**GESTIONE DELLA MEMORIA**

**MEMORIA VIRTUALE**

Lo scopo delle tecniche di gestione di memorie è quello di tenere contemporaneamente più processi in memoria per permettere la multiprogrammazione, tutta via le tecniche di gestione della memoria come paginazione segmentazione e memoria partizionata richiedono che l’intero processo sia caricato in memoria.

La memoria virtuale è una tecnica che permette che un processo possa essere eseguito anche se non è completamente caricato in memoria.

Il vantaggio della memoria virtuale è che si possono avere programmi di dimensione superiore alla dimensione della memoria fisica.

La memoria virtuale si fonda sulla separazione della memoria logica dalla memoria fisica, questa separazione consente di avere una memoria virtuale molto ampia anche se la memoria fisica disponibile è molto più piccolo.

La memoria virtuale oltre a separare la memoria logica dalla memoria fisica permette a due o più processi di condividere la memoria e i file, attraverso la condivisione delle pagine.

Il caricamento di un eseguibile in memoria può avvenire in due modi:

1. La prima possibilità è che si carichi l’intero programma prima dell’esecuzione.
2. La seconda possibilità è che si caricano in memoria solo le pagine che servono, questa tecnica è detta paginazione su richiesta ed è adotta nei sistemi con memoria virtuale. Le pagine sono caricate in memoria solo quando richieste durante l’esecuzione del programma, quindi può capitare che le pagine a cui non si accede mai non saranno mai caricate in memoria.

Il modulo del sistema operativo che consente la sostituzione delle pagine è detto paginatore.

Quando un processo sta per essere caricato in memoria, il paginatore ipotizza quali pagine saranno usate, prima che il processo sia nuovamente scaricato dalla memoria.

Anziché caricare l’intero processo in memoria, il paginatore trasferirà in memoria solo le pagine che ritiene necessarie. In questo modo si evita di caricare in memoria pagine che non saranno mai utilizzate, riducendo il tempo di avvicendamento e la richiesta di memoria fisica.

Con tale schema è necessario che l’architettura disponga di un meccanismo in grado di distinguere le pagine presenti in memoria da quelle presenti sul disco. A tal fine è utilizzabile un bit di validità che se impostato a valido significa che la pagina è presente in memoria mentre se è impostato a non valido significa che la pagina non è valida oppure la pagina è valida ma attualmente è nel disco.

Occorre notare che se una pagina è contrassegnata come non valida non sortisce alcun effetto sul processo se quest’ultimo non tenta di accedevi.

Se l’ipotesi del paginatore è esatte e in memoria sono caricate tutte le pagine effettivamente necessarie allora il processo verrà eseguito proprio come se fossero caricate in memoria tutte le pagine.

Se invece il processo tenta di accedere ad una pagina che non è presente in memoria e quindi è contrassegnata come non valida allora si genera un’eccezione di page fault (pagina mancate).

L’architettura di paginazione traducendo l’indirizzo attraverso la tabella delle pagine nota il bit di validità impostato a non valido e invia un segnale di eccezione al sistema operativo.

Tale eccezione è dovuta ad un insuccesso del sistema operativo nella scelta delle pagine da caricare in memoria.

Procedure di gestione di page fault:

1. Si controlla nella tabella interna al processo allo scopo di stabilire se il riferimento fosse un acceso alla memoria valido o non valido.
2. Se il riferimento era non valido si termina il processo altrimenti se il riferimento è valido si carica la pagina richiesta in memoria.
3. Si individua un frame libero
4. Si programma un’operazione sui dischi in moda da trasferire la pagina richiesta ne frame assegnato
5. Quando la lettura dal disco è terminata si modificano la tabella interna del processo e la tabella delle pagine.
6. Si riavvia il processo.

Durante l’esecuzione di un processo utente, se si verifica un page fault il sistema operativo determina la locazione del disco in cui risiede la pagina desiderata e carica in un dei frame liberi la pagina, ma se la lista dei frame liberi è vuota (tutta la memoria è in uso) ne consegue che la pagina richiesta non può essere caricata in memoria; il sistema operativo ha tre possibilità:

1. Terminare il processo
2. Scaricare dalla memoria l’intero processo abbassando il grado di multiprogrammazione
3. Sostituzione delle pagine

La sostituzione delle pagine e una tecnica utilizzata quando non esiste alcun frame libero e consiste nell’individuare un frame inutilizzato e liberarlo; per liberare un frame si scrive il suo contenuto all’interno dell’area di avvicendamento e si modifica la tabella delle pagine per indicare che la pagina non si trova più in memoria. Il frame liberato può contenere la pagina che ha causato l’eccezione.

