Lezione 10

Gestione della memoria virtuale

Sistemi operativi

22 maggio 2012

Marco Cesati

System Programming Research Group
Università degli Studi di Roma Tor Vergata

Di cosa parliamo in questa lezione?

La gestione della memoria virtuale:

1. Regioni di memoria
2. Algoritmi di sostituzione delle pagine
3. Politiche di assegnazione della memoria fisica

La memoria virtuale

La separazione dello spazio di indirizzamento fisico da quello virtuale porta al concetto di *memoria virtuale*

Sistema operativo con memoria virtuale

È in grado di associare a ciascun processo un insieme di pagine di memoria (indirizzi virtuali) che non debbono essere necessariamente associate a pagine fisiche

La memoria virtuale è basata su diversi meccanismi hardware e software, tra cui:

- I circuiti di traduzione degli indirizzi (da virtuale a fisico)
- La tecnica della paginazione su richiesta (demand paging)
- La tecnica della condivisione delle pagine fisiche
- La tecnica della copiatura su scrittura (COW)

La memoria virtuale è legata anche alla gestione della memoria secondaria

Regioni di memoria virtuale

Ogni SO deve tenere traccia degli indirizzi virtuali che un processo è autorizzato ad utilizzare e che costituiscono la sua memoria virtuale

Regione di memoria virtuale

Insieme di indirizzi virtuali assegnato ad un processo e caratterizzato da

- indirizzo virtuale iniziale
- lunghezza
- provenienza dei dati nella regione
- attributi e diritti di accesso

Le regioni di memoria virtuale sono memorizzate in una apposita struttura di dati attestata nel PCB che consente di svolgere efficientemente le seguenti operazioni:

- Ricerca di una regione contenente un dato indirizzo
- Fusione di regioni adiacenti
- Rimozione di una regione o di parte di essa

Provenienza dei dati in una regione di memoria virtuale

La provenienza dei dati in una regione condiziona molti aspetti legati alla gestione della memoria virtuale

- Regione *anonima*: le pagine contengono inizialmente zeri, e sono destinate a contenere i dati elaborati dal processo
  - Il SO assegna page frame riempite con zeri
- Regione *mappata su file* (*file memory mapping*): le pagine contegono inizialmente i dati contenuti in una certa posizione di un file in memoria secondaria
  - Le page frame sono inizializzate con i dati trasferiti dal file

Per le regioni mappate su file vi sono due possibilità:

- *Mappatura condivisa*: le modifiche alle pagine vengono automaticamente convertite dal SO in modifiche al file in memoria secondaria
- *Mappatura privata:* le modifiche alle pagine non hanno conseguenze sul file in memoria secondaria

Sovrallocazione della memoria

Il più immediato vantaggio di un SO con memoria virtuale è la possibilità di sovrallocare la memoria

Sovrallocazione della memoria

Il SO permette di attivare un insieme di processi concorrenti le cui esigenze di memoria totali superano la capacità della memoria fisica a disposizione nel sistema

Nei SO progettati per architetture basate sulla paginazione la sovrallocazione della memoria dipende da tre meccanismi:

- La paginazione su richiesta
- L'algoritmo di sostituzione delle pagine
- La politica di assegnazione delle page frame

La sovrallocazione della memoria dipende anche e in modo essenziale dalla possibilità di memorizzare il contenuto delle pagine su memoria secondaria

Gestione della memoria virtuale
Marco Cesati

Schema della lezione
Regioni di memoria
Sostituzione delle pagine
Assegnazione della memoria fisica

Salvataggio delle pagine in memoria secondaria

- Una pagina appartenente ad una regione di memoria anonima non è associata ad alcun file
  - Il SO salva il contenuto della pagina in una area di memoria secondaria detta *area di swap*
- Una pagina appartenente ad una regione di memoria mappata su file di tipo condiviso:
  - Se non è stata modificata non è necessario salvarla (i dati sono recuperabili dal file)
  - Se è stata modificata, il SO scrive le modifiche sul file
- Una pagina appartenente ad una regione di memoria mappata su file di tipo privato:
  - Se non è stata modificata, non è necessario salvarla (i dati possono essere recuperati dal file)
  - Se è stata modificata deve essere salvata nell'area di swap

Un processo può utilizzare pagine non contenute in alcuna regione di memoria? ***No!***

Durante la gestione del page fault il SO invece di assegnare una page frame invia un segnale di *Segmentation Fault*

