Selezione di esercizi di sistemi operativi (parte 2)

1 - Si supponga che in un sistema di allocazione basata su paginazione un processo acceda in sequenza alle seguenti pagine.

0, 1, 2, 1, 3, 4, 1, 2, 0, 1

Calcolare il numero di page fault, indicando in caso l'eventuale vittima, utilizzando:

-l'algoritmo FIFO con una memoria fisica di 4 frame
-l'algoritmo LRU con una memoria fisica di 3 o 4 frame
-l'algoritmo ottimo con una memoria fisica di 3 o 4 frame
-l'algoritmo second chance (clock) usando solo un bit (bit di riferimento) con una memoria fisica di 4 frame

Traccia di soluzione

-FIFO: 7 page fault
-LRU: 7 page fault con 3 frame, 6 con 4
-Ottimo: 6 page fault con 3 frame, 5 con 4
-Clock :6 page fault

2 - Perché un sistema di paginazione non implementato in hardware non è efficiente? Che cos'è il TLB (Translation- Lookaside-Buffer)?

Traccia di soluzione

Per ogni dato richiesto in memoria sono necessari 2 accessi in memoria (più la somma dell'offset), Il primo sulla page table, il secondo per accedere effettivamente al dato in caso di paginazione eseguita con successo . Più importante: il controllo degli accessi comporterebbe l'intervento del S.O. ad ogni accesso.
Il TLB è una memoria cache che mantiene un numero limitato di associazioni tra numero di pagina e frame. La traduzione tra page index e frame index è molto più veloce rispetto ad un accesso diretto a memoria, la ricerca (lookup) della chiave viene spesso effettuata in hardware in parallelo.

3 - Dire quali delle seguenti affermazioni sulle politiche di sostituzione delle pagine è corretta, fornendo una breve spiegazione:

a) la politica LRU causa in ogni caso meno page fault della politica del Clock
b) la politica del Clock non causa mai meno page fault della politica LRU;
c) la politica del Clock non causa mai più page fault della politica LRU;
d) la politica FIFO non causa mai meno page fault della politica LRU;
e) la politica FIFO non causa mai più page fault della politica LRU;
f) la politica FIFO equivale ad una politica del Clock con zero bit di informazione sull'uso di ogni pagina.

Traccia di soluzione

a) falsa

b) falsa, si prenda ad esempio la sequenza 0 1 2 0 3 1 con 3 frame
c) Falsa
d) falsa, si prenda ad esempio la sequenza 0 1 2 0 3 1 con 3 frame
e) falsa
f) vera

4 - In un sistema di allocazione della memoria a partizionamento dinamico, la lista delle partizioni libere (hole) contiene, nell'ordine, una partizione da 100K, una da 500K, una da 200K, una da 300K e una da 600K. Come verrebbero allocate le seguenti richieste di memoria:

212K, 417K, 112K e 426K

se arrivano in quest'ordine ?

- **-utilizzando l'algoritmo First fit partendo dall'inizio**
- **-utilizzando l'algoritmo First fit partendo dal punto in cui era terminata la ricerca precedente**
- **-utilizzando l'algoritmo Best fit**
- **-utilizzando l'algoritmo Worst fit**

Traccia di soluzione

First fit partendo dall'inizio
212K → 500K, 417K → 600K, 112K → 500K, 426K → NON ALLOCABILE

First fit partendo dal punto in cui era terminata la ricerca precedente
212K → 500K, 417K → 600K, 112K → 600K, 426K → NON ALLOCABILE

Best fit
212K → 300K, 417K → 500K, 112K → 200K, 426K → 600K

Worst fit
212K → 600K, 417K → 500K, 112K → 600K, 426K → NON ALLOCABILE

5 - In un sistema real-time, quale politica è consigliabile adottare per la sostituzione delle pagine: locale o globale? Perché? E in un sistema mainframe?

Traccia di soluzione

Sistema real-time → locale, così si ottiene maggior controllo e determinismo temporale
Sistema mainframe → globale, maggior utilizzo delle risorse

6 - Fornire una definizione di località di memoria, correlando questo concetto con il fenomeno del trashing. Quante località si possono creare nel caso di copia di un'area di memoria contigua? Fornire un numero minimo spiegando qual è il ruolo di ognuna (si supponga il sistema sia sprovvisto di registri)

Traccia di soluzione

Località: area di memoria (insieme di pagine) usata in maniera quasi contemporanea (e

spesso ripetitiva) da un processo. Quando la dimensione delle località di un processo supera la dimensione della memoria disponibile per il processo, si genera una sequenza continua di page fault con conseguente sovraccarico del sistema dovuto alle continue operazione di swap-out e swap-in → trashing

In caso di copia, almeno 3 località : sorgente, destinazione, indici

7 - Descrivere il concetto di I/O Interlock, spiegando i motivi per cui è stato introdotto e i problemi che può evitare

Traccia di soluzione

L'I/O Interlock è un meccanismo per cui la pagina utilizzata per memorizzare dati letti da disco oppure per contenere dai che verranno scritti a disco non deve essere coinvolta in operazioni di swap-out durante tutta la copia.
Questo per evitare di raddoppiare il tempo di accesso a disco ma soprattutto per evitare che le letture/scritture DMA vadano a sovrascrivere frame appartenenti ad altri processi.

