Sistemi Operativi

Ionut Zbirciog

16 November 2023

MEMORIA VIRTUALE

Mentre da un lato i registri base e limite possono essere utilizzati per creare l'astrazione degli spazi degli indirizzi, dall'altro sorge un nuovo problema: la gestione del bloatware, cioè del software sempre più "gonfio". Le dimensioni della memoria sono in costante aumento, ma quelle dei software aumentano ancora più velocemente.

L'idea alla base della memoria virtuale è che ogni programma ha un suo spazio degli indirizzi, suddiviso in parti chiamati pagine. Ogni pagina è un intervallo di indirizzi contigui. Queste pagine sono mappate sulla memoria fisica, ma per eseguire il programma non è indispensabile che tutte le pagine siano contemporaneamente nella memoria fisica. Quando il programma fa riferimento a una parte del suo spazio degli indirizzi che è nella memoria fisica, l'hardware esegue direttamente la mappatura necessaria. Quando il programma fa riferimento a una parte del suo spazio degli indirizzi che non è nella memoria fisica, il sistema operativo viene allertato, va a prelevare la parte mancante ed esegue nuovamente fallita.

Implementazioni diverse della memoria virtuale adottano scelte diverse rispetto a queste unità; oggi la maggior parte dei sistemi impiega una tecnica detta paging (paginazione) che prevede unità di dimensioni fisse, ad esempio di 4 KB. Una soluzione alternativa detta segmentazione usa come unità interi segmenti di dimensione variabile.

PAGINAZIONE

La maggior parte dei sistemi di memoria virtuale usa una tecnica chiamata paginazione o paging. Su qualsiasi computer i programmi fanno riferimento a un set di indirizzi di memoria, come ad esempio quando un programma esegue un'istruzione come `MOV REG,1000`.

```
MOV REG,1000
```

Gli indirizzi generati dal programma sono chiamati indirizzi virtuali e formano lo spazio degli indirizzi virtuali. Sui computer senza memoria virtuale, l'indirizzo virtuale è posto direttamente sul bus di memoria e provoca la lettura o la scrittura della parola della memoria fisica con lo stesso indirizzo. Quando è utilizzata la memoria virtuale, gli indirizzi virtuali non vanno direttamente al bus di memoria, ma a una MMU (Memory Management Unit, unità di gestione della memoria) che mappa gli indirizzi virtuali sugli indirizzi della memoria fisica (circuito presente nella CPU).

Lo spazio degli indirizzi virtuali è suddiviso in unità di dimensione fissa, chiamate pagine. Le unità corrispondenti nella memoria fisica sono chiamate frame o page frame. Le pagine e i frame

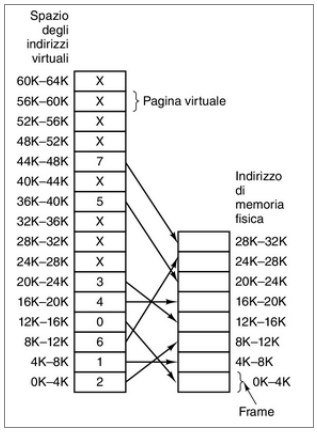

Figure 1: Una tabella delle pagine con 16 pagine e 8 frame

sono della stessa dimensione, ad esempio da 4 KB. Nei sistemi reali le pagine vanno da 512 byte fino a 64 KB.

- 64 Kb di pagine virtuali possono essere mappate in 8 frame, usando 16 bit per indirizzo.
- I trasferimenti fra RAM e memoria non volatile sono sempre di pagine intere. Molti processori supportano più dimensioni di pagina, che possono essere mescolati e abbinati a discrezione del sistema operativo.
- Ad esempio, l'architettura x86-64 supporta pagine da 4 KB, 2 MB e 1-GB, e potremmo utilizzare le pagine da 4 KB per le applicazioni utente e una singola pagina da 1 GB per il kernel.

