soddisfatto (compatibilmente con la disponibilità) le richieste pendenti dei processi rimanenti:

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Assegnazione attuale				
	R1	R2	R3	R4
A				
B				
C				
D				
E				

Esigenza residua (considerata la molteplicità e l'assegnazione attuale				
	R1	R2	R3	R4
A				
B				
C				
D				
E				

Molteplicità			
R1	R2	R3	R4
4	5	5	6

Disponibilità			

3. Verifica dello stato:

- Il processo _____ può terminare. La disponibilità di {R1, R2, R3, R4} diviene { _____ }

- Il processo _____ può terminare. La disponibilità di {R1, R2, R3, R4} diviene { _____ }

- Il processo _____ può terminare. La disponibilità di {R1, R2, R3, R4} diviene { _____ }

- Il processo _____ può terminare. La disponibilità di {R1, R2, R3, R4} diviene { _____ }

- Il processo _____ può terminare. La disponibilità di {R1, R2, R3, R4} diviene { _____ }

Lo stato è [sicuro/non sicuro]: _____

Soluzione

1. Stato raggiunto dopo la soppressione dei processi D ed E:

Assegnazione attuale				
	R1	R2	R3	R4
A	2	1		1
B	1	2	1	
C		2		2
D	-	-	-	-
E	-	-	-	-

Esigenza residua (considerata la molteplicità e l'assegnazione attuale				
	R1	R2	R3	R4
A	1	0	1	2
B	2	1	1	3
C	2	1	2	1
D	-	-	-	-
E	-	-	-	-

Molteplicità			
R1	R2	R3	R4
4	5	5	6

Disponibilità			
1	0	4	3

2. Stato raggiunto dopo l'assegnazione di un'istanza di R1 al processo A (le richieste pendenti di B e C non possono essere soddisfatte con l'attuale disponibilità):

Assegnazione attuale				
	R1	R2	R3	R4
A	3	1		1
B	1	2	1	
C		2		2
D	-	-	-	-
E	-	-	-	-

Esigenza residua (considerata la molteplicità e l'assegnazione attuale				
	R1	R2	R3	R4
A	0	0	1	2
B	2	1	1	3
C	2	1	2	1
D	-	-	-	-
E	-	-	-	-

Molteplicità			
R1	R2	R3	R4
4	5	5	6

Disponibilità			
0	0	4	3

3. A partire da questo stato la terminazione dei processi è sempre garantita. Infatti:

Il processo A può terminare
La disponibilità di {R1, R2, R3, R4} diviene { 3,1,4,4 }
- Il processo B può terminare
La disponibilità di {R1, R2, R3, R4} diviene { 4,3,5,4}
- Il processo C può terminare

In altri termini, lo stato è sicuro e lo stallo è definitivamente eliminato.

Sistemi Operativi e Laboratorio, Prova del 16/1/2015

Esercizio 6 (3 punti)

Un disco con 2 facce, 100 settori per traccia e 100 cilindri ha un tempo di seek proporzionale al numero di cilindri attraversati e pari a 1 ms per ogni cilindro. Il periodo di rotazione è di 10 msec: conseguentemente il tempo impiegato per percorrere un settore è di 0,1 msec. La politica di scheduling è la *shortest seek first.*

A un certo tempo (convenzionalmente indicato come t= 0) termina l'esecuzione dei comandi sul cilindro 44 e sono pendenti i seguenti comandi (tutti i tempi sono espressi in millisecondi):

tempo	cilindro	settore	faccia
0	91	12	0
0	87	1	0
0	12	9	1

Successivamente pervengono, ai tempi indicati, le seguenti richieste di lettura o scrittura:

tempo	cilindro	settore	faccia
126	86	21	1
130	44	25	0
137	91	90	0

Calcolare il tempo necessario per eseguire tutte queste operazioni. Per il ritardo rotazionale dopo un'operazione di *seek* si assume sempre il valore di caso peggiore, pari a un periodo di rotazione (10 msec). Nel caso in cui ci siano due richieste da servire sullo stesso cilindro queste vengono servite in ordine FIFO.

Il controllore è dotato di sufficiente capacità di buffering ed è sempre in grado di accettare senza ritardo i dati letti dal disco o quelli da scrivere sul disco.

Soluzione

op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	
op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	
op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	
op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	
op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	
op. su cilindro:		settore:							
inizio:		seek:		rotazione:		percorrenza:		fine:	

Soluzione

op. su cilindro:	12	settore:	9						
inizio:	0	seek:	32	rotazione:	10	percorrenza:	0,1	fine:	42,1
op. su cilindro:	87	settore:	1						
inizio:	42,1	seek:	75	rotazione:	10	percorrenza:	0,1	fine:	127,2
op. su cilindro:	86	settore:	21						
inizio:	127,2	seek:	1	rotazione:	10	percorrenza:	0,1	fine:	138,3
op. su cilindro:	91	settore:	12						
inizio:	138,3	seek:	5	rotazione:	10	percorrenza:	0,1	fine:	153,4
op. su cilindro:	91	settore:	90						
inizio:	153,4	seek:	0	rotazione:	7,7	percorrenza:	0,1	fine:	161,2
op. su cilindro:	44	settore:	25						
inizio:	161,2	seek:	47	rotazione:	10	percorrenza:	0,1	fine:	218,3

Esercizio 7 (2 punti)

Quali delle seguenti operazioni possono essere eseguite da un processo in stato utente?

