Lezione 17 [11/11/22]

Paginazione

Con il metodo della paginazione la memoria viene suddivisa in blocchi di dimensione fissa

Lo spazio degli indirizzi logici di un processo può essere allocato in blocchi anche non contigui di memoria fisica

I blocchi della memoria fisica sono detti **frame**, mentre i blocchi della memoria logica sono detti **pagine**

Quando un processo è pronto per essere eseguito si caricano le sue pagine nei blocchi di memoria disponibili, prendendole dalla memoria ausiliaria, che è divisa in blocchi di dimensione fissa uguale a quella dei frame (le pagine hanno la stessa grandezza dei frame)

Si risolve il problema della frammentazione esterna (si potrà avere invece frammentazione interna)

Bisogna mantener traccia dei frame liberi: per eseguire un processo di n pagine ci sarà bisogno di n frame liberi in cui caricare il processo

Una **tabella delle pagine** verrà utilizzata per tradurre l’indirizzo logico in un indirizzo fisico

La paginazione non è altro che una forma di rilocazione dinamica: ogni indirizzo logico è associato a un indirizzo fisico dall’hardware di paginazione

La paginazione è simile all’utilizzo di una tabella di registri base (rilocazione), uno per ciascun frame di memoria

Con la paginazione si può evitare la frammentazione esterna: **qualsiasi** blocco di memoria libero si può assegnare a un processo che ne abbia bisogno
Tuttavia si può avere la frammentazione interna: i frame vengono allocati come unità. Se i requisiti di memoria non combaciano con i limiti di pagina, l'ultimo frame allocato può non essere completamente pieno

Il caso peggiore si presenta quando un processo necessita di n pagine più un byte. Anche in questo caso bisogna allocare $n+1$ pagine

Di solito, in media, con la paginazione si ha una frammentazione interna di mezza pagina per processo

Schema di traduzione degli indirizzi

Ogni indirizzo generato dalla CPU è diviso in due parti:

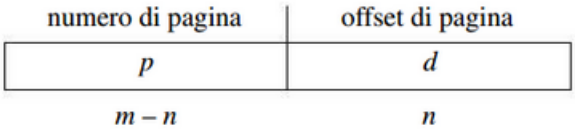
- **Numero di pagina (p)** - usato come indice per la tabella delle pagine. Essa contiene l'indirizzo base di ogni pagina nella memoria fisica
- **Scostamento di pagina (offset) (d)** - viene combinato con l'indirizzo di base per definire l'indirizzo della memoria fisica che viene inviato all'unità di memoria

La dimensione di una pagina, che è la stessa di un frame, può variare da un compilatore all'altro a seconda del tipo di hardware

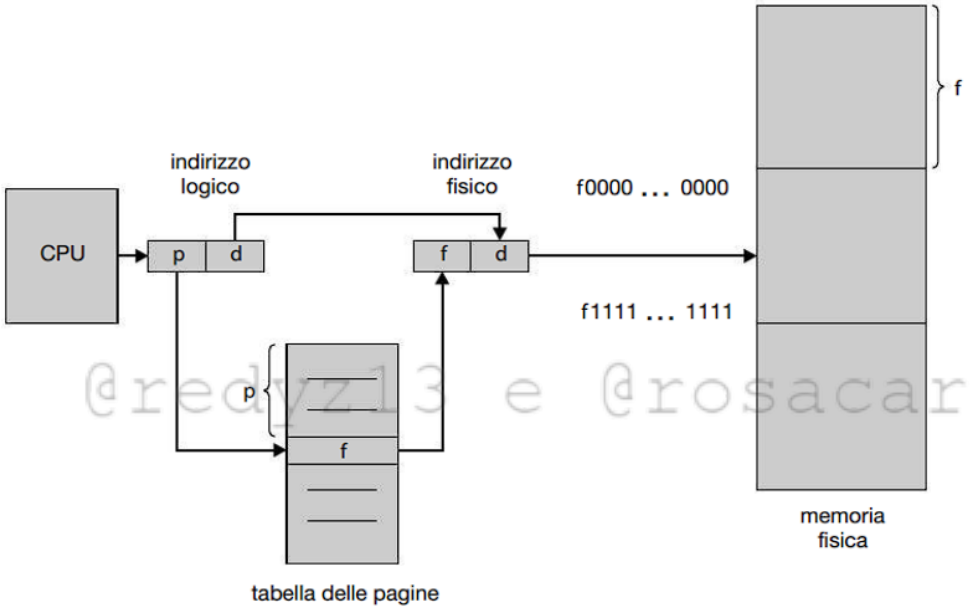
Di solito, comunque, si preferisce una potenza di 2 compresa fra 512 byte e 16 MB

Se la dimensione dello spazio degli indirizzi logici è 2^m e la dimensione di una pagina è di 2^n unità di indirizzamento, allora gli $m-n$ bit più significativi di un indirizzo logico indicano il numero di pagina, e gli n bit meno significativi indicano l'offset di pagina

