Sistemi Operativi

ESERCITAZIONE

GESTIONE DELLA MEMORIA

Esercizio 1 (vale 3 punti)

In un sistema con gestione della memoria principale a partizioni, dire come sono utilizzate le seguenti partizioni di memoria di grandezza 100KB, 500KB, 200KB, 300KB e 600KB (nell'ordine), da ciascuno degli algoritmi First-fit, Best-fit e Worst-fit per allocare i seguenti processi di grandezza 212KB, 417KB, 112KB e 426KB (nell'ordine).

Quale algoritmo usa la memoria in maniera più efficiente?

Soluzione

- La traccia dell'esercizio non precisa a quale tipo di gestione a partizioni è soggetta la memoria principale
- Escludendo la gestione a partizioni multiple, poiché non c'è alcun riferimento al fatto che lo spazio virtuale possa essere segmentato, restano le seguenti due possibilità:
  - Partizioni fisse: al momento in cui una partizione viene assegnata, in generale, resterà dello spazio inutilizzato interno alla partizione; quindi, si ha frammentazione interna
  - Partizioni variabili: al momento in cui una partizione viene assegnata, lo spazio inutilizzato interno alla partizione andrà a formare una nuova partizione libera; quindi, non si ha frammentazione interna, semmai frammentazione esterna

Partizioni fisse: First-fit

1. il processo di 212KB è posto nella partizione di 500KB
2. il processo di 417KB è posto nella partizione di 600KB
3. il processo di 112KB è posto nella partizione di 200KB
4. il processo di 426KB deve aspettare

Dati

- Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
- Processi: 212KB, 417KB, 112KB e 426KB

Partizioni fisse: Best-fit

1. il processo di 212KB è posto nella partizione di 300KB
2. il processo di 417KB è posto nella partizione di 500KB
3. il processo di 112KB è posto nella partizione di 200KB
4. il processo di 426KB è posto nella partizione di 600KB

Dati

- Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
- Processi: 212KB, 417KB, 112KB e 426KB

Partizioni fisse: Worst-fit

1. il processo di 212KB è posto nella partizione di 600KB
2. il processo di 417KB è posto nella partizione di 500KB
3. il processo di 112KB è posto nella partizione di 300KB
4. il processo di 426KB deve aspettare

Dati

- Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
- Processi: 212KB, 417KB, 112KB e 426KB

Partizioni variabili: First-fit

1. il processo di 212KB è posto nella partizione di 500KB; come residuo viene creata una nuova partizione di 500KB-212KB=288KB e la lista delle partizioni disponibili è ora 100KB, 288KB, 200KB, 300KB e 600KB (nell'ordine)
2. il processo di 417KB è posto nella partizione di 600KB; come residuo viene creata una nuova partizione di 183KB e la lista delle partizioni disponibili è ora 100KB, 288KB, 200KB, 300KB e 183KB (nell'ordine)

Dati

- Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
- Processi: 212KB, 417KB, 112KB e 426KB

Partizioni variabili: First-fit

3. il processo di 112KB è posto nella partizione di 288KB creata come residuo quando è stato allocato il processo di 212KB; come residuo viene creata una ulteriore partizione di 176KB e la lista delle partizioni disponibili è ora 100KB, 176KB, 200KB, 300KB e 183KB (nell'ordine)
4. il processo di 426KB deve aspettare

Dati

- Partizioni: 100KB, 288KB, 200KB, 300KB e 183KB
- Processi: 112KB e 426KB

Partizioni variabili: Best-fit

1. il processo di 212KB è posto nella partizione di 300KB; come residuo viene creata una nuova partizione di 88KB e lista partizioni 100KB, 500KB, 200KB, 88KB e 600KB
2. il processo di 417KB è posto nella partizione di 500KB; come residuo viene creata una nuova partizione di 83KB e lista partizioni 100KB, 83KB, 200KB, 88KB e 600KB
3. il processo di 112KB è posto nella partizione di 200KB; come residuo viene creata una nuova partizione di 88KB e lista partizioni 100KB, 83KB, 88KB, 88KB e 600KB
4. il processo di 426KB è posto nella partizione di 600KB; come residuo viene creata una nuova partizione di 174KB e lista partizioni 100KB, 83KB, 200KB, 88KB e 174KB

Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
Processi: 212KB, 417KB, 112KB e 426KB

Partizioni variabili: Worst-fit

1. il processo di 212KB è posto nella partizione di 600KB; come residuo viene creata una nuova partizione di 388KB e lista partizioni 100KB, 500KB, 200KB, 300KB e 388KB
2. il processo di 417KB è posto nella partizione di 500KB; come residuo viene creata una nuova partizione di 83KB e lista partizioni 100KB, 83KB, 200KB, 300KB e 388KB
3. il processo di 112KB è posto nella partizione di 388KB; come residuo viene creata una ulteriore partizione di 276KB e lista partizioni 100KB, 83KB, 200KB, 300KB e 276KB
4. il processo di 426KB deve aspettare

Partizioni: 100KB, 500KB, 200KB, 300KB e 600KB
Processi: 212KB, 417KB, 112KB e 426KB

Soluzione

Quale algoritmo usa la memoria in maniera più efficiente?

- In questo esempio, Best-fit risulta essere l'algoritmo migliore, sia che le partizioni siano fisse sia che siano variabili, perché riesce ad allocare tutti i processi (il prezzo da pagare è dato dalla ricerca della partizione più appropriata, come del resto accade pure con Worst-fit)
- First-fit e Worst-fit hanno un maggiore spreco di memoria e infatti in entrambi i casi l'ultimo processo è costretto ad attendere che si liberi una partizione più grande di quelle presenti per poter essere servito
  - Worst-fit crea residui più grandi e, quindi, più facilmente utilizzabili per allocare altri processi in caso di gestione con partizioni variabili
  - Worst-fit è utile solo nel caso di partizioni variabili

Esercizio 2 (vale 3 punti)

Si consideri la seguente tabella dei segmenti.

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Quali sono gli indirizzi fisici che corrispondono ai seguenti indirizzi logici?

(1) 0,430
(2) 1,10
(3) 2,100
(4) 3,400
(5) 4,112

Soluzione

1) 0,430: poiché 430 < 600 (valore limite), la traduzione si può effettuare e l'indirizzo è 219 + 430 = 649

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Soluzione

1) 0,430: poiché 430 < 600 (valore limite), la traduzione si può effettuare e l'indirizzo è 219 + 430 = 649
2) 1,10: poiché 10 < 14 (valore limite), la traduzione si può effettuare e l'indirizzo è 2300 + 10 = 2310

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Soluzione

1) 0,430: poiché 430 < 600 (valore limite), la traduzione si può effettuare e l'indirizzo è 219 + 430 = 649
2) 1,10: poiché 10 < 14 (valore limite), la traduzione si può effettuare e l'indirizzo è 2300 + 10 = 2310
3) 2,100: poiché 100 ≮ 100 (valore limite), il riferimento è illegale e viene generato un trap al SO (la traduzione non si può effettuare)

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

Soluzione

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

4) 3,400: poiché 400 < 580 (valore limite), la traduzione si può effettuare e l'indirizzo è 1327 + 400 = 1727

Soluzione

segmento	base	limite
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

4) 3,400: poiché 400 < 580 (valore limite), la traduzione si può effettuare e l'indirizzo è 1327 + 400 = 1727

5) 4,112: poiché 112 ≮ 96 (valore limite), il riferimento è illegale e viene generato un trap al SO (la traduzione non si può effettuare)

Soluzione

1) 0,430: poiché 430 < 600 (valore limite), la traduzione si può effettuare e l'indirizzo è 219 + 430 = 649
2) 1,10: poiché 10 < 14 (valore limite), la traduzione si può effettuare e l'indirizzo è 2300 + 10 = 2310
3) 2,100: poiché 100 ≮ 100 (valore limite), il riferimento è illegale e viene generato un trap al SO (la traduzione non si può effettuare)
4) 3,400: poiché 400 < 580 (valore limite), la traduzione si può effettuare e l'indirizzo è 1327 + 400 = 1727
5) 4,112: poiché 112 ≮ 96 (valore limite), il riferimento è illegale e viene generato un trap al SO (la traduzione non si può effettuare)