Gestione del Page fault modificato in modo da includere la sostituzione delle pagine

1. S’individua la locazione nel disco della pagina richiesta.
2. Si cerca un frame libero, se esiste lo si usa; altrimenti si impiega un algoritmo di sostituzione delle pagine per scegliere un frame vittima;
3. Si scrive la pagina vittima nel disco, si modificano le tabelle delle pagine e quelle dei frame.
4. Si scrive la pagina richiesta nel frame appena liberato, si modificano la tabella delle pagine e quella dei frame
5. Si riavvia il processo

Occorre notare che se non esiste alcun frame libero sono necessari due trasferimenti di pagina, uno fuori e uno dentro la memoria. Questa situazione raddoppia il tempo di servizio dell’eccezione di pagina mancante e aumenta anche il tempo effettivo d’accesso.

Questo sovraccarico si può ridurre usando un bit di modifica, ad ogni pagina (o frame) è associato un bit di modifica che s’imposta automaticamente ogni volta che nella pagina si scrive una parola o un byte, indicando che la pagina è stata modificata.

Quando si sceglie una pagina da sostituire si esamina il suo bit di modifica è se attivo significa che la pagina è stata modificata rispetto a quando era stata letta dal disco, in questo caso la pagina deve essere scritta sul disco.

Se il bit di modifica non è attivo significa che la pagina non è stata modificata da quando è stata caricata in memoria, quindi se la sua copia sul disco non è stata sovrascritta da qualche altra pagina, non è necessario scrivere nel disco la pagina già c’è.

La sostituzione di una pagina è fondamentale al fine della paginazione su richiesta perché completa la separazione tra memoria logica e memoria fisica.

**ALGORITMI DI SOSTITUZIONE DELLE PAGINE**

**Sostituzione delle pagine secondo l’ordine di arrivo (FIFO)**

L’algoritmo FIFO è l’algoritmo più semplice e prevede che ad ogni pagina si associ l’istante di tempo in cui la pagina è stata portata in memoria e se si deve sostituire una pagina si selezione quella che si trova in memoria da più tempo.

Non è necessario registrare l’istante di tempo in cui si carica una pagina in memoria basta semplicemente strutturare tutte le pagine in memoria secondo una coda FIFO, in questo caso si sostituisce la pagina che si torva in testa alla coda e la nuova pagina si aggiunge alla fine della coda.

L’algoritmo FIFO è facile da capire e da programmare però è poco efficiente, infatti la pagina sostituita potrebbe contenere un modulo di inizializzazione usato molto tempo prima e che non serve più, ma potrebbe anche contenere una variabile molto usata, inizializzata precedentemente e ancora in uso.

**Anomalia di Belady** ci dice che con alcuni algoritmi di sostituzione delle pagine, la frequenza dell’assenza delle pagine può aumentare con l’aumentare del numero dei frame asseganti.

L’algoritmo FIFO soffre dell’anomalia di Belady.

**Sostituzione ottimale delle pagine**

Si sostituisce la pagina che non si userà per il periodo di tempo più lungo.

Difficile da implementare perché richiede la conoscenza futura della successione dei riferimenti.

Non soffre della anomalia di belady

**Sostituzione delle pagine usate meno recentemente (LRU)**

Si sostituisce le pagine che non viene utilizzata da più tempo.

La sostituzione LRU associa ad ogni pagina l’istante in cui è stata usata l’ultima volta e quando occorre sostituire la pagina si sceglie quella che non è stata usata da più tempo.

Non soffre della anomalia di belady

**Varianti LRU**

Sono pochi i sistemi che dispongono di un’architettura adatta ad una vera sostituzione LRU delle pagine.

Molti sistemi possono fornire un aiuto un bit di riferimento.

Il bit di riferimento ad una pagina è impostato automaticamente dall’architettura del sistema ogni volta che si fa riferimento a quella pagina.