Esempi

1. L'istruzione `MOV REG,8192` è trasformata effettivamente in `MOV REG,24576` poiché l'indirizzo virtuale 8192 (nella pagina virtuale 6) è mappato sul 24576 (nel frame fisico 24K–28K).
2. L'indirizzo virtuale 20500 dista 20 byte dall'inizio della pagina virtuale 3 (indirizzi virtuali da 20480 a 24575) e la sua mappatura sull'indirizzo fisico è 12288 + 20 = 12308.

Questa capacità di mappare le 16 pagine virtuali su uno qualsiasi degli 8 frame, impostando in modo appropriato la mappa della MMU, da sola non basta a risolvere il problema dello spazio degli indirizzi virtuali più grande della memoria fisica. Poiché abbiamo solo 8 frame fisici, solo otto delle pagine virtuali sono mappate sulla memoria fisica. Le altre, mostrate nella figura con una X, non sono mappate. Nell'hardware, un bit presente/assente tiene traccia di quali pagine sono presenti fisicamente in memoria.

Che cosa accade se, per esempio, il programma fa riferimento a indirizzi non mappati usando l'istruzione `MOV REG,32780` che è il byte 12 all'interno della pagina virtuale 8 (che parte da 32768)? La MMU rileva che la pagina non è mappata (è contrassegnata da una X nella figura) e causa una trap della CPU verso il sistema operativo. Questa trap è chiamata page fault (errore di pagina).

Il sistema operativo preleva un frame poco utilizzato e ne scrive il contenuto su disco (se non è già presente). Prende poi la pagina alla quale è stato appena fatto riferimento e la pone nel frame appena liberato, cambia la mappa e riavvia l'istruzione che era in trap.

MMU

Guardiamo adesso all'interno della MMU per vedere come funziona e perché abbiamo scelto di usare una dimensione di pagina che sia una potenza di 2.

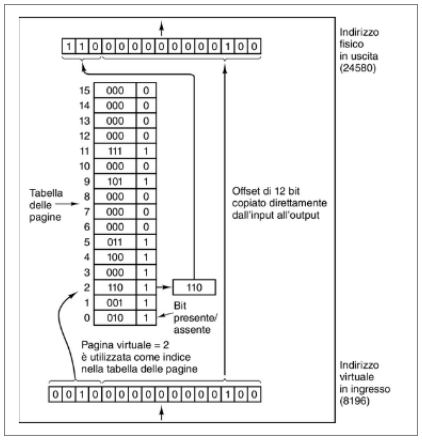

Figure 2: MMU

Nella figura viene mostrato un esempio di un indirizzo virtuale, 8196 (0010000000000100 in binario), mappato usando la mappa della MMU mostrata in figura. L'indirizzo virtuale di 16 bit in ingresso è suddiviso in un numero di pagina di 4 bit e un offset di 12 bit. Con 4 bit per il numero di pagina (bit usati per la conversione da indirizzo virtuale a indirizzo fisico), possiamo avere 16 pagine (2^4 pagine) e con 12 bit di offset (bit usati per scorrere all'interno della pagina) possiamo indirizzare 4096 byte (4Kb) per pagina, ovvero 4Kb * 16 pagine = 64Kb per ogni processo.

Il numero di pagina è usato come indice nella tabella delle pagine che porta al numero di frame corrispondente alla pagina virtuale. Se il bit presente/assente è 0, avviene una trap al sistema operativo. Se il bit è 1, il numero di frame trovato nella tabella delle pagine viene copiato nei tre bit più significativi del registro di output, insieme all'offset di 12 bit che è copiato senza modifiche dall'indirizzo virtuale in arrivo. Insieme formano un indirizzo fisico di 15 bit. Il registro di output è poi posto sul bus di memoria come indirizzo fisico di memoria.