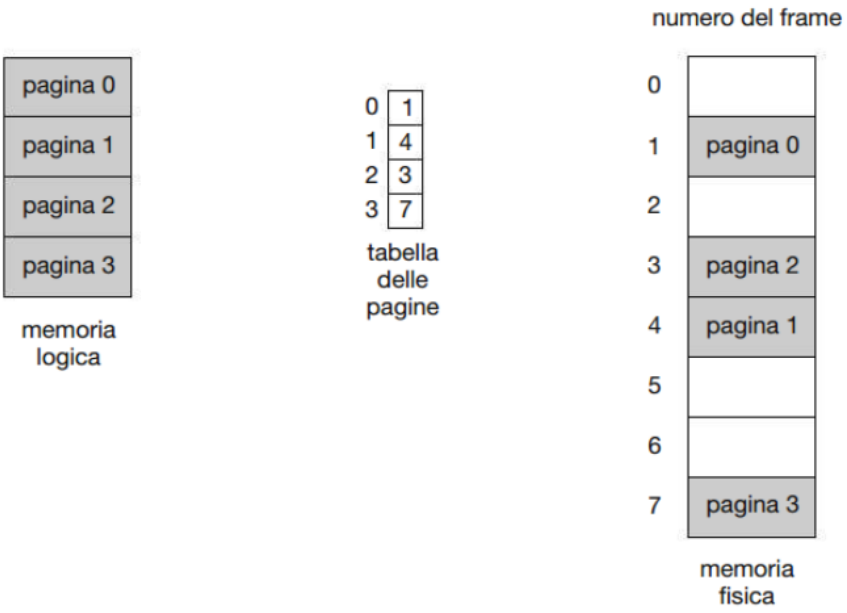
L’indirizzo logico ha quindi la forma seguente:



Architettura di paginazione:



Modello di paginazione di memoria logica e memoria fisica:



Esempio di paginazione per una memoria di 32 byte con pagine di 4 byte

Supponiamo di avere una memoria di 32 byte con ogni pagina di 4 byte (ossia 32/4 = 8 pagine)

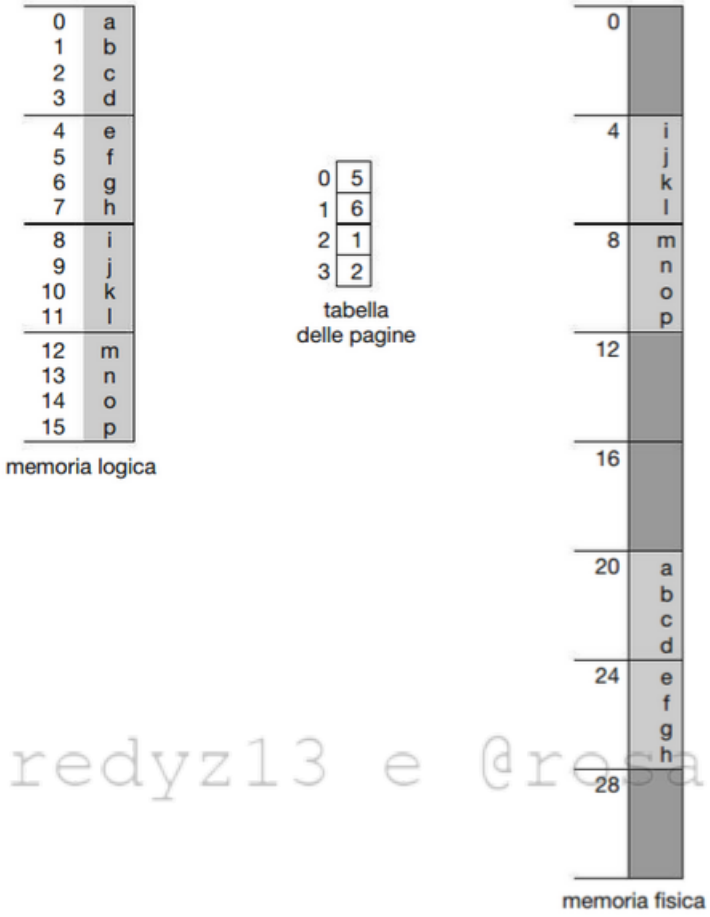
L'indirizzo logico 0 è pagina 0, offset 0. Nella tabella delle pagine, la pagina 0 si trova nel frame 5

Quindi l'indirizzo logico 0 viene mappato nell'indirizzo fisico 20 $(20 = (5 * 4) + 0)$

L'indirizzo logico 3 (pagina 0, offset 3) viene mappato nell'indirizzo fisico 23 $(23 = (5 * 4) + 3)$