Algoritmi di sostituzione delle pagine fisiche

Se il SO consente di sovraccaricare la memoria, ciascuna richiesta di assegnazione di una nuova page frame può fallire perché queste sono tutte in uso

Quando nel sistema non vi è più un numero sufficiente di page frame libere, il SO avvia un meccanismo di recupero della memoria fisica (page frame reclaiming)

Uno dei suoi componenti fondamentali tenta di recuperare page frame assegnate ai processi attivi nel sistema:

- *Avvicendamento* (*swapping*): nei sistemi senza paginazione e senza memoria virtuale l'intero processo vittima viene salvato in memoria secondaria e tutte le sue page frame recuperate
- *Sostituzione delle pagine*: il SO seleziona un piccolo numero di pagine "vittime" in modo equo tra i vari processi, le salva in memoria secondaria e recupera le page frame

Nei SO moderni il termine *swapping* è utilizzato spesso come sinonimo per *sostituzione delle pagine*

Algoritmi di sostituzione delle pagine fisiche (2)

Ciascun SO definisce un proprio algoritmo di sostituzione delle pagine: unico, specifico ed ottimizzato per il tipo di utilizzo ed il carico atteso del sistema

Una metrica per valutarne la qualità è la frequenza delle assenze di pagina (page fault)

- Si fissa una quantità di pagine fisiche a disposizione
- Si fissa una determinata sequenza di accessi alle pagine
- Si simula il comportamento dell'algoritmo di sostituzione delle pagine contando il numero di page fault risultanti
- Tanto minore è il numero di page fault, tanto migliore è l'algoritmo

Ciascun *page fault* comporta un considerevole ritardo nei tempi d'esecuzione di un processo, pertanto le prestazioni globali di un SO dipendono in modo cruciale dalla qualità del suo algoritmo di sostituzione

Algoritmo di sostituzione delle pagine FIFO

Il più semplice algoritmo di sostituzione delle pagine è di tipo FIFO: dovendo sostituire una pagina, si sceglie quella a cui è stata assegnata una page frame da più tempo

Può essere realizzato:

- memorizzando l'istante di tempo delle assegnazioni delle page frame, oppure
- mantenendo una struttura FIFO con tutte le pagine dei processi ordinata per istante di assegnazione

Quali sono i vantaggi e svantaggi di questo algoritmo?

- Il suo pregio principale è la semplicità di realizzazione
- Però non offre buone prestazioni, perché l'istante di tempo in cui è stata assegnata una pagina fisica non è correlato all'utilizzo della pagina stessa

Algoritmo di sostituzione delle pagine FIFO (2)

Supponiamo di avere a disposizione tre pagine fisiche e di aver sovrallocato la memoria con otto pagine logiche; la sequenza dei riferimenti in memoria sia

$$7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1$$

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
	0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
		1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1

Totale: 15 page fault

Aumentando il numero di page frame a disposizione il numero di page fault tende necessariamente a diminuire? ***No!***

In effetti potrebbe anche aumentare: anomalia di Belady

Anomalia di Belady

Comportamento paradossale di un algoritmo di sostituzione per il quale aggiungendo pagine fisiche a disposizione il numero di page fault aumenta invece di diminuire

Esempio di applicazione dell'alg. FIFO con 3 e 4 page frame:

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	4	4	4	5	5	5	5	5	5
	2	2	2	1	1	1	1	1	3	3	3
		3	3	3	2	2	2	2	2	4	4

Totale: 9 page fault

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	5	5	5	5	4	4
	2	2	2	2	2	2	1	1	1	1	5
		3	3	3	3	3	3	2	2	2	2
			4	4	4	4	4	4	3	3	3

Totale: 10 page fault

Gestione della memoria virtuale
Marco Cesati

Schema della lezione
Regioni di memoria
Sostituzione delle pagine
Assegnazione della memoria fisica

Algoritmo ottimale di sostituzione delle pagine

Strategia ottimale

La pagina migliore da sostituire è quella che non verrà utilizzata per il tempo più lungo

- L'algoritmo è ottimale poiché il numero di page fault è il più piccolo possibile
- L'algoritmo non è pratico, perché è molto difficile prevedere gli accessi alle pagine futuri

Esempio di applicazione dell'algoritmo OPT:

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
	0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1

Totale: 9 page fault

Algoritmo di sostituzione delle pagine LRU

Dal principio di località temporale dei programmi si ha che:

La probabilità di accedere ad una pagina utilizzata di recente è maggiore della probabilità di accedere ad una pagina utilizzata nel lontano passato

Anche se non è possibile prevedere in modo certo gli accessi alla memoria, si può inferire la storia futura da quella passata:

L'algoritmo LRU (Least Recently Used) associa ad ogni pagina l'istante in cui è stata acceduta per l'ultima volta, e seleziona come pagina da sostituire la pagina che non è stata usata per il periodo di tempo più lungo

Detto S una sequenza di accessi alla memoria, ed S' la sequenza di accessi rovesciata, la frequenza di page fault dell'algoritmo OPT su S è uguale a quella di LRU su S'

Algoritmo di sostituzione delle pagine LRU (2)

Esempio di applicazione dell'alg. LRU con 3 page frame:

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7

Totale: 12 page fault

Qual è la difficoltà più evidente nell'implementare l'alg. LRU?

Memorizzare l'istante dell'ultimo accesso a ciascuna pagina

Due possibili implementazioni:

- Contatore incrementato ad ogni accesso in memoria e salvato nella voce della tabella di paginazione
- Pila di numeri di pagina ordinata per ultimo accesso

In entrambi i casi, una implementazione puramente software sarebbe troppo costosa in termini di tempo d'accesso

Algoritmi di sostituzione a pila

L'alg. LRU è un esempio di algoritmo di sostituzione a pila:

Caratterizzazione degli algoritmi a pila

L'insieme delle pagine caricate in memoria avendo a disposizione n page frame è sempre un sottoinsieme dell'insieme delle pagine caricate in memoria avendo a disposizione $n+1$ page frame

Qual è il principale vantaggio degli algoritmi a pila?

Gli algoritmi di sostituzione delle pagine a pila non sono soggetti all'anomalia di Belady

Infatti aumentando il numero di page frame non si può avere un numero maggiore di page fault perché tutte le pagine che erano in memoria con meno page frame sono ancora presenti

I bit di riferimento delle pagine

Anche a livello hardware, implementare un meccanismo che memorizzi gli istanti di tempo oppure l'ordine degli ultimi accessi alle pagine di memoria è costoso

Tuttavia molte architetture offrono il *bit di riferimento* (*access bit*) delle pagine:

- Contenuto in ciascuna voce delle tabelle di paginazione
- Impostato automaticamente a 1 per ogni accesso alla pagina corrispondente
- Periodicamente impostato a 0 dal SO

Grazie ai bit di riferimento è possibile conoscere quali pagine di memoria sono state accedute in un certo intervallo di tempo (ma non l'ordine esatto)

Gli algoritmi di sostituzione delle pagine nei SO moderni utilizzano i bit di riferimento per approssimare l'alg. LRU

Algoritmo di sostituzione con bit supplementari

Un metodo per sfruttare i bit di riferimento messi a disposizione dall'hardware consiste nel campionare periodicamente e salvare lo stato dei bit di tutte le pagine

Ad intervalli di tempo regolari viene eseguita una procedura che effettua una scansione di tutte le voci di tutte le tabelle di paginazione; per ciascuna voce

- legge il valore del bit di riferimento
- lo salva in un contatore associato alla pagina (scorrendo i bit già presenti verso destra)
- azzera il bit di riferimento

Ad esempio, se il contatore associato a ciascuna pagina ha 8 bit ed il bit di riferimento è letto ogni 100 ms:

- 000000000: pagina non acceduta negli ultimi 800 ms
- 110101000: pagina acceduta da meno di 100 ms, e con almeno 4 accessi negli ultimi 900 ms
- 000001111: almeno 400 ms dall'ultimo accesso

Algoritmo di sostituzione con bit supplementari (2)

L'algoritmo di sostituzione delle pagine con bit supplementari seleziona sempre una delle pagine con il minimo valore costituito dal bit di riferimento e dal contatore

- Se $x > y$, la pagina con il valore x è stata acceduta più di recente della pagina con il valore y
- È un algoritmo che approssima LRU (l'approssimazione è data dalla frequenza con cui si campionano i bit di riferimento e la dimensione del contatore)
- Non è possibile determinare l'ordine LRU di pagine con lo stesso valore: l'algoritmo ne seleziona una a caso, oppure adotta un criterio FIFO, o le sostituisce tutte...

