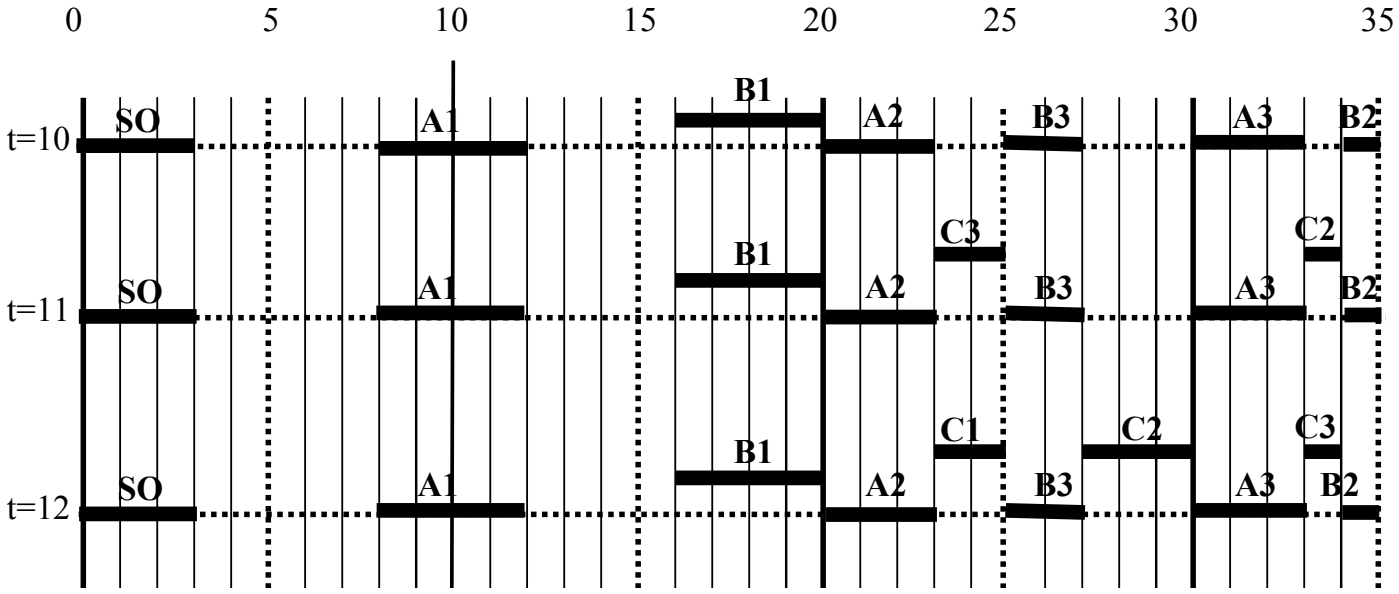
ESERCIZIO 1 (4 punti)

In un sistema simile a Unix che gestisce la memoria con segmentazione e caricamento in partizioni variabili, il sistema operativo occupa una partizione con origine 0 e lunghezza 3. Inoltre al tempo t=10 sono stati caricati in memoria i seguenti processi:

- il processo A, che occupa tre partizioni:
  - una partizione A1 con origine 8 e lunghezza 4 per il codice
  - una partizione A2 con origine 20 e lunghezza 3 per i dati
  - una partizione A3 con origine 30 e lunghezza 3 per lo stack
- il processo B, che occupa tre partizioni:
  - una partizione B1 con origine 16 e lunghezza 4 per il codice
  - una partizione B2 con origine 34 e lunghezza 1 per i dati
  - una partizione B3 con origine 25 e lunghezza 2 per lo stack

Tutte le lunghezze delle partizioni sono espresse in *Mbyte* e che la memoria ha dimensione 35, inoltre il gestore della memoria adotta politica *best fit* per l'assegnazione delle partizioni.
Al tempo t=11 il processo B esegue una fork e viene creato il processo C. Successivamente al tempo 12 il processo C esegue una exec per eseguire un nuovo codice. Si supponga che il nuovo codice occupi una partizione C1 di lunghezza 2,e che utilizzi inizialmente una partizione C2 di lunghezza 3 per i dati e una partizione C3 di lunghezza 1 per lo stack.
Utilizzando il grafico sotto riportato, mostrare come evolve l'occupazione della memoria ai tempi 11 e 12.