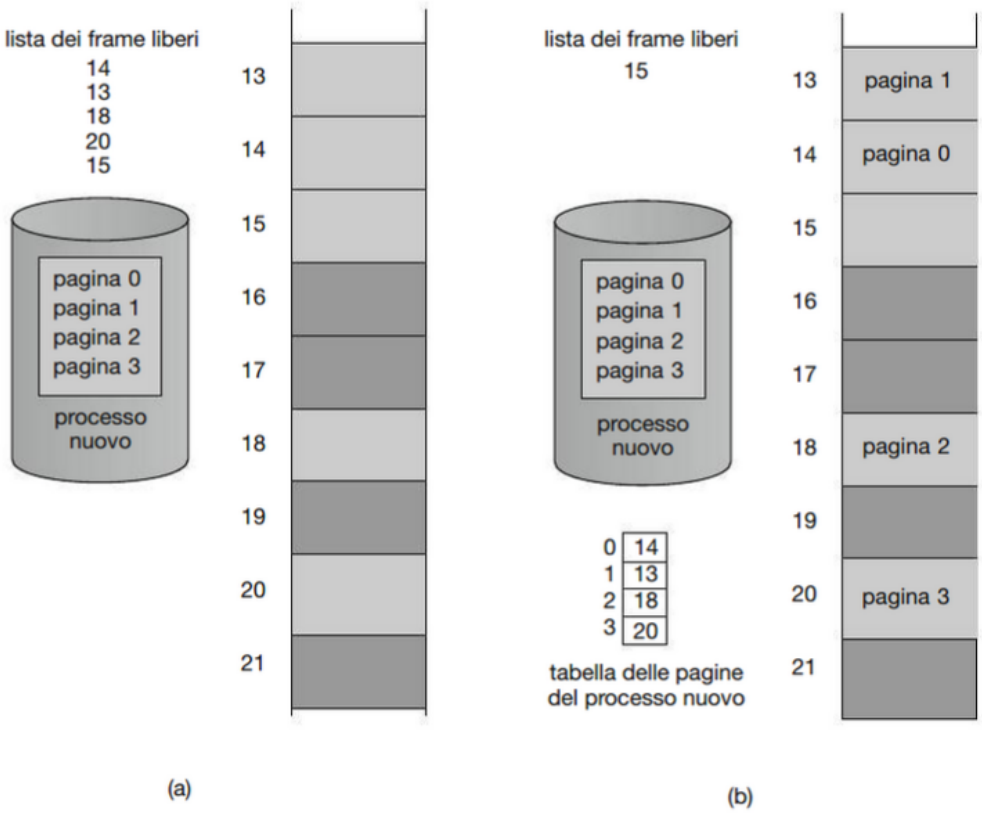
L'indirizzo logico 4 è pagina 1, offset 0; secondo la tabella delle pagine, la pagina 1 è mappata nel frame 6. L'indirizzo 4 viene mappato nell'indirizzo fisico 24 $(24 = (6 * 4) + 0)$

L'indirizzo logico 13 viene mappato nell'indirizzo fisico 9



Blocchi di memoria liberi:

Prima dell’assegnazione Dopo l’assegnazione



Quando si deve eseguire un processo, si esamina la sua dimensione espressa in pagine. Poiché ogni pagina del processo necessita di un blocco di memoria, se il processo richiede n pagine, devono essere disponibili almeno n blocchi di memoria

Si carica la prima pagina del processo in uno dei blocchi di memoria assegnati e si inserisce il numero del blocco di memoria nella tabella delle pagine relativa a quel processo in questione

La pagina successiva si carica in un altro blocco di memoria e, anche in questo caso, s’inserisce il numero del blocco di memoria nella tabella delle pagine

Tabella dei frame - contiene un elemento per ogni frame, indicando se è libero oppure assegnato e, se è assegnato, a quale pagina di quale processo o di quali processi. Permette al S.O. di essere informato sui dettagli dell'allocazione

Il S.O. conserva una copia della tabella delle pagine per ciascun processo

Questa copia viene usata anche dal dispatcher per impostare l'hardware di paginazione quando a un processo sta per essere assegnata la CPU

La paginazione fa quindi aumentare la durata dei cambi di conteso

Implementazione della tabella delle pagine (hardware paginazione)

Ogni S.O. ha il proprio metodo per memorizzare le tabelle delle pagine

Alcuni memorizzano una tabella per ogni processo: il PCB contiene un puntatore alla tabella delle pagine. Per avviare un processo il dispatcher ricarica i registri utente e imposta i corretti valori della page table hardware, usando la tabella delle pagine presente in memoria e relativa al processo

L'implementazione hardware della tabella delle pagine si realizza nel caso più semplice tramite uno specifico insieme di **registri** che il dispatcher ricarica come ricarica gli altri

La tabella delle pagine è mantenuta della memoria centrale

Il **registro di base della tabella delle pagine (page-table base register - PTBR)** punta alla tabella delle pagine. Il cambio delle tabelle delle pagine richiede solo di modificare questo registro

Il **registro di lunghezza della tabella delle pagine (page-table length register - PTLR)** indica le dimensioni della tabella delle

pagine

In questo schema l’accesso ad ogni dato o istruzione necessita di due accessi alla memoria, uno per la tabella delle pagine ed uno per il dato o istruzione