Esercizio 3 (vale 4 punti)

Sapendo che un SO implementa uno schema di gestione della memoria con paginazione a domanda, con pagine da 2K parole (unità indirizzabili), si calcoli il numero di page fault causati da un processo che genera i seguenti indirizzi logici:

10260, 4140, 8200, 2070, 10264, 4140, 12300, 10265, 14400, 12300, 10266, 12304, 4140

nei casi in cui si applicano gli algoritmi di sostituzione delle pagine FIFO ed LRU e nell'ipotesi che al processo siano assegnati 4 frame.

Soluzione

10260, 4140, 8200, 2070, 10264, 4140, 12300,
10265, 14400, 12300, 10266, 12304, 4140

Innanzitutto, trasformiamo la sequenza di indirizzi logici in una sequenza di numeri di pagine virtuali

- Allo scopo di determinare il numero di page fault, non è importante lo specifico indirizzo da tradurre, ma la pagina virtuale a cui appartiene

Soluzione

10260, 4140, 8200, 2070, 10264, 4140, 12300,
10265, 14400, 12300, 10266, 12304, 4140

Dal momento che ogni pagina virtuale contiene 2048 parole, la sequenza di indirizzi logici corrisponde alla seguente sequenza di riferimenti a pagine virtuali:

5, 2, 4, 1, 5, 2, 6, 5, 7, 6, 5, 6, 2

ottenuti

- prendendo il quoziente intero della divisione di ciascun indirizzo logico per la dimensione della pagina
- o, più semplicemente, confrontando ciascun indirizzo logico con alcuni multipli della dimensione della pagina, e cioè

0, 2048, 4096, 6144, 8192, 10240, 12288, 14336, 16384

0 1 2 3 4 5 6 7

Soluzione: FIFO

5, 2, 4, 1, 5, 2, 6, 5, 7, 6, 5, 6, 2

Applicando l'algoritmo FIFO, con 4 frame abbiamo:

5	2	4	1	5	2	6	5	7	6	5	6	2
5	2	4	1	1								
	5	2	4	4								
		5	2	2								
			5	5								
*	*	*	*									

Per prevenire errori, conviene mantenere le pagine caricate in memoria in un dato momento nell'ordine FIFO con cui sono entrate: in alto quella entrata per ultima, in basso quella entrata per prima e quindi 'vittima' da sostituire in caso di page fault

Soluzione: FIFO

5, 2, 4, 1, 5, 2, 6, 5, 7, 6, 5, 6, 2

Applicando l'algoritmo FIFO, con 4 frame abbiamo:

5	2	4	1	5	2	6	5	7	6	5	6	2
5	2	4	1	1	1	6	5	7	7	7	7	2
	5	2	4	4	4	1	6	5	5	5	5	7
		5	2	2	2	4	1	6	6	6	6	5
			5	5	5	2	4	1	1	1	1	6
*	*	*	*			*	*	*				*

Pertanto, il numero totale di page fault risultante è 8

Soluzione: LRU

5, 2, 4, 1, 5, 2, 6, 5, 7, 6, 5, 6, 2

Applicando l'algoritmo LRU, con 4 frame abbiamo:

5	2	4	1	5	2	6	5	7	6	5	6	2
5	2	4	1	5	2							
	5	2	4	1	5							
		5	2	4	1							
			5	2	4							
*	*	*	*									

Conviene mantenere le pagine caricate in memoria in un dato momento nell'ordine dei loro più recenti riferimenti: in alto quella riferita più di recente, in basso quella riferita meno di recente e quindi 'vittima' da sostituire in caso di page fault

Soluzione: LRU

5, 2, 4, 1, 5, 2, 6, 5, 7, 6, 5, 6, 2

Applicando l'algoritmo LRU, con 4 frame abbiamo:

5	2	4	1	5	2	6	5	7	6	5	6	2
5	2	4	1	5	2	6	5	7	6	5	6	2
	5	2	4	1	5	2	6	5	7	6	5	6
		5	2	4	1	5	2	6	5	7	7	5
			5	2	4	1	1	2	2	2	2	7
*	*	*	*			*		*				