Operazione:	**Eseguibili dai processi in stato utente [SI/NO] ?**
Eseguire l'istruzione IRET	
Salvare i registri generali nel proprio descrittore di processo	
Riattivare un processo sospeso	

Invocare una V() per gli studenti dell'A.A. antecedente al 2013/14: invocare una signal()	
Eseguire l'istruzione Test and Set Lock	
Invocare una yield()	

Soluzione

Operazione:	**Eseguibili dai processi in stato utente**
Eseguire l'istruzione IRET	NO
Salvare i registri generali nel proprio descrittore di processo	NO
Riattivare un processo sospeso	NO
Invocare una V() per gli studenti dell'A.A. antecedente al 2013/14: invocare una signal()	SI
Eseguire l'istruzione Test and Set Lock	SI
Invocare una yield()	SI

Esercizio 8 (2 punti)

Si consideri un sistema nel quale sono definiti i semafori *sem1* e *sem2* e i processi P1, P2, P3, P4 e P5.
Al tempo *T* i semafori hanno la seguente configurazione:
Sem1: valore 0, coda P4 → P5
Sem2: valore 2, coda Ø
Allo stesso tempo, la CodaPronti ha la configurazione P2 → P3 e il processo P1 è in esecuzione.
Lo scheduler dei processi non prevede il prerilascio del processore.
Indicare come si modificano i semafori e la CodaPronti e quale processo è in esecuzione se si verificano (in alternativa) le due seguenti sequenze di eventi:

Per gli studenti dell'A.A. 2013/14:

1) P1 esegue *P(Sem1)* e successivamente il processo in esecuzione esegue *P(Sem2);*
2) P1 esegue *V (Sem1)* e successivamente il processo in esecuzione esegue *V(Sem2);*

per gli studenti degli A.A. antecedenti al 2013/14:

1) P1 esegue *wait* (*Sem1*) e successivamente il processo in esecuzione esegue *wait(Sem2);*
2) P1 esegue *signal (Sem1)* e successivamente il processo in esecuzione esegue *signal(Sem2);*

Soluzione

	Sequenze di eventi	In Esecuzione	Coda Pronti	Sem1	Sem2
a-1	P1 esegue *P(Sem1)/wait (Sem1)*				
a-2	Il processo in esecuzione esegue *P(Sem2)/ wait(Sem2)*				
b-1	P1 esegue *V(Sem1)/ signal (Sem1)*				
b-2	Il processo in esecuzione esegue *V(Sem2)/ signal(Sem2)*				

Soluzione

	Sequenze di eventi	In Esecuzione	Coda Pronti	Sem1	Sem2
a-1	P1 esegue *P(Sem1)/wait (Sem1)*	P2	P3	0, P4→P5→P1	2, Ø
a-2	Il processo in esecuzione esegue *P(Sem2)/ wait(Sem2)*	P2	P3	0, P4→P5→P1	1, Ø

b-1	P1 esegue *V(Sem1)/ signal (Sem1)*	P1	P2→P3→P4	0, P5	2, Ø
b-2	Il processo in esecuzione esegue *V(Sem2)/ signal(Sem2)*	P1	P2→P3→P4	0, P5	3, Ø

Esercizio 9 (2 punti)

In un sistema che gestisce la memoria con segmentazione a domanda, l'indirizzo logico è di 28 bit, dei quali i primi 6 bit codificano l'indice di segmento e i restanti 22 bit definiscono l'offset. Ogni elemento della tabella dei segmenti è formato da 7 byte, dei quali 1 byte contiene indicatori utili al gestore della memoria, 3 byte contengono il valore base, e i restanti 3 byte contengono il valore limite.

Si chiede:

1. Il massimo numero di segmenti di un processo;
2. La massima dimensione di un segmento (in byte);
3. La massima dimensione della tabella dei segmenti (in byte);
4. Il massimo valore possibile per l'indirizzo di origine di un segmento;

Soluzione

1. Massimo numero di segmenti di un processo: _____ segmenti
2. Massima dimensione di un segmento (in byte): _____ byte
3. Massima dimensione della tabella dei segmenti (in byte): _____ byte
4. Massimo valore possibile per l'indirizzo di origine di un segmento: _____

Soluzione

5. Massimo numero di segmenti di un processo: 2^6 segmenti = 64 segmenti
6. Massima dimensione di un segmento (in byte): 2^{22} byte = 4 Mbyte
7. Massima dimensione della tabella dei segmenti (in byte): 64*7= 448 byte
8. Massimo valore possibile per l'indirizzo di origine di un segmento: 2^{24}-1