SOLUZIONE



ESERCIZIO 2 (4 punti)

In un sistema che gestisce la memoria con paginazione, sono presenti i processi A, B, C e D. Lo stato di occupazione della memoria è descritto dalla *Core Map*, dove per ogni blocco si specifica nell'ordine: il processo a cui appartiene la pagina caricata, l'indice della pagina caricata e il bit di pagina riferita.
Al tempo *t* la configurazione della *CoreMap* è quella riportata in figura, dove, per esempio, nel blocco 14 è caricata la pagina 1 del processo D con bit di pagina riferita uguale a 1.
L'algoritmo di sostituzione è il *Second Chance* (globale) e prima di ogni invocazione il puntatore è posizionato sul blocco successivo alla *vittima* individuata nell'invocazione precedente.

B,2 1	D,3 1	C,1 0	C,8 0	A,6 1	B,3 0	A,2 1	C,5 1	C,2 0	D,4 0	B,8 1	B,9 0	C,6 1	A,3 0	D,1 1	A,0 1	D,0 1	D,9 0	C,4 1
0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16	17	18

Mostrare come si modifica la *CoreMap* se al tempo *t* il puntatore è posizionato sul blocco 13 e si verifica la seguente sequenza di eventi:

1. tempo t+1: il processo D riferisce la pagina 9
2. tempo t+2: il processo A riferisce la pagina 5
3. tempo t+3: il processo B riferisce la pagina 1
4. tempo t+4: il processo C riferisce la pagina 7

SOLUZIONE

Tempo t+1:

B,2 1	D,3 1	C,1 0	C,8 0	A,6 1	B,3 0	A,2 1	C,5 1	C,2 0	D,4 0	B,8 1	B,9 0	C,6 1	A,3 0	D,1 1	A,0 1	D,0 1	D,9 1	C,4 1
0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16	17	18

Tempo t+2:

B,2 1	D,3 1	C,1 0	C,8 0	A,6 1	B,3 0	A,2 1	C,5 1	C,2 0	D,4 0	B,8 1	B,9 0	C,6 1	A,5 1	D,1 1	A,0 1	D,0 1	D,9 1	C,4 1
0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16	17	18

Tempo t+3:

B,2 0	D,3 0	B,1 1	C,8 0	A,6 1	B,3 0	A,2 1	C,5 1	C,2 0	D,4 0	B,8 1	B,9 0	C,6 1	A,5 1	D,1 0	A,0 0	D,0 0	D,9 0	C,4 0
0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16	17	18

Tempo t+4:

B,2 0	D,3 0	B,1 1	C,7 1	A,6 1	B,3 0	A,2 1	C,5 1	C,2 0	D,4 0	B,8 1	B,9 0	C,6 1	A,5 1	D,1 0	A,0 0	D,0 0	D,9 0	C,4 0
0	1	2	3	4	5	6	**7**	8	9	10	11	12	13	14	15	16	17	18

ESERCIZIO 3 (4 punti)

Si consideri un sistema che gestisce la memoria con paginazione dinamica con le seguenti caratteristiche:

- indirizzi logici di 32 bit e ampiezza dello spazio logico di ogni processo pari a 2^{32} byte;
- pagine logiche e blocchi fisici di 8 kByte
- tabelle delle pagine a due livelli; la tabella di primo livello comprende 2^{10} elementi, quelle di secondo libello 2^9 elementi;
- gli elementi della tabella di primo livello e di quelle di secondo livello hanno lunghezza pari a 24 bit, di cui 2 sono indicatori e i restanti 22 codificano l'indice di un blocco fisico.