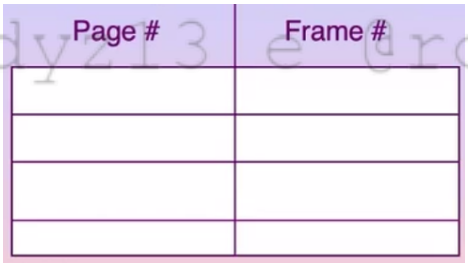
Il problema dei due accessi alla memoria può essere risolto attraverso l’utilizzo di una memoria cache speciale, molto veloce, detta **memoria associativa (associative memory** or **translation look-aside buffers - TLBs)**

Memoria Associativa (TLB)

Ogni registro è formato da una chiave e un valore

Quando ai registri associativi si presenta un elemento, viene confrontato con tutte le chiavi contemporaneamente, riducendo il tempo di ricerca (questa soluzione presenta uno svantaggio legato al tipo di hardware, molto costoso)

Memoria associativa, ricerca parallela:



La TLB contiene una piccola parte della tabella delle pagine: quando la CPU genera un indirizzo logico, si presenta il suo numero di pagina alla TLB. Se tale numero è presente, il corrispondente numero del frame è immediatamente disponibile e si usa per accedere alla memoria

Traduzione dell'indirizzo (A', A'')

- Se A' (chiave) è in un registro associativo, il numero di frame (A'') (valore) si ottiene tramite memoria associativa con una ricerca parallela (insieme alle istruzioni della CPU)
- Altrimenti il numero di frame si ottiene dalla tabella delle pagine in memoria
- **Insuccesso della TLB (TLB miss)**- Quando nella TLB non è presente il numero della pagina
  In questi casi si consulta la tabella delle pagine in memoria (quest'operazione viene effettuata automaticamente a livello hardware o tramite un interrupt al S.O.)

Se la TLB è piena di elementi, se ne deve scegliere uno da sostituire

Alcuni elementi dalle TLB risultano **vincolati**: non si possono rimuovere dalla TLB

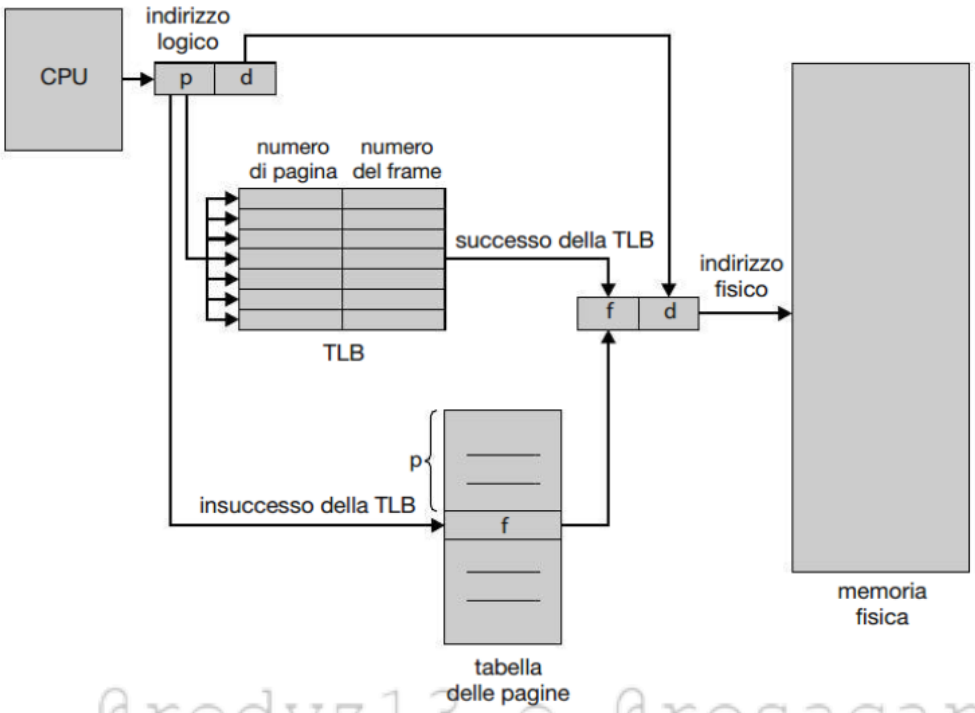
Alcuni TLB memorizzano gli **identificatori dello spazio d'indirizzi** (**address space identifier, ASID)** in ciascun elemento della TLB: identifica univocamente ciascun processo e si usa per fornire al processo corrispondente la protezione del suo spazio di indirizzi

La TLB si assicura che l'ASID per il processo attualmente in esecuzione corrisponda all'ASID associato alla pagina virtuale

L'ASID permette alla TLB di contenere nello stesso istante elementi di diversi processi.
Se la TLB non permette l'uso di ASID distinti, ogni volta che si seleziona una differente tabella delle pagine (a ogni cambio di contesto) si deve **cancellare (flush)** la TLB, in modo da assicurare che il successivo processo in esecuzione non faccia uso di errate informazioni di traduzione

- **Tasso di successi (hit ratio)** - percentuale di volte che un numero di pagina si trova nel TLB: dipende anche dal numero di registi associativi.