STRUTTURA DI UNA VOCE NELLA TABELLA DELLE PAGINE

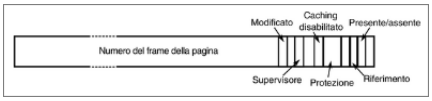

Figure 3: Struttura di una voce nella tabella delle pagine

Nella figura è mostrato l'esempio di una voce della tabella delle pagine. La dimensione cambia a seconda del computer, ma quella usuale in un generico computer odierno è di 64 bit. Il campo più importante è il numero del frame (frame number): l'obiettivo della mappatura delle pagine è ottenere questo valore. Se la dimensione della pagina è 4 KB, per il numero del frame sono necessari solo i 52 bit più significativi, e rimangono 12 bit per codificare altre informazioni sulla pagina. Ad esempio, il bit Presente/assente indica se la voce è valida e possa essere usata. Se questo bit è 0, la pagina virtuale cui appartiene la voce non è effettivamente in memoria. L'accesso alla voce della tabella delle pagine con questo bit impostato a 0 causa un page fault.

I bit Protezione specificano quali tipi di accesso sono consentiti. Nella forma elementare questo campo contiene 1 bit, con 0 che significa lettura/scrittura e 1 per la sola lettura. Un'impostazione più sofisticata ha 3 bit, ogni bit per consentire lettura, scrittura ed esecuzione della pagina. Il bit Supervisor è in qualche modo correlato, e indica se la pagina sia accessibile soltanto al codice con privilegi, ovvero al sistema operativo (o supervisore) oppure anche ai programmi utente. Qualsiasi tentativo di accesso a una pagina supervisore da parte di un programma utente causa un page fault.

I bit Modificato e Riferimento tengono traccia dell'uso della pagina. Quando viene scritta una pagina, l'hardware imposta automaticamente il bit Modificato. Questo bit è valorizzato quando il sistema operativo decide di riutilizzare un frame. Se la sua pagina è stata modificata (cioè è "sporca") deve essere riscritta sulla memoria non volatile. Se non è stata modificata (cioè è "pulita") può essere abbandonata, poiché la copia su disco o SSD è ancora valida. Talvolta il bit è chiamato dirty bit poiché riflette lo stato della pagina.

Il bit Riferimento è impostato ogni qualvolta si faccia riferimento alla pagina, sia in lettura sia in scrittura. Serve per aiutare il sistema operativo a scegliere una pagina da "sfrattare" quando si verifica un page fault; le pagine inutilizzate sono più "sfrattabili" di quelle usate, e questo bit gioca un ruolo importante in molti degli algoritmi di sostituzione delle pagine che studieremo in seguito.

Per un processo, l'indirizzo in memoria della "sua" tabella delle pagine è scritto nel registro Page Table Base Register.

COME VELOCIZZARE LA PAGINAZZIONE

Problema

1. La mappatura dall'indirizzo virtuale all'indirizzo fisico deve essere veloce;
2. Anche se lo spazio virtuale degli indirizzi è enorme, la tabella delle pagine non deve essere troppo grande.

Soluzioni

1. **Tabella delle pagine in Registri Hardware**: All'avvio di un processo, il sistema operativo carica i registri con la tabella delle pagine del processo, presa da una copia che tiene in memoria. Durante l'esecuzione del processo non sono necessari altri riferimenti alla memoria per la tabella delle pagine. Il vantaggio di questo metodo è che è semplice e non richiede riferimenti alla memoria durante la mappatura. Lo svantaggio è che è insopportabilmente dispendioso se la tabella delle pagine è estesa; nella maggior parte dei casi non è praticabile. Un altro è che dover caricare l'intera tabella delle pagine a ogni cambio di contesto compromette le prestazioni.

2. **Tabella delle pagine in Memoria RAM**: Questo design consente di cambiare la mappatura virtuale-fisica a ogni cambio di contesto ricaricando un solo registro. Lo svantaggio è che naturalmente richiede uno o più riferimenti di memoria per la lettura della tabella delle pagine durante l'esecuzione di ogni istruzione, rendendola molto lenta.