Pertanto, il numero totale di page fault risultante è 6

Esercizio 4 (vale 4 punti)

Si consideri un sistema che implementa uno schema di gestione della memoria a paginazione con indirizzi virtuali a 30 bit, indirizzi fisici a 20 bit, pagine di 4KB e descrittori di pagina di 16 byte:

a) qual è la struttura dell'indirizzo virtuale e dell'indirizzo fisico?
b) di quante pagine e byte sono costituiti rispettivamente lo spazio di indirizzamento virtuale e quello fisico?
c) qual è la struttura e quali sono le dimensioni complessive e delle singole tabelle in una organizzazione con tabella delle pagine a 2 livelli in cui i 10 bit più significativi dell'indirizzo corrispondono al livello più alto?
d) quante pagine occupa un programma il cui codice occupa 21000 byte? Quanta memoria viene sprecata a causa della frammentazione interna?

Soluzione (a)

a) qual è la struttura dell'indirizzo virtuale e dell'indirizzo fisico?

L'offset dipende solo dalla dimensione delle pagine. Dato che le pagine sono di 4KB = 2^{12} byte, l'offset è costituito da 12 bit. Quindi abbiamo:

- struttura dell'indirizzo virtuale: 18 bit (i più significativi) per indirizzare le pagine virtuali, 12 bit per l'offset;
- struttura dell'indirizzo fisico: 8 bit (i più significativi) per indirizzare le pagine fisiche, 12 bit per l'offset.

Dati:

- indirizzi virtuali a 30 bit
- indirizzi fisici a 20 bit
- pagine di 4KB
- descrittori di pagina di 16 byte

Soluzione (b)

b) di quante pagine e byte sono costituiti rispettivamente lo spazio di indirizzamento virtuale e quello fisico?

- Data la struttura dell'indirizzo virtuale, il numero delle pagine virtuali è 2^{18} = 256K, mentre la dimensione complessiva dello spazio virtuale è di 2^{30} byte, cioè 1GB.
- Data la struttura dell'indirizzo fisico, il numero delle pagine fisiche è 2^{8} = 256, mentre la dimensione complessiva dello spazio fisico è di 2^{20} byte, cioè 1MB.

Dati:
- indirizzi virtuali a 30 bit
- indirizzi fisici a 20 bit
- pagine di 4KB
- descrittori di pagina di 16 byte

Soluzione (c)

c) qual è la struttura e quali sono le dimensioni complessive e delle singole tabelle in una organizzazione con tabella delle pagine a 2 livelli in cui i 10 bit più significativi dell'indirizzo corrispondono al livello più alto?

Una tabella delle pagina a 2 livelli è costituita da:

- Una tabella di primo livello in cui viene usato come indice il primo gruppo di bit
  - In questo caso, i 10 bit più significativi, pertanto la tabella ha 2^{10} elementi
- Una tabella di secondo livello in corrispondenza a ciascun elemento della tabella di primo livello
  - Nel caso dell'esercizio avremo 2^{10} tabelle, ciascuna con 2^{8} elementi, poiché viene usato come indice il secondo gruppo di bit, cioè i restanti 8 bit meno significativi dei 18 dedicati ad indicizzare le pagine virtuali

Soluzione (c)

c) qual è la struttura e quali sono le dimensioni complessive e delle singole tabelle in una organizzazione con tabella delle pagine a 2 livelli in cui i 10 bit più significativi dell'indirizzo corrispondono al livello più alto?

- Ciascun elemento della tabella di primo livello è un puntatore alla corrispondente tabella di secondo livello e deve poter contenere perlomeno un indirizzo virtuale (30 bit)
  - quindi 4 = 2^2 byte (dev'essere un multiplo del byte)
- La tabella di primo livello ha quindi 2^{10} elementi di 2^2 byte ciascuno ed occupa complessivamente 2^{10} x 2^2 byte = 2^{12} byte = 4KB (= 1 pagina)

Soluzione (c)

c) qual è la struttura e quali sono le dimensioni complessive e delle singole tabelle in una organizzazione con tabella delle pagine a 2 livelli in cui i 10 bit più significativi dell'indirizzo corrispondono al livello più alto?