Architettura di paginazione con memoria associativa (TLB):



Protezione della memoria

Associato ad ogni frame vi è un **bit di protezione**, che normalmente si trova nella tabella delle pagine

Questo bit viene utilizzato per la protezione della memoria e determina se una pagina è di lettura e scrittura o di sola lettura

- In questo modo si evita che si vada a scrivere su pagine di sola lettura

Nel caso si tenti di scrivere su una pagina di sola lettura viene generato un **trap** dell'hardware al sistema operativo

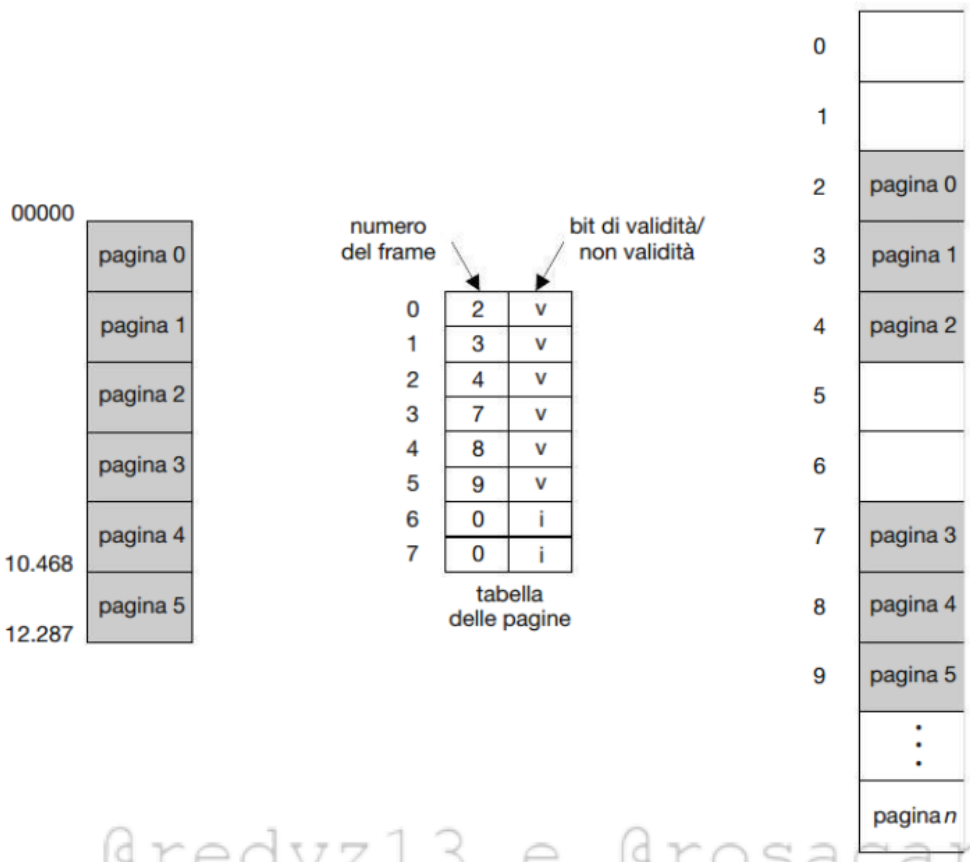
Un ulteriore bit, detto **bit di validità** viene associato a ciascun elemento della tabella delle pagine

Quando è posto a **valido** indica che la pagina associata è nello spazio di indirizzi logici del processo e pertanto è una pagina a cui il processo può legalmente accedere

In caso è posto a **invalido** allora la pagina non è nello spazio degli indirizzi logici del processo

Il PTLR viene confrontato ad ogni indirizzo logico per verificare che si trovi nell'intervallo valido per il processo

Bit di validità in una tabella delle pagine:



Struttura della tabella delle pagine

La maggior parte dei moderni computer supporta uno spazio di indirizzi logici estremamente grande

In un ambiente di questo tipo la stessa tabella delle pagine finirebbe per diventare eccessivamente grande

Chiaramente, sarebbe meglio evitare di allocare la tabella delle pagine in modo contiguo in memoria centrale

Una soluzione semplice consiste nel divedere la tabella delle pagine in parti più piccole; questo risultato può essere ottenuto in molti modi differenti

- Paginazione gerarchica
- Tabella delle pagine di tipo hash
- Tabella delle pagine invertita

Paginazione gerarchica: esempio di paginazione a due livelli

Il metodo consiste nell'adottare uno schema di paginazione a più livelli, nel quale la stessa tabella delle pagine viene paginata