TLB (Translation Lookaside Buffer)

Ogni istruzione richiede l'accesso alla memoria per prelevare l'istruzione stessa e un ulteriore accesso per la tabella delle pagine. Questo raddoppio degli accessi alla memoria riduce le prestazioni di metà.

I progettisti di computer hanno escogitato una soluzione basata sull'osservazione che la maggior parte dei programmi tende a fare un gran numero di riferimenti a un piccolo numero di pagine e non il contrario. Dunque solo una piccola parte delle voci della tabella delle pagine viene letta frequentemente; il resto è poco utilizzato.

La soluzione è dotare i computer di un piccolo dispositivo hardware per mappare gli indirizzi virtuali sugli indirizzi fisici senza passare dalla tabella delle pagine. Il dispositivo è chiamato TLB (Translation Lookaside Buffer) o qualche volta anche memoria associativa. Si trova abitualmente all'interno della MMU e consiste di un numero ridotto di voci, otto in questo caso, ma raramente più di 256. Ciascuna voce contiene informazioni riguardo una pagina, tra cui il numero di pagina virtuale, un bit impostato quando la pagina viene modificata, il codice di protezione (i permessi di lettura, scrittura ed esecuzione) e il frame fisico in cui si trova la pagina. Questi campi hanno una corrispondenza uno-a-uno con i campi nella tabella delle pagine, eccetto che per il numero di pagina virtuale, che non è necessario nella tabella delle pagine. Un altro bit indica se la voce è valida (cioè in uso) o no.

Valido	*Pagina virtuale*	*Modificato*	*Protezione*	*Frame*
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Figure 4: TLB con 8 voci

Funzionamento della TLB

Quando un indirizzo virtuale viene presentato alla MMU per la traduzione, l'hardware prima guarda se il suo numero di pagina virtuale è presente nel TLB confrontandolo simultaneamente (cioè in parallelo) con tutte le voci. Questa operazione richiede un hardware specializzato, presente in tutte le MMU dotate di TLB. Se trova un riscontro valido e l'accesso non viola i bit di protezione, il frame è prelevato direttamente dal TLB, senza andare alla tabella delle pagine in memoria. Se il numero di pagina virtuale è presente nel TLB, ma l'istruzione prova a scrivere su una pagina di sola lettura, si genera un errore di protezione (Segmentation Fault).

È interessante ciò che accade quando il numero di pagina virtuale non è nel TLB. La MMU rileva il TLB miss (assenza dal TLB) ed esegue una normale ricerca sulla tabella delle pagine. Quindi sfratta una delle voci dal TLB e la rimpiazza con la voce della tabella delle pagine appena trovata, così se quella pagina viene riutilizzata a breve, la seconda volta si avrà una TLB hit (presenza nel TLB) invece di un TLB miss. Quando una voce è eliminata dal TLB, il bit Modificato viene copiato di nuovo nella voce della tabella delle pagine nella memoria. Gli altri valori sono già lì, eccetto il bit Riferimento. Quando il TLB viene caricato dalla tabella delle pagine, tutti i campi vengono presi dalla memoria.

La modifica dei bit di accesso ad una pagina richiedono l'aggiornamento del TLB. Per garantire la coerenza, la voce corrispondente nel TLB viene eliminata o aggiornata.

Alcune architetture RISC come SPARC, MIPS e HP PA gestiscono le voci del TLB a livello software. Quando si verifica un TLB miss, invece di far andare la MMU alla tabella delle pagine per cercare e prelevare il necessario riferimento alla pagina, viene semplicemente generato un errore di TLB e il problema passa al sistema operativo. Il sistema deve trovare la pagina, rimuovere una voce dal TLB, inserirne una nuova e riavviare l'istruzione in errore. Naturalmente tutto questo deve avvenire con una manciata di istruzioni, poiché i TLB miss sono molto più frequenti dei page fault, ed è importante capire il perché.

Quanti tipi ci sono di TLB MISS e quali sono?