- Ciascun elemento di una tabella di secondo livello è un descrittore di pagina e quindi è formato da 16 = 2^4 byte
- Ogni tabella di secondo livello ha quindi 2^8 elementi di 2^4 byte ciascuno, quindi occupa complessivamente $2^8 \cdot 2^4$ byte = 2^{12} byte = 4KB (= 1 pagina)
- Dato che le tabelle di secondo livello sono 2^{10}, esse occupano complessivamente $2^{10} \cdot 2^{12}$ byte = 2^{22} byte = 4MB (= 1024 pagine!)

Soluzione (d)

d) quante pagine occupa un programma il cui codice occupa 21000 byte? Quanta memoria viene sprecata a causa della frammentazione interna?

- Il programma viene allocato in pagine da 4KB, perciò il numero delle pagine da esso occupate è [21000/4096] = 6 (cioè il più piccolo intero maggiore o uguale al quoziente tra la dimensione del programma e quella delle pagine)
- A causa della frammentazione interna l'ultima pagina è occupata solo parzialmente e c'è uno spreco pari a (6·4096)–21000 = 24576–21000 = 3576 byte

Esercizio 5 (vale 4 punti)

Si consideri un sistema che usa uno schema di gestione della memoria principale con paginazione a domanda in cui il più grande indice utilizzabile nella tabella delle pagine di un processo è 3FFF (esadecimale). Un indirizzo fisico del sistema è scritto su 24 bit di cui 13 sono dedicati per esprimere il numero di frame.

a) Quanto è grande, in MB, lo spazio di indirizzamento logico del sistema? (motivare le risposta esplicitando i calcoli effettuati)

b) Quali informazioni conterrà (necessariamente) ciascun descrittore di pagina di questo sistema se viene usato l'algoritmo di sostituzione con seconda chance?

c) Nel caso non si verifichino mai page fault, qual è, in nanosecondi, il tempo effettivo di accesso in memoria del sistema se viene usato un TLB con un tempo di accesso di 5 nanosecondi, un hit-ratio del 95% e un tempo di accesso in RAM di 0,08 microsecondi? (È sufficiente riportare l'espressione aritmetica che fornisce il risultato finale)

Soluzione (a)

a) Quanto è grande, in megabyte, lo spazio di indirizzamento logico del sistema? (motivare le risposta esplicitando i calcoli effettuati)

- Poiché l'indice di pagina più grande è 3FFF, che in binario necessita di 14 bit per essere espresso, ci possono essere al massimo 2^{14} pagine
- Inoltre, poiché un indirizzo fisico è di 24 bit di cui 13 sono dedicati per il numero di frame, la dimensione di un frame, e quindi di una pagina, è di 2^{11} byte (24 - 13 = 11)
- Quindi, lo spazio di indirizzamento logico è di $2^{14} \times 2^{11}$ byte = 2^{25} byte = 32MB

Dati:
- più grande indice di pagina 3FFF
- indirizzo fisico scritto su 24 bit
- numero di frame scritto su 13 bit

Soluzione (b)

b) Quali informazioni conterrà (necessariamente) ciascun descrittore di pagina di questo sistema se viene usato l'algoritmo di sostituzione con seconda chance?

Dovrà contenere perlomeno il numero del frame che ospita la pagina corrispondente al descrittore, e visto che la paginazione è a domanda, il bit di presenza della pagina e il bit di uso (o riferimento)

Soluzione (c)

c) Nel caso non si verifichino mai page fault, qual è, in nanosecondi, il tempo effettivo di accesso in memoria del sistema se viene usato un TLB con un tempo di accesso di 5 nanosecondi, un hit-ratio del 95% e un tempo di accesso in RAM di 0,08 microsecondi? (È sufficiente riportare l'espressione aritmetica che fornisce il risultato finale)