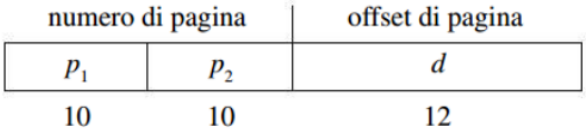
Un indirizzo logico (su una macchina a 32 bit con pagina di 4 Kb) viene diviso in:

- Un numero di pagina di 20 bit
- Uno scostamento di pagina di 12 bit

Siccome la tabella delle pagine è paginata, il numero di pagina è ulteriormente diviso in:

- Un numero di pagina di 10 bit
- Uno scostamento di pagina di 10 bit

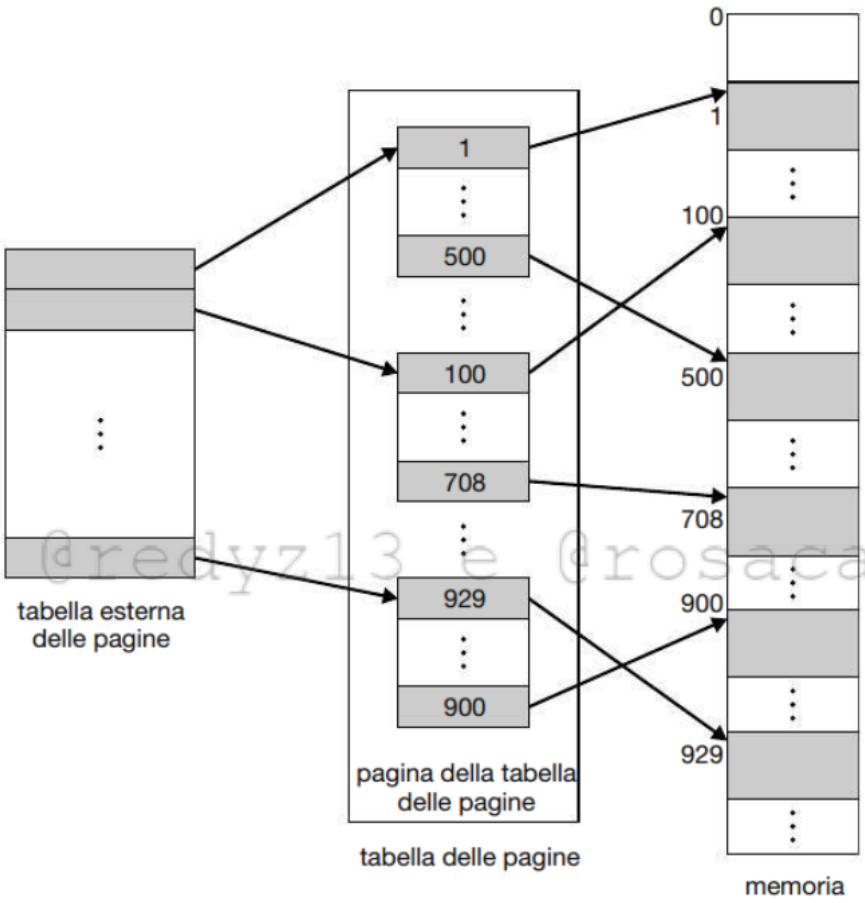
Quindi un indirizzo logico può essere così schematizzato:



Dove p_1 è un indice della tabella delle pagine di primo livello (tabella esterna delle pagine) e p_2 è lo scostamento all'interno della pagina indicata dalla tabella esterna delle pagine

Visto che la traduzione degli indirizzi avviene dalla tabella esterna delle pagine verso l'interno, questo metodo è anche noto come **tabella delle pagine ad associazione diretta (forward-mapped page table)**

Schema di una tabella delle pagine a due livelli:



Traduzione degli indirizzi per un’architettura a 32 bit con paginazione a 2 livelli:

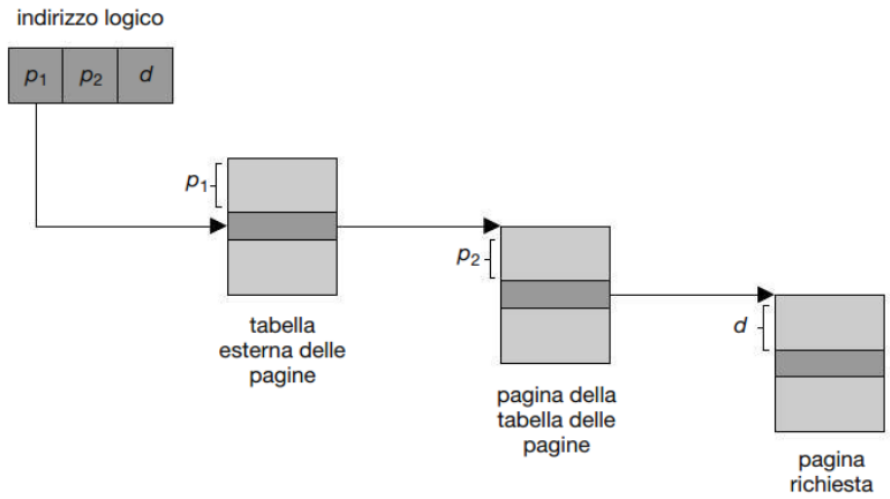


Tabella delle pagine di tipo hash

Metodo comune per gli spazi di indirizzi relativi ad architetture oltre i 32 bit

Consiste nell’impiego di una tabella delle pagine di tipo **hash** in cui l’argomento della **funzione di hashing** è il numero della pagina virtuale