In questo sistema è presente il processo P, con codice e dati di lunghezza rispettivamente uguale a 400 Mbyte e 200 Mbyte. L'area riservata per la pila è ampia 2 Mbyte. Codice e dati sono collocati nello spazio logico consecutivamente a partire dal suo estremo inferiore (indirizzo logico 0) e la pila a partire dall'estremo superiore (con espansione verso il basso).
Si chiede:

1. la lunghezza del campo offset, in numero di bit;
2. la lunghezza delle componenti dell'indirizzo logico che indicizzano, rispettivamente, la tabella di primo livello e quelle di secondo livello, in numero di bit;
3. lo spazio occupato in memoria dalla tabella di primo livello e da ogni tabella di secondo livello (in byte);
4. la massima dimensione della memoria fisica (numero di blocchi e di byte, espressi come potenze di 2);
5. il numero di tabelle di secondo livello necessarie per la traduzione degli indirizzi del processo P (considerando le regioni dello spazio logico nelle quali sono collocati il codice, i dati e la pila del processo);
6. nell'ipotesi che tutte le tabelle delle pagine necessarie per la traduzione degli indirizzi del processo P siano caricate in memoria, lo spazio (in byte) occupato complessivamente dalla tabella di primo livello e da quelle di secondo livello.

SOLUZIONE

1. lunghezza del campo offset : 13 bit;
2. lunghezza della componenti dell'indirizzo logico che indicizzano, rispettivamente, la tabella di primo livello e quelle di secondo livello: rispettivamente 10 e 9 bit;
3. spazio occupato in memoria dalla tabella di primo livello: $3 * 2^{10}$ byte= 3 Kbyte;

   spazio occupato in memoria da ogni tabella di secondo livello: $3 * 2^9$ byte = 1,5 Kbyte;
4. massima dimensione della memoria fisica : 2^{22} blocchi; $2^{22} * 2^{13} = 2^{35}$ byte (32 Gbyte);
5. numero di tabelle di secondo livello necessarie per la traduzione degli indirizzi del processo P:

   - per il codice e i dati, che occupano i 600 Mbyte iniziali dello spazio logico: $600 * 2^{20}/2^{13} = 150 * 2^9$ pagine;

   $150 * 2^9/2^9 =$ 150 tabelle

   - per la pila, che occupa i 2 Mbyte finali dello spazio logico: $2 * 2^{20}/2^{13} = 2^8$ pagine; 1 tabella
6. spazio (in byte) occupato complessivamente dalla tabella di primo livello e da quelle di secondo livello:

   $3 * 2^{10} + 151 * 3 * 2^9$ byte= 229,5 Kbyte.

ESERCIZIO 4 (4 punti)

Un disco con 2 facce, 200 settori per traccia e 100 cilindri ha un tempo di seek proporzionale al numero di cilindri attraversati e pari a 0,2ms per ogni cilindro. Il periodo di rotazione è di 20 msec: conseguentemente il tempo impiegato per percorrere un settore è di 0,1 msec.
A un certo tempo (convenzionalmente indicato come t= 0) termina l'esecuzione dei comandi sul cilindro 61 e sono pervenute, nell'ordine, le seguenti richieste di lettura o scrittura:

- clindro 59, faccia 0, settore 100
- clindro 69, faccia 1, settore 120
- clindro 77, faccia 1, settore 5
- clindro 98, faccia 0, settore 55

Al tempo 42 arrivano due ulteriori comandi di lettura:

- clindro 59, faccia 0, settore 5
- clindro 82, faccia 0, settore 60

Calcolare il tempo necessario per eseguire tutte queste operazioni supponendo che si adotti la politica di scheduling SCAN con fase di salita attiva al tempo 0.
Il tempo di esecuzione di ogni operazione è uguale alla somma dell'eventuale tempo di *seek,* del ritardo rotazionale (tempo necessario per raggiungere il settore indirizzato) e del tempo di percorrenza del settore indirizzato.
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.
Per il ritardo rotazionale dopo un'operazione di *seek* si assume sempre il valore di caso peggiore, pari a un intero periodo di rotazione (20 msec). Si assuma inoltre che i comandi sullo stesso cilindro vengano eseguiti in ordine FIFO.