La formula che permette di calcolare il tempo effettivo di accesso in memoria (Effective Memory Access Time, EMAT) è

$$EMAT = p \times (\text{tempo acc. memoria} + \text{tempo acc. TLB}) + (1 - p) \times (2 \times \text{tempo acc. memoria} [+ \text{tempo acc. TLB}])$$

dove p è la probabilità che la ricerca in TLB abbia successo
Con i dati del problema, la formula assume la forma seguente:

$$EMAT = 0{,}95 \times (80+5) + 0{,}05 \times (2 \times 80 + 5) = 89 \text{ nanosecondi}$$

Se l'HW che supporta la ricerca in parallelo nel TLB e nella RAM

$$EMAT = 0{,}95 \times (80+5) + 0{,}05 \times (2 \times 80) = 88{,}75 \text{ nanosecondi}$$

Esercizio 6 (vale 3 punti)

Si consideri un processo con 7 pagine logiche di 1KB, e le seguenti configurazioni

della tabella delle pagine:

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
0	1 (=si)	1 (=si)	1	512
1	0 (=no)	0 (=no)	0	0
2	1	0	0	102
3	0	0	0	0
4	1	0	1	428
5	1	1	0	123
6	1	1	1	812

e del TLB di 4 posizioni

Posizione	Valida	Pagina Virtuale	Pagina Fisica
0	1	5	123
1	1	6	812
2	0	0	0
3	1	0	512

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 5 (= ⌊5130/1024⌋ = quoziente intero della divisione dell'indirizzo logico 5130 con la dimensione della pagina 1024) il cui descrittore si trova nel TLB (in posizione 0)
- L'indirizzo viene quindi tradotto usando le informazioni nel TLB
- TLB e tabella delle pagine non vengono modificati

Posizione	Valida	Pagina Virtuale	Pagina Fisica
0	1	5	123
1	1	6	812
2	0	0	0
3	1	0	512

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 5 (= ⌊5130/1024⌋ = quoziente intero della divisione dell'indirizzo logico 5130 con la dimensione della pagina 1024) il cui descrittore si trova nel TLB (in posizione 0)
- L'indirizzo viene quindi tradotto usando le informazioni nel TLB
- Se invece l'accesso fosse in scrittura, il TLB resterebbe inalterato, mentre nel corrispondente descrittore nella tabella delle pagine la pagina verrebbe marcata come modificata

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
…	…	…	…	…
4	1	0	1	428
5	1	1	1	123
6	1	1	1	812

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 2 (= ⌊2150/1024⌋) il cui descrittore non è presente nel TLB

Posizione	Valida	Pagina Virtuale	Pagina Fisica
0	1	5	123
1	1	6	812
2	0	0	0
3	1	0	512

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 2 (= ⌊2150/1024⌋) il cui descrittore non è presente nel TLB
- Quindi, per tradurre l'indirizzo, si cercano le informazioni nella tabella delle pagine
- Si scopre così che la pagina si trova già in memoria centrale

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
0	1 (=si)	1 (=si)	1	512
1	0 (=no)	0 (=no)	0	0
2	1	0	0	102
3	0	0	0	0
4	1	0	1	428
5	1	1	0	123
6	1	1	1	812

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- La pagina viene allora marcata 'Riferita' e 'Modificata' (perché l'accesso è in scrittura), e le informazioni ad essa relative vengono caricate anche nel TLB

- La tabella delle pagine in corrispondenza della pagina logica 2 è modificata per segnalare che la pagina è stata riferita di recente e potenzialmente (sarà) modificata (dato che l'accesso è in scrittura):

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
1	0 (=no)	0 (=no)	0	0
2	1	1	1	102
…	…	…	…	…

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- La pagina viene allora marcata 'Riferita' e le informazioni ad essa relative vengono caricate anche nel TLB, dove la pagina è anche marcata 'Modificata' (perché l'accesso è in scrittura)

- Il TLB, nella riga 2 (posizione libera) è modificato nel modo seguente:

Posizione	Valida	Pagina Virtuale	Pagina Fisica
0	1	5	123
1	1	6	812
2	1	2	102
3	1	0	512

- Si noti che ora il TLB è pieno

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 3 (= ⌊4000/1024⌋) il cui descrittore non è presente nel TLB