1. **Soft Miss:** la pagina di riferimento non è nel TLB ma è nella memoria. In questo caso basta aggiornare il TLB, non serve alcun I/O da disco o SSD. Solitamente per trattare un soft miss ci vogliono 10-20 istruzioni macchina che possono essere completate in pochi nanosecondi.

2. **Hard Miss:** la pagina richiesta non è in memoria (e ovviamente nemmeno nel TLB). Per prelevare la pagina serve un accesso al disco o all'SSD, nell'ordine dei millisecondi a seconda del tipo di memoria non volatile. Un hard miss è milioni di volte più lento di un soft miss.

Cercare la mappatura nella gerarchia delle tabelle delle pagine è un'operazione che prende il nome di *page table walk* (passeggiata per la tabella delle pagine).

Un miss non è solamente soft o hard; alcuni miss sono leggermente più soft (o leggermente più hard) rispetto ad altri. Si supponga, per esempio, che la page walk non trovi la pagina all'interno della tabella delle pagine del processo e il programma incorra in un errore di pagina. Le possibilità sono tre.

1. La pagina potrebbe essere in memoria, ma non nella tabella delle pagine di questo processo, magari perché è stata caricata dalla memoria non volatile da parte di un altro processo. In questo caso non è necessario accedere nuovamente alla memoria non volatile, ma è sufficiente mappare la pagina in maniera appropriata nella tabella delle pagine. Questo è un miss abbastanza soft, chiamato anche *minor page fault*.

2. Un *major page fault* si può verificare se la pagina dev'essere caricata dalla memoria non volatile.

3. Il programma abbia semplicemente avuto accesso a un indirizzo non valido e non sia necessario aggiungere alcuna mappatura al TLB. In questo caso, il sistema operativo solitamente termina il programma con un segmentation fault.

DIMENSIONE DELLE TABELLE DELLE PAGINE

Problema: con 32 bit si hanno 2^{20} pagine per ogni processo, con 64 bit si hanno 2^{32} pagine per ogni processo, che porta ad uno spreco di memoria perché si ha un enorme spazio di indirizzi e un enorme tabella delle pagine che bisogna tenere in memoria.

Soluzione: Tabelle Multi livello. Un semplice esempio è illustrato nella figura sotto. Abbiamo un indirizzo virtuale a 32 bit partizionato in un campo PT1 a 10 bit, un campo PT2 a 10 bit e un campo Offset a 12 bit. Poiché gli offset sono 12 bit, le pagine sono da 4 KB e ce ne sono in totale 2^{20}. Il segreto del metodo della tabella delle pagine multilivello sta nell'evitare di mantenere tutte le tabelle delle pagine in memoria per tutto il tempo. In particolare quelle non necessarie dovrebbero essere messe da parte (viene creata una nuova tabella quando è necessario).

CR3 register: Registro speciale per puntare al vertice della gerarchia delle tabelle di pagina.

Con 64 bit si hanno tabelle di pagine a 4 livelli. I processori oggi utilizzano tutte le 512 voci di ciascuna tabella, arrivando a una quantità di memoria indirizzabile pari a $2^9 \times 2^9 \times 2^9 \times 2^9 \times 2^1 2 = 2^{48}$ byte. Si poteva aggiungere anche un altro livello, ma probabilmente i produttori hanno pensato che, almeno per il momento, 256 TB di memoria indirizzabile sarebbero stati sufficienti.

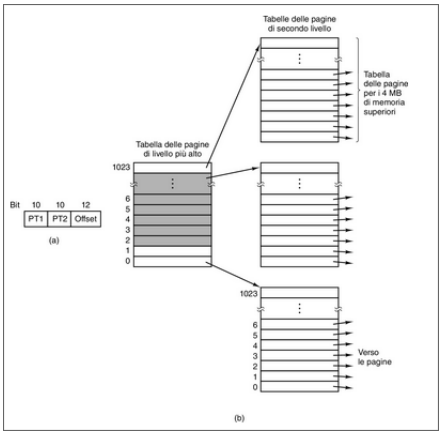

Figure 5: Tabella multi livello