SOLUZIONE

op.su cilindro:	69	settore:	120						
inizio:	0	seek:	1,6	rotazione:	20	percorrenza:	0,1	fine:	21,7
op.su cilindro:	77	settore:	5						
inizio:	21,7	seek:	1,6	rotazione:	20	percorrenza:	0,1	fine:	43,4
op.su cilindro:	82	settore:	60						
inizio:	43,4	seek:	1	rotazione:	20	percorrenza:	0,1	fine:	64,5
op.su cilindro:	98	settore:	55	faccia1					
inizio:	64,5	seek:	3,2	rotazione:	20	percorrenza:	0,1	fine:	87,8
op.su cilindro:	59	settore:	100	faccia0					
inizio:	87,8	seek:	7,8	rotazione:	20	percorrenza:	0,1	fine:	115,7
op.su cilindro:	59	settore:	5						
inizio:	115,7	seek:	0	rotazione:	10,4	percorrenza:	0,1	fine:	126,2

ESERCIZIO 5 (4 punti)

In un file system UNIX i blocchi del disco hanno ampiezza di 1Kbyte e gli i-node,che occupano 1 blocco, contengono 10 indirizzi diretti e 3 indirizzi indiretti. Si consideri il file individuato dal pathname *SistemiOperativi*/*appunti*/*esercizi,* relativo alla directory corrente, e si facciano le seguenti ipotesi:

- la directory corrente è caricata in memoria
- le directory *SistemiOperativi* e *appunti* (ciascuna delle quali occupa 1 blocco) e il file *esercizi* risiedono su disco.
- gli i-node delle directory *SistemiOperativi* e *appunti* e quello del file *esercizi* risiedono su disco, rispettivamente nei blocchi 12, 15 e 88.
- lo i-node della directory *SistemiOperativi* contiene un solo indirizzo diretto, con valore *27*
- analogamente, quello della directory *appunti* contiene un solo indirizzo diretto, con valore *48.*
- gli indirizzi contenuti nello i-node del file *terzo appello* hanno i valori:

Indirizzo	1	2	3	4	5	6	7	8	9	10	11	12	13
Valore	51	52	53	70	72	73	75	100	101	76	96	∅	∅

- gli indirizzi contenuti nel blocco indiretto semplice hanno i valori:

Indirizzo	1	2	3	4	5	6	7	8	9	10	……..
Valore	42	64	65	66	78	90	91	93	75	∅	

- il valore corrente del puntatore di lettura è 8716 e la lunghezza corrente del file è di 18632 byte.

In questa situazione, il proprietario della directory *SistemiOperativi*, che ha i necessari diritti, esegue una chiamata di sistema per leggere 9000 caratteri dal file *esercizi.* Si chiede:

1. quanti e quali blocchi del disco vengono letti per completare l'esecuzione;
2. quanti caratteri vengono estratti da ciascun blocco dati del file *esercizi* che viene letto per eseguire l'operazione.

SOLUZIONE

Il puntatore di lettura è posizionato sul carattere 8716 *mod* 1024= 524 del blocco 9176 *div* 1024= 8, indirizzato dall'indirizzo diretto 9.

L'ultimo carattere da leggere ha indice 8716+ 9000= 17716 e corrisponde al carattere 308 del blocco 17, individuato dall'indirizzo 8 del blocco indiretto semplice. Pertanto:

1. Per completare l'operazione vengono letti i seguenti blocchi:
   - blocco 12 , che contiene lo i-node di *SistemiOperativi*
   - blocco 27 , che contiene la directory *SistemiOperativi*
   - blocco 15 , che contiene lo i-node di *appunti*
   - blocco 48, che contiene la directory *appunti*
   - blocco 88 , che contiene lo i-node di *esercizi*
   - blocchi 101 e 76, indirizzati dagli indirizzi diretti 9 e 10
   - blocco 96 , che contiene il blocco indiretto semplice
   - blocchi 42, 64, 65, 66, 78, 90, 91, 93 indirizzati dai primi 8 indirizzi del blocco indiretto semplice.
2. Caratteri estratti dai blocchi dati del file *esercizi* che sono interessati alla lettura:
   - 500 caratteri dal blocco 101
   - 1024 caratteri dal blocco 76
   - 1024 caratteri da ciascuno del blocchi 42, 64, 65, 66, 78, 90, 91
   - 308 caratteri dal blocco 95