Il numero di bit del contatore è un compromesso tra la durata del periodo di osservazione, la quantità di memoria occupata dai contatori, ed il costo dell'operazione di aggiornamento

Algoritmo di sostituzione con seconda chance

L'algoritmo di sostituzione delle pagine con seconda chance (o algoritmo a orologio) è un algoritmo di tipo FIFO basato sul bit di riferimento

- Tutte le pagine sono incluse in una lista circolare
- Il SO memorizza la posizione in lista dell'ultima pagina analizzata
- In caso di necessità di recupero di una page frame, il SO considera il bit di riferimento della pagina corrente:
  - Se è zero, la pagina viene selezionata, rimossa dalla lista, e al suo posto è inserita la nuova pagina ($\Rightarrow$ FIFO)
  - Se è uno, le viene data una seconda chance: il bit di riferimento è impostato a zero, e si analizza la pagina successiva

Una pagina a cui è data una seconda chance non verrà sostituita finché tutte le altre pagine non siano state sostituite o data loro una seconda chance

Se tutte le pagine sono accedute con alta frequenza, questo algoritmo diventa equivalente all'algoritmo FIFO

I bit di modifica delle pagine

Generalmente le architetture hardware offrono all'interno delle voci delle tabelle di paginazione un altro bit oltre a quello di riferimento: il *bit di modifica* (*dirty bit*)

- Contenuto in ciascuna voce delle tabelle di paginazione
- Impostato automaticamente a 1 per ogni accesso in scrittura alla pagina corrispondente
- Impostato a 0 esclusivamente dal SO

Qual è l'utilità dei bit di modifica delle pagine?

- Quando una pagina viene sostituita, in generale è necessario salvare in memoria secondaria il suo contenuto
- Se il contenuto di una pagina è stato letto o già salvato in memoria secondaria, e la pagina non è stata modificata, il SO non ha necessità di salvare di nuovo il suo contenuto
- Le pagine non modificate possono essere sostituite molto più velocemente di quelle modificate

Algoritmo di sostituzione con seconda chance migliorato

L'algoritmo di sostituzione delle pagine con seconda chance migliorato è una variante in cui si considera la coppia di valori

(bit di riferimento, bit di modifica)

- $(0,0)$: né usata di recente né modificata: miglior scelta per la sostituzione
- $(0,1)$: non usata di recente ma modificata: anche se non è stata utilizzata di recente, deve essere salvata in memoria secondaria
- $(1,0)$: usata di recente ma non modificata: probabilmente sarà utilizzata ancora nel prossimo futuro
- $(1,1)$: usata di recente e modificata: peggior scelta per la sostituzione

Quando il SO cerca una pagina da sostituire, dapprima scandisce la lista circolare cercando quelle con valore $(0,0)$, poi se la ricerca non ha avuto successo continua cercando quelle con valore $(0,1)$, e così via

Algoritmi di sostituzione delle pagine con conteggio

Un'altra classe di algoritmi di sostituzione delle pagine è basata su contatori di accesso alle pagine

- Algoritmo LFU (Least Frequently Used): si sostituisce la pagina con valore minore nel contatore; l'idea è che le pagine utilizzate attivamente sono accedute spesso e quindi hanno valori grandi
- Algoritmo MFU (Most Frequently Used): si sostituisce la pagina con valore maggiore nel contatore; l'idea è che le pagine con valori bassi sono state assegnate più di recente

In generale questo tipo di algoritmi sono poco utilizzati:

- sono costosi in termini di tempo d'esecuzione
- non approssimano bene l'algoritmo ottimale OPT

Memorizzazione transitoria delle pagine

Tutti gli algoritmi di sostituzione delle pagine possono beneficiare di tecniche software che riducono i tempi d'esecuzione delle singole sostituzioni

Ad esempio, generalmente i SO definiscono una *riserva (pool) di page frame libere* da utilizzare per soddisfare nell'immediato le richieste di memoria

Esempio di procedura implementata da un SO:

- Ogni richiesta di memoria per una singola page frame viene esaudita attingendo al pool, se possibile
- Ogni singola page frame rilasciata viene inserita nel pool
- L'algoritmo di sostituzione delle pagine è attivato con bassa priorità quando il numero di page frame nel pool scende al di sotto di una certa soglia
- Se il pool si svuota completamente, ogni richiesta di page frame deve essere esaudita previa sostituzione di una pagina, come al solito

Memorizzazione transitoria delle pagine (2)

- Il vantaggio offerto dal pool di page frame libere è che le richieste di page frame singole tendono ad essere esaudite con maggiore velocità
  - Infatti non è necessario attendere la conclusione di un trasferimento in memoria secondaria per esaudire la richiesta
  - Il trasferimento di dati avviene in modo concorrente rispetto all'esecuzione del processo che ha richiesto la page frame
- Un altro vantaggio è che il SO può tenere traccia delle pagine che erano memorizzate nelle page frame del pool
  - Un accesso ad una di queste pagine risulta in un page fault che può essere gestito senza trasferimenti dalla memoria secondaria