Posizione	Valida	Pagina Virtuale	Pagina Fisica
0	1	5	123
1	1	6	812
2	1	2	102
3	1	0	512

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 3 (= ⌊4000/1024⌋) il cui descrittore non è presente nel TLB
- Quindi, per tradurre l'indirizzo, si cercano le informazioni nella tabella delle pagine
- Si scopre così che la pagina non si trova in memoria centrale

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
0	1 (=si)	1 (=si)	1	512
1	0 (=no)	0 (=no)	0	0
2	1	1	1	102
3	0	0	0	0
4	1	0	1	428
5	1	1	0	123
6	1	1	1	812

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Si accede alla pagina logica 3 (= ⌊4000/1024⌋) il cui descrittore non è presente nel TLB
- Quindi, per tradurre l'indirizzo, si cercano le informazioni nella tabella delle pagine
- Si scopre così che la pagina non si trova in memoria centrale

Viene pertanto generato un page fault il cui effetto, tra l'altro, è quello di restituire l'indirizzo fisico XYZ del frame in cui viene inserita la pagina virtuale 3

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
0	1 (=si)	1 (=si)	1	512
1	0 (=no)	0 (=no)	0	0
2	1	1	1	102
3	0	0	0	0
4	1	0	1	428
5	1	1	0	123
6	1	1	1	812

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Dopodiché, nel descrittore corrispondente la pagina verrà marcata come riferita e modificata ed il descrittore sarà caricato nel TLB (in una qualche posizione)
- La tabella delle pagine in corrispondenza della pagina logica 3 è modificata come segue:

Pag. Virtuale	Presente	Riferita	Modificata	Pag. Fisica
0	1 (=si)	1 (=si)	1	512
1	0 (=no)	0 (=no)	0	0
2	1	1	1	102
3	1	1	1	XYZ
4	1	0	1	428
5	1	1	0	123
6	1	1	1	812

Soluzione

Dire cosa accade e mostrare come viene modificato il contenuto del TLB e della tabella delle pagine se il processo accede in lettura alla locazione 5130, e in scrittura alle locazioni 2150 e 4000.

- Dopodiché, nel descrittore corrispondente la pagina verrà marcata come riferita e modificata ed il descrittore sarà caricato nel TLB (in una qualche posizione)

- La riga del TLB corrispondente alla pagina (che andrà a sostituire quella di una qualche altra pagina, visto che il TLB è pieno, per esempio in posizione 1) avrà il seguente contenuto:

Posizione	Valida	Pagina Virtuale	Pagina Fisica
…			
1	1	3	XYZ
…			

Esercizio 7 (vale 5 punti)

Si consideri un sistema dotato di 8 pagine fisiche di memoria tutte allocate ad altrettante pagine logiche di due processi A e B. Si supponga che all'istante corrente l'ordine di caricamento e l'associazione tra pagine logiche e fisiche sia il seguente:

Pag. fisica	6	7	0	1	2	3	4	5
Pag. logica	7	4	1	5	9	4	2	8
riferita	1	0	0	0	0	0	0	1
processo	A	A	B	A	A	B	B	A

Si supponga ora che il processo A riferisca la seguente sequenza di pagine logiche: 2, 5, 0, 12, 9 e che in caso di necessità si possano sostituire anche pagine del processo B (strategia di sostituzione globale).

Si descriva brevemente, ma precisamente, l'algoritmo dell'orologio per la sostituzione delle pagine e lo si applichi a partire dalla pagina più a sinistra (pagina fisica 6) aggiornando lo stato della memoria in conseguenza al verificarsi dei page fault e delle corrispondenti sostituzioni di pagina.