ESERCIZIO 6 (2 punti)

In un sistema che gestisce la memoria con segmentazione (con spazio logico suddiviso in segmento codice, segmento dati e pila) e e caricamento in partizioni variabili con rilocazione dinamica, i registri base e limite del segmento codice, dati e stack hanno a un certo istante i valori B_0= 7.400 e L_0= 38.019, B_1= 110.100 e L_1= 8.022 e B_2= 75.000 e L_2= 13.250, rispettivamente.
Supponendo che il processo in esecuzione riferisca i seguenti indirizzi logici (formati da coppie contenenti l'indice di segmento e l'offset): <3,2.720>, <0,27.200>, <0,38.019>, <2,7.999>, <1,6.000>, <1,9.980> dire se ognuno dei precedenti indirizzi logici è legittimo e, in caso affermativo, calcolare il corrispondente indirizzo fisico.

SOLUZIONE

Indirizzo logico	Legittimo?	Indirizzo fisico
1. <3,2.720>	NO	
2. <0,27.200>	SI	34.600
3. <0,38.019>	NO	
4. <2,7.999>	SI	82.999
5. <1,6.000>	SI	116.100
6. <1,9.980>	NO	

ESERCIZIO 7 (2 punti)

In un disco RAID livello 4 il disco virtuale V ha la capacità di 2^{12} blocchi di 1 kByte , numerati da 0 a $2^{12}-1$, ed è realizzato mediante 5 dischi fisici, D(0), D(1), D(2), D(3) e D(4) ciascuno della capacità di 2^{10} blocchi di 1 kByte, numerati da 0 a $2^{10}-1$. Il blocco *b* del disco V è mappato nel blocco *b* ***div*** *4* del disco fisico di indice *b* ***mod*** *4*.
Per ciascuna delle seguenti operazioni, ciascuna delle quali interessa più blocchi consecutivi del disco virtuale:

1) lettura dei blocchi 2019, 2020, 2021,2022,2023
2) lettura dei blocchi 268, 269, 270, 271

si determini quali dischi fisici e quali blocchi sono interessati all'operazione e quanto tempo (espresso come massimo numero di accessi per disco) è necessario per completare l'operazione.

SOLUZIONE

a) Lettura dei blocchi virtuali 2019, 2020, 2021,2022,2023

   dischi fisici e i relativi blocchi fisici interessati:

   disco 3, blocco 504 ;disco 0, blocco 505 ; disco 1, blocco 505 ; disco 2, blocco 505 ; disco 3, blocco 505 ;

   tempo di esecuzione (unità di misura: tempo di accesso): 2

b) Lettura dei blocchi virtuali 268, 269, 270, 271

   dischi fisici e i relativi blocchi fisici interessati:

   disco 0, blocco 67 ; disco 1 blocco 67; disco 2, blocco 67; disco 3, blocco 67;

   tempo di esecuzione (unità di misura: tempo di accesso): 1

ESERCIZIO 8 (2 punti)

In un disco con blocchi di 4 Kbyte (= 2^{12} byte), è definito un file system FAT. Gli elementi della FAT sono in corrispondenza biunivoca con i blocchi fisici del disco. Ogni elemento contiene l'indice di un blocco del disco, codificato con 24 bit.
Il primo blocco di ogni file è identificato dalla coppia (*nomefile, indiceblocco)* contenuta nella rispettiva directory. Gli elementi successivi sono identificati da una lista concatenata di indici di blocchi, realizzata sulla FAT.