Per la gestione delle collisioni, ogni elemento della tabella hash contiene una lista concatenata degli elementi (numero di pagina virtuale, indirizzo di pagina fisica, **puntatore next**) che la tabella hash fa corrispondere alla stessa locazione

Quindi si applica la funzione hash al numero della pagina virtuale e si scorre la lista relativa all’elemento della tabella hash ottenuto, fino a che non si indentifica la corrispondente pagina fisica

Una variante di questo è per i sistemi a 64 bit: la **tabella delle pagine a gruppi (clustered page table)**. La differenza è che ciascun elemento della tabella hash possiede un riferimento a pagine fisiche corrispondenti a un cluster di pagine virtuali contigue.

Le tabelle delle pagine a gruppi sono utili per gli spazi di indirizzi **sparsi**: in essi, i riferimenti alla memoria non sono contigui, ma distribuiti per tutto lo spazio di indirizzi

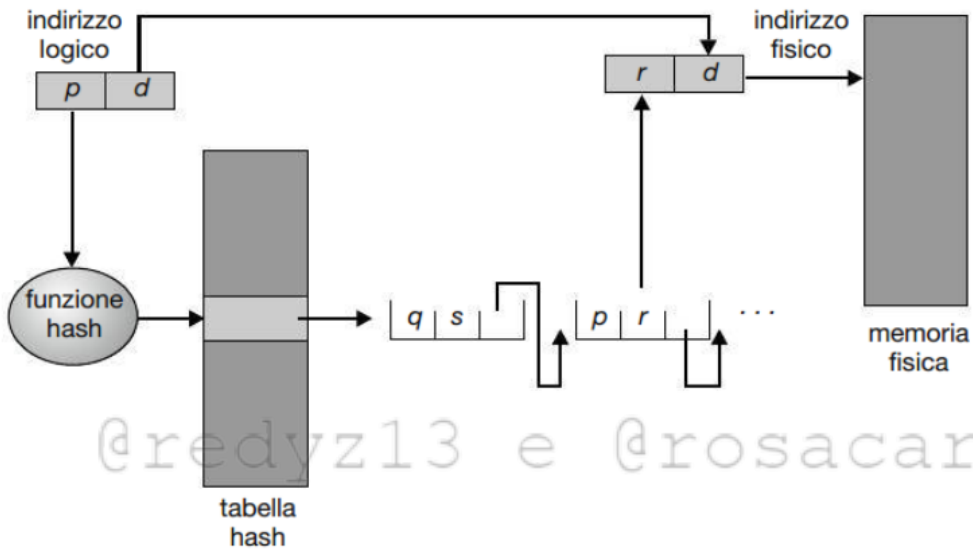


Tabella delle pagine invertita

Generalmente, si associa una tabella delle pagine ad ogni processo

Tale tabella contiene un elemento per ogni pagina virtuale che il processo sta utilizzando, ossia vi sono elementi corrispondenti ad ogni indirizzo virtuale, a prescindere dalla validità di quest'ultimo

Questa è una rappresentazione naturale della tabella poiché i processi fanno riferimento alle pagine tramite gli indirizza virtuali delle pagine stesse

Il sistema operativo deve poi tradurre questo riferimento in un indirizzo di memoria fisica

Poiché la tabella è ordinata per indirizzo virtuale, il sistema operativo può calcolare in che punto della tabella si trovi l'elemento dell'indirizzo fisico associato, e utilizzare direttamente tale valore

Uno degli inconvenienti, in questo caso, è costituito dalla dimensione di ciascuna tabella delle pagine, che può avere milioni di elementi e consumare grandi quantità di memoria fisica

Per risolvere questo problema si può fare uso della **tabella delle pagine invertita**: ha un elemento per ogni pagina reale (frame) di memoria

Ciascun elemento è quindi costituito dall'indirizzo virtuale della pagina memorizzata in quella reale locazione di memoria, con informazioni sul processo che possiede tale pagina

Quindi, nel sistema esiste solo una tabella delle pagine che ha un solo elemento per ciascuna pagina di memoria fisica

Questo richiede molto spesso la memorizzazione di un identificatore dello spazio di indirizzi in ciascun elemento della tabella delle pagine, perché essa contiene di solito molti spazi di indirizzi diversi associati alla memoria fisica: l'identificatore permette che una data pagina logica sia associata alla pagina fisica corrispondente

Ciascun indirizzo virtuale è formato dalla seguente tripla: **<id-processo, numero di pagina, offset>**