8 - Descrivere le strutture dati Open-file table globale e locale, elencando quali informazioni possono contenere (direttamente o attraverso un puntatore) ed indicando come vengono gestite in caso di apertura di un file.

Traccia di soluzione

Globale: contiene le informazioni circa tutti i file aperti nel sistema, ogni entry corrisponde ad un file aperto da qualche processo e punta ad esempio ad un FCB che contiene informazioni circa i permessi di accesso, il proprietario, la dimensione, dove si trova a disco il file, data modifica, ecc...

Locale: punta a entry della tabella globale, per ogni file contiene informazioni circa l'indicatore di posizione e la modalità di apertura.

Apertura:
-Si consulta l'open-file table globale per verificare se il file è già aperto da un altro processo.
-In caso affermativo, si aggiunge una entry nella open-file table di processo e la si fa puntare alla corrispondente entry nella tabella globale. Un contatore associato a quest'ultima viene incrementato.
-Viceversa, prima viene caricato il FCB e aggiornata la tabella globale.

9 - Descrivere i tre tipi di allocazione dei file su disco: contigua, a lista linkata, con blocco indice. Quali tipi di frammentazione caratterizzano ciascuna delle suddette organizzazioni? Quale tipo di accesso a file (diretto o sequenziale) è efficiente nelle diverse organizzazioni? Quale vantaggio comporta l'uso di una FAT (File Allocation Table) rispetto all'organizzazione a blocchi linkati ?

Traccia di soluzione

Contigua: Ciascun file occupa un insieme di blocchi contigui sul disco. Per reperire il file è necessario conoscere solamente la locazione iniziale (indice di blocco) e la lunghezza (in blocchi) → frammentazione sia interna che esterna, molto efficiente con accesso sia sequenziale e che diretto.

Lista linkata: Ciascun file è composto da una linked-list di blocchi a disco: i blocchi possono essere sparsi ovunque nel disco → frammentazione interna, efficiente con accesso sequenziale.

Blocco indice: Ogni file utilizza un blocco indice, ovvero un blocco che contiene i puntatori a tutti i suoi blocchi → frammentazione interna, efficiente con accesso sequenziale e diretto.

FAT: allocazione concatenata attraverso una tabella in cui ogni entry contiene l'indirizzo del successivo blocco appartenente al file. Accesso casuale migliorato rispetto a linked list.

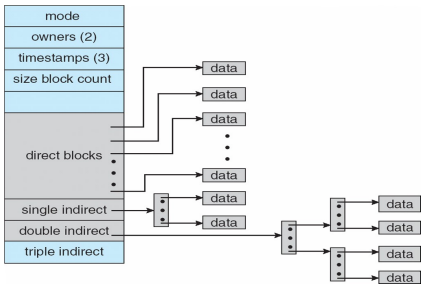
10 - In una organizzazione della directory basata su grafo, come si possono evitare i cicli?

Traccia di soluzione

-Si ammettono solo link a file, oppure
-Si utilizza un algoritmo di identificazione di loop prima di creare un link
-Si attraversa la struttura senza seguire i link a directory
-In caso di self reference, si usa un garbage collector: si attraversa il file system, marcando i file accessibili. In un secondo passaggio in open-file table globale si rimuovono i file non marcati.

11 - Si supponga di avere uno schema combinato di allocazione del file system, simile a quello adottata in UNIX, in cui vi sono:

-12 puntatori diretti a blocchi
-1 puntatore a blocco indice
-1 puntatore ad un blocco indice con due livelli



Sia la dimensione di un blocco di 512 byte e la dimensione dei puntatori a blocchi di 4 byte, qual è la dimensione massima di un file per il quale non sono necessari accessi aggiuntivi per accedere a qualunque blocco? Qual è la dimensione massima di un file supportata dal sistema? Quanti accessi aggiuntivi sono necessari per accedere al byte alla posizione 900K?

Traccia di soluzione

- 512*12 = 6 K
- 512*12 + (512/4) * 512 + (512/4)*(512/4)* 512 = 8262 K
- 2 accessi aggiuntivi