1. Qual è la massima capacità del disco, espressa in blocchi e in byte?
2. Quanti byte occupa la FAT?
3. Supponendo che la memoria sia gestita con paginazione dinamica, con pagine logiche e blocchi fisici di 4Kbyte, e che che l'origine della FAT sia allineata con l'origine di una pagina, quante pagine occupa complessivamente la FAT? (Nota: la gestione con paginazione interessa anche la FAT)
4. Se si esegue una lettura da un file, il cui puntatore di lettura è posizionato all'interno del decimo blocco del file, quanti errori di pagina si verificano, nel caso peggiore, per conoscere la posizione sul disco (espressa come indice di blocco) del blocco che contiene il primo carattere da leggere?

SOLUZIONE

1. Capacità del disco: 2^{24} blocchi; 2^{36} byte (64 Gbyte)
2. Numero complessivo di byte occupati dalla FAT: $3 * 2^{24} = 48 * 2^{20}$ byte (48 Mbyte)
3. Numero di pagine occupate dalla FAT: $48 * 2^{20}/2^{12} = 12 * 2^{10}$ pagine
4. Per conoscere la posizione sul disco del blocco che contiene il primo carattere si deve raggiungere il 9° elemento della FAT, che contiene l'indice del 10° blocco del file. Si devono quindi percorrere 9 elementi della FAT, che nel caso peggiore appartengono a pagine distinte, nessuna delle quali è presente in memoria: quindi si possono verificare 9 errori di pagina.

ESERCIZIO 9 (2 punti)

In un file system UNIX i blocchi del disco hanno ampiezza di 1Kbyte e gli i-node contengono 10 indirizzi diretti e 2 indirizzi indiretti. Tutti gli indirizzi hanno una lunghezza di 4 byte.
Si chiede:

1. la massima capacità del disco che ospita il file system, in blocchi e in byte
2. la massima dimensione dei file indirizzabili dall'i-node, in blocchi e in byte.

SOLUZIONE

1. Massima capacità del disco che ospita il file system: 2^{32} blocchi; $2^{32} * 2^{10} = 2^{42}$ blocchi (4 Tbyte)
2. considerato che:
   - lo i-node indirizza direttamente 10 blocchi
   - il blocco indiretto semplice indirizza $2^{10} / 4 = 256$ blocchi dati
   - il blocco indiretto doppio indirizza $2^{10} / 4 = 256$ blocchi indiretti di secondo livello, ciascuno dei quali indirizza 256 blocchi dati,

la massima dimensione dei file indirizzabili dall'i-node è pari a $10 + 256 + 256^2 = 65.802$ blocchi, ovvero $65.802 * 2^{10} = 67.381.248$ byte (circa 64 Mbyte)

ESERCIZIO 10 (2 punti)

In un file system UNIX si consideri il file */usr/tizio /esercitazioni/progetto1.exe,* creato dall'utente *tizio*.
I diritti associati alle directory *usr, tizio, esercitazioni* e al file *progetto1.exe* sono i seguenti

	owner (r w x)	*group (r w x)*	*others (r w x)*
usr	*1 0 1*	*1 0 1*	*1 0 1*
tizio	*1 1 1*	*1 0 1*	*1 0 1*
esercitazioni	*1 1 1*	*1 0 1*	*1 0 0*
progetto1.exe	*1 1 1*	*1 0 1*	*0 0 1*

Quali operazioni possono eseguite sulla directory *esercitazioni* e sul file *progetto1.exe* dall'utente *caio* nelle seguenti ipotesi:

1. *tizio* e *caio* appartengono allo stesso gruppo;
2. *tizio* e *caio* appartengono a gruppi diversi;

SOLUZIONE

IPOTESI	OPERAZIONI		
caio e *tizio* appartengono allo stesso gruppo	Listing di esercitazioni: SI	Modifica di *esercitazioni*: NO	Seguire i link di *esercitazioni*: SI
	Listing di *progetto1.exe:* SI	Modifica di *progetto1.exe*: NO	Esecuzione di *progetto1.exe*: SI
caio e *tizio* appartengono a gruppi diversi	Listing di esercitazioni: SI	Modifica di *esercitazioni*: NO	Seguire i link di *esercitazioni*: NO
	Listing di *progetto1.exe:* NO	Modifica di *progetto1.exe*: NO	Esecuzione di *progetto1.exe*: NO