Ogni elemento della tabella delle pagine invertita è una coppia **<id-processo, numero di pagina>** (dove l'id processo assume

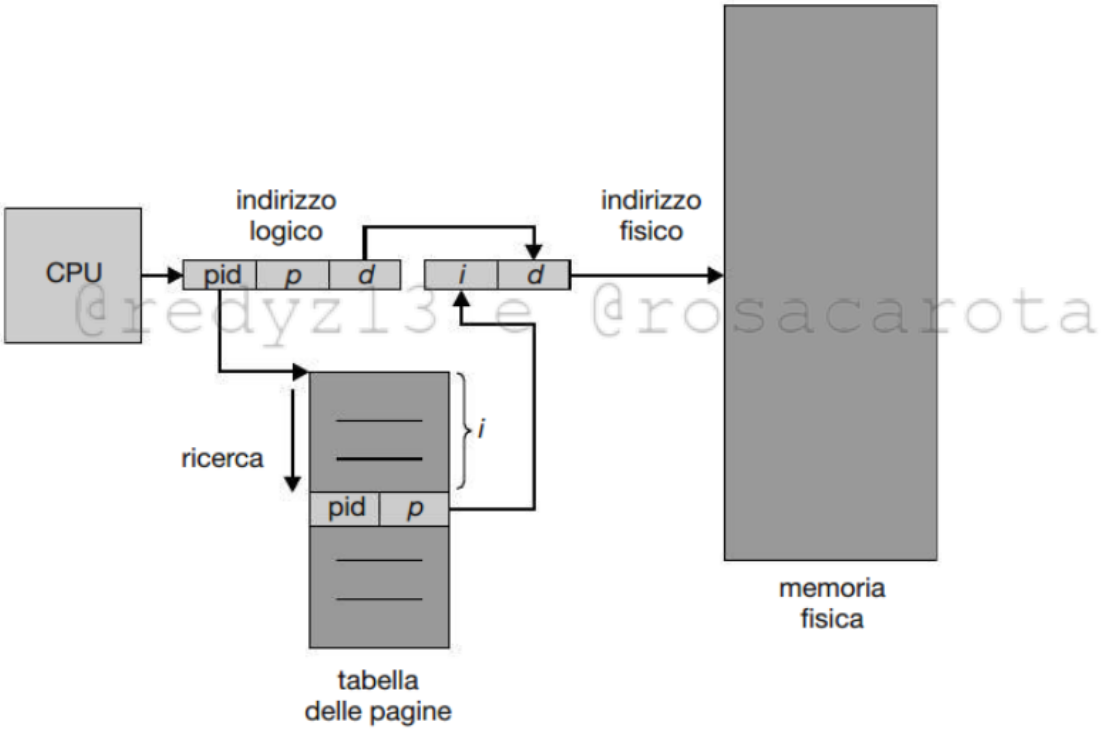
l’identificatore dello spazio di indirizzi

Quando viene effettuato un riferimento alla memoria, parte dell’indirizzo virtuale, formato da <id-processo, numero di pagina>, viene presentato al sottosistema di memoria (MMU)

Quindi viene cercata una corrispondenza nella tabella delle pagine invertita

Se tale corrispondenza viene trovata, ad esempio sull’elemento i, viene generato l’indirizzo fisico **<i, offset>**, altrimenti è stato tentato un accesso illegale ad un indirizzo

Esempio di tabella delle pagine invertita:



Sebbene questo schema riduca la quantità di memoria necessaria per memorizzare ogni tabella delle pagine, aumenta però la quantità di tempo necessaria per cercare nella tabella quando viene fatto riferimento a una pagina

La tabella delle pagine invertita è ordinata per indirizzo fisico, mentre le ricerche vengono effettuate su indirizzi virtuali quindi potrebbe risultare necessario controllare tutta la tabella

Per trovare una corrispondenza occorre cercare in tutta la tabella, e questa ricerca richiede molto tempo

Per alleviare il problema può essere utilizzata una tabella hash per limitare la ricerca a un solo, o a pochi, elementi della tabella delle pagine

Pagine condivise

Un altro vantaggio della paginazione consiste nella possibilità di **condividere** codice comune, cosa importante soprattutto in un ambiente con time-sharing

Un codice eseguibile è detto **rientrante (o puro)** se è un codice non automodificante, poiché non cambia mai durante l'esecuzione

Quindi, due o più processo possono eseguire lo stesso codice nello stesso momento

Ciascun processo dispone di una propria copia dei registri e di una memoria dove conserva i dati necessari alla propria esecuzione

- Ad es. nella memoria fisica è presente solo una copia dell'editor
- La tabella delle pagine di ogni utente fa corrispondere gli stessi blocchi di memoria contenenti l'editor, mentre le pagine dei dati sono mappate su frame diversi

Possono essere condivisi anche altri programmi di utilizzo frequente: compilatori, sistemi di finestre, database e così via

Per essere condivisibile, il codice deve essere rientrante

Esempio di pagine condivise: