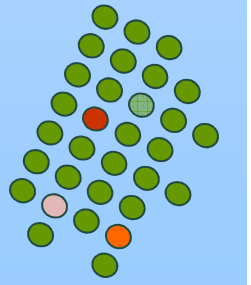
Gestione della memoria centrale

Capitolo 8 - Silberschatz

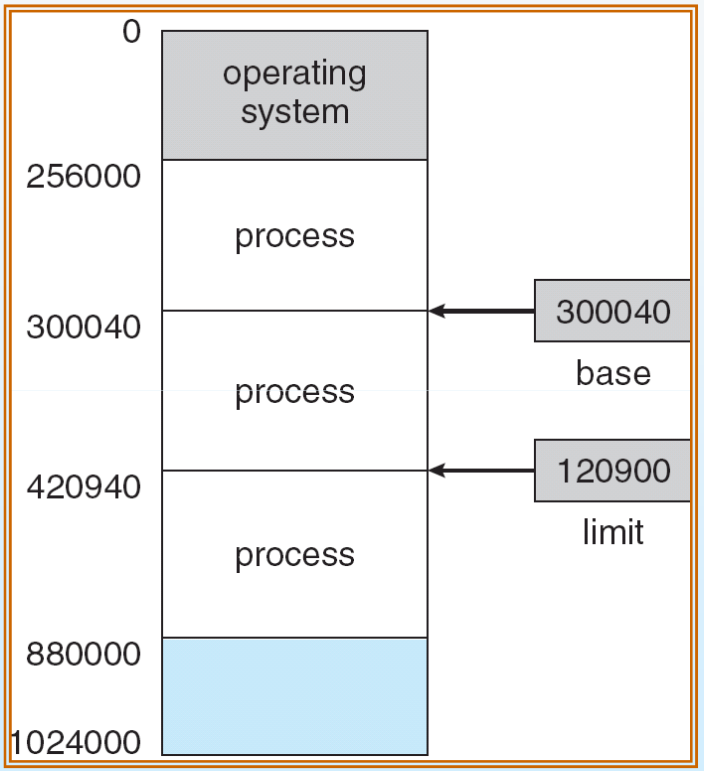


Background