Altra tecnica per ridurre i tempi dell'algoritmo di sostituzione: scandire regolarmente ma con bassa priorità le tabelle di paginazione e avviare il trasferimento in memoria secondaria delle pagine modificate

Assegnazione della memoria fisica ai processi

Un altro componente del SO legato alla gestione della memoria virtuale è quello che controlla le assegnazioni di memoria fisica ai processi

A causa della sovrallocazione della memoria, la dimensione dello spazio di memoria virtuale richiesto da ciascun processo può essere maggiore della effettiva disponibilità di memoria fisica per lo stesso processo

Esistono diversi approcci possibili:

- Nessuna politica di allocazione: i processi non hanno una quota predefinita di memoria fisica a loro disposizione
- Allocazione uniforme: a ciascun processo viene assegnata una quota della memoria fisica che dipende dalla capacità della RAM e dal numero di processi
- Allocazione proporzionale: a ciascun processo viene assegnata una quantità di memoria fisica che dipende dalla capacità della RAM e dalla dimensione relativa della propria memoria virtuale rispetto a quella di tutti i processi

Assegnazione della memoria fisica ai processi (2)

Problemi nei diversi approcci:

- Nella allocazione uniforme tutti i processi ottengono la stessa quota a prescindere dalle reali esigenze
- Sia nella allocazione uniforme che in quella proporzionale non si tiene conto della priorità dei processi o del loro stato di esecuzione (in attesa di I/O, sospesi dall'utente, ...)
- Sia nella allocazione uniforme che in quella proporzionale un algoritmo di sostituzione delle pagine *globale* può togliere page frame ad un processo per darle ad un altro
  - Gli algoritmi di sostituzione *locali* cercano una pagina da sostituire all'interno delle tabelle di paginazione dello stesso processo che ha effettuato la richiesta

In molti sistemi operativi non viene utilizzata alcuna politica di assegnazione predefinita della memoria fisica ai processi

Paginazione degenere

In assenza di politiche di assegnazione predefinita della memoria fisica ai processi è possibile che il sistema assuma uno stato "patologico"

Paginazione degenere (*trashing*)

Condizione dovuta alla mancanza di page frame libere che causa il verificarsi a ripetizione di page fault da parte dei processi in esecuzione, con conseguente esecuzione continua dell'algoritmo di sostituzione delle pagine

Lo stato di paginazione degenere porta in pratica al blocco dell'attività del sistema: ciascun processo cerca di continuare l'esecuzione accedendo alle proprie pagine, ma per assegnare le page frame richieste il SO toglie le page frame ad altri processi attivi, causando quindi a breve nuove page fault

Alla fine tutti i processi sono bloccati in attesa di trasferimenti di I/O, e l'unica attività della CPU consiste nell'eseguire l'algoritmo di sostituzione delle pagine

L'insieme di lavoro di un processo

Un metodo per evitare la paginazione degenere consiste nel fare in modo che ogni processo abbia a disposizione una quantità di memoria fisica sufficiente per continuare l'esecuzione senza un numero eccessivo di page fault

Insieme di lavoro (working set)

Le ultime pagine fisiche utilizzate da un processo in un intervallo temporale di lunghezza prefissata τ

- La definizione dipende dalla lunghezza τ dell'intervallo
  - Alternativa: considerare gli ultimi Δ riferimenti alle pagine
- A causa del principio di località l'insieme di lavoro contiene la maggior parte delle page frame che saranno necessarie per l'esecuzione nel prossimo futuro
- La dimensione dell'insieme di lavoro indica la quota di page frame necessaria ad evitare la paginazione degenere

Se la somma degli insiemi di lavoro supera la quantità di memoria fisica disponibile il SO sospende un processo "vittima" recuperando tutte le page frame (avvicendamento)

Frequenza delle eccezioni di pagina assente

Un altro meccanismo per evitare la paginazione degenere consiste nel monitorare la frequenza dei page fault (PFF, Page Fault Frequency)

- Se la frequenza dei page fault di un processo scende sotto una determinata soglia, è possibile sottrarre page frame al processo senza gravi conseguenze
- Se la frequenza dei page fault per un certo processo sale oltre una determinata soglia è necessario assegnargli nuove page frame e/o evitare che gli vengano sottratte

Anche in questo caso se la memoria fisica libera si esaurisce completamente potrebbe essere necessario sospendere un processo e recuperare tutte le sue page frame