Soluzione

- È un'implementazione efficiente dell'algoritmo con seconda chance in cui le pagine sono gestite come una lista circolare
- Partiziona le pagine in 2 categorie tramite il bit di riferimento R:
  - R = 1: quelle usate recentemente (stima del *working set*)
  - R = 0: quelle usate meno di recente
- Mantiene nella variabile *vittima* l'indice della prima pagina da esaminare (che è la pagina successiva a quella che è stata selezionata per ultima)
- Quando viene invocato, l'algoritmo
  1. Considera la pagina il cui indice è in *vittima*
  2. Controlla il bit di riferimento di tale pagina
     - Se R = 0: seleziona la pagina, incrementa la variabile *vittima* e termina
     - Se R = 1: pone R = 0, incrementa la variabile *vittima* e torna al punto 1
- La pagina selezionata viene sostituita da una nuova pagina che viene inserita nella lista circolare nella posizione corrispondente (con R = 1)

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	6	7	0	1	2	3	4	5
Pag. logica	7	4	1	5	9	4	2	8
riferita	1	0	0	0	0	0	0	1
processo	A	A	B	A	A	B	B	A

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	6	7	0	1	2	3	4	5
Pag. logica	7	4	1	5	9	4	2	8
riferita	1	0	0	0	0	0	0	1
processo	A	A	B	A	A	B	B	A

- Al riferimento della pagina logica 2 di A si verifica un page fault e viene rimossa la pagina 4 di A
- Per comodità, riordino le colonne della tabella, portando a destra quelle già esaminate, cosiché *vittima* punta sempre alla prima colonna
- Lo stato della memoria diventa:

Pag. fisica	0	1	2	3	4	5	6	7
Pag. logica	1	5	9	4	2	8	7	2
riferita	0	0	0	0	0	1	0	1
processo	B	A	A	B	B	A	A	A

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	0	1	2	3	4	5	6	7
Pag. logica	1	5	9	4	2	8	7	2
riferita	0	0	0	0	0	1	0	1
processo	B	A	A	B	B	A	A	A

- Al riferimento della pagina logica 5 di A non si verifica alcun page fault
- Quindi l'algoritmo non viene richiamato e vittima continua ad avere lo stesso valore (in pratica, non ho bisogno di riordinare le colonne)
- Lo stato della memoria diventa:

Pag. fisica	0	1	2	3	4	5	6	7
Pag. logica	1	5	9	4	2	8	7	2
riferita	0	1	0	0	0	1	0	1
processo	B	A	A	B	B	A	A	A

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	0	1	2	3	4	5	6	7
Pag. logica	1	5	9	4	2	8	7	2
riferita	0	1	0	0	0	1	0	1
processo	B	A	A	B	B	A	A	A

- Al riferimento della pagina logica 0 di A si verifica un page fault e viene rimossa la pagina 1 di B (la strategia globale lo consente)
- Lo stato della memoria diventa:

Pag. fisica	1	2	3	4	5	6	7	0
Pag. logica	5	9	4	2	8	7	2	0
riferita	1	0	0	0	1	0	1	1
processo	A	A	B	B	A	A	A	A

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	1	2	3	4	5	6	7	0
Pag. logica	5	9	4	2	8	7	2	0
riferita	1	0	0	0	1	0	1	1
processo	A	A	B	B	A	A	A	A

- Al riferimento della pagina logica 12 di *A* si verifica un page fault e viene rimossa la pagina 9 di *A*
- Lo stato della memoria diventa:

Pag. fisica	3	4	5	6	7	0	1	2
Pag. logica	4	2	8	7	2	0	5	12
riferita	0	0	1	0	1	1	0	1
processo	B	B	A	A	A	A	A	A

Soluzione

Sequenza di riferimenti: 2, 5, 0, 12, 9

Stato della memoria

Pag. fisica	3	4	5	6	7	0	1	2
Pag. logica	4	2	8	7	2	0	5	12
riferita	0	0	1	0	1	1	0	1
processo	B	B	A	A	A	A	A	A

- Al riferimento della pagina logica 9 di A si verifica un page fault e viene rimossa la pagina 4 di B
- Lo stato della memoria diventa:

Pag. fisica	4	5	6	7	0	1	2	3
Pag. logica	2	8	7	2	0	5	12	9
riferita	0	1	0	1	1	0	1	1
processo	B	A	A	A	A	A	A	A

Soluzione

Stato della memoria finale

Pag. fisica	4	5	6	7	0	1	2	3
Pag. logica	2	8	7	2	0	5	12	9
riferita	0	1	0	1	1	0	1	1
processo	B	A	A	A	A	A	A	A

Quindi, in totale si verificano 4 page fault