I bit di riferimento sono associati a ogni elemento della tabella delle pagine.

Inizialmente i bit di riferimento di tutte le pagine sono a zero e s’imposta ad 1 il bit di ogni pagina a cui si fa riferimento dopo qualche tempo è possibile stabilire quali pagine sono state usate esaminando semplicemente il bit di riferimento. Non è però possibile conoscere l’ordine d’uso.

**Bit di riferimento supplementari**

Ulteriori informazioni si possono ottenere registrando il bit di riferimento ad intervalli regolari.

È possibile conservare in una tabella in memoria una serie di bit per ogni pagina.

Ad intervalli regolari ad esempio ogni 100ms, un segnale di interruzione del timer del sistema trasferisce il controllo al sistema operativo che sposta il bit di riferimento nel bit più significativo della sequenza traslando gli altri di 1 bit ed eliminando il bit più significativo in questo caso si sostituisce la pagina con la sequenza minore.

**Algoritmo seconda chance**

L’algoritmo della seconda chance è un algoritmo di tipo FIFO.

Dopo aver selezionato una pagina si controlla il bit di riferimento se è a 0 si sostituisce la pagina altrimenti se il bit è impostato ad 1 si dà una seconda chance alla pagina e la selezione passa alla pagina successiva.

Per realizzare un algoritmo con seconda chance si utilizza una coda circolare in cui un puntatore indica qual è la prima pagina da sostituire.

Quando serve un frame si va avanzare il puntatore finché non si trovi in corrispondenza di una pagina con il bit di riferimento a 0, a ogni passo si azzera il bit di riferimento appena esaminato.

Una volta trovata una pagina vittima si sostituisce e si inserisce la nuova pagina nella coda circolare nella posizione corrispondente.

**Algoritmo seconda chance migliorato**

l’algoritmo della seconda chance può essere migliorato considerando i bit di riferimento e di modifica come una coppia ordinata con cui si possono ottenere 4 classi:

0,0 la pagina non è stata né usata recentemente e né modificata ottima pagina da sostituire

0,1 non usata recentemente ma modificata, la pagina prima di essere sostituita de essere scritta in memoria

1,0 usata recentemente, non modificata probabilmente la pagina sarà presto ancora usata

1,1 usata recentemente e modificata probabilmente la pagina sarà presto ancora usata e dovrà essere in memoria prima di essere sostituita.

Si sostituisce la prima pagina che si trova nella classe minima.

Prima che la pagina da sostituire venga individuata bisogna scandire la coda circolare più volte.

La differenza principale tra l’algoritmo di seconda chance con quello migliorato è che nel migliorato si dà la preferenza alle pagine modificate al fine di ridurre il numero di I/O richiesti.

**Algoritmi basati sul conteggio**

In questi algoritmi ogni volta che una pagina viene richiamata dalla CPU viene aumentato il suo contatore

**sostituzione della pagina meno usata**: si sostituisce la pagina col contatore minore in quanto si suppone che non sia particolarmente utile.

**sostituzione della pagina più usata**: si presuppone che le pagine con contatore basso siano state

inserite da poco per cui debbano essere ancora usate

**Algoritmi di sostituzione**

Gli algoritmi di sostituzione si dividono in due categorie:

**globali**: il frame da liberare viene scelto tra tutti i processi.

**locali**: il frame da liberare viene scelto solo tre i frame già appartenenti al processo

Si è notato che gli algoritmi globali aumentano la produttività del sistema ma soffrono perché si estendo situazioni di malessere anche agli altri processi, se c’è un processo che ha bisogno di tanti frame contemporaneamente, con una strategia globale io vado a rubare agli altri i frame.

**TRASHING**

Si parla di trashing qunado si verifica un numero troppo elevato di page fault e si spende più tempo per lo swapping che per la reale esecuzione dei processi.

Il trashing è un femomeno che consiste nell’attività della CPU a trasferire continuamente pagine tra la memoria fisica e la swap area.

Gli effetti di questa situazione si possono limitare usando un algoritmo di sostituzione locale, o algoritmo di sostituzione per priorità. Con la sostituzione locale, se un processo ricade nell’attività di paginazione degenere, non può sottrare frame a un altro processo e quindi provocare a sua volta la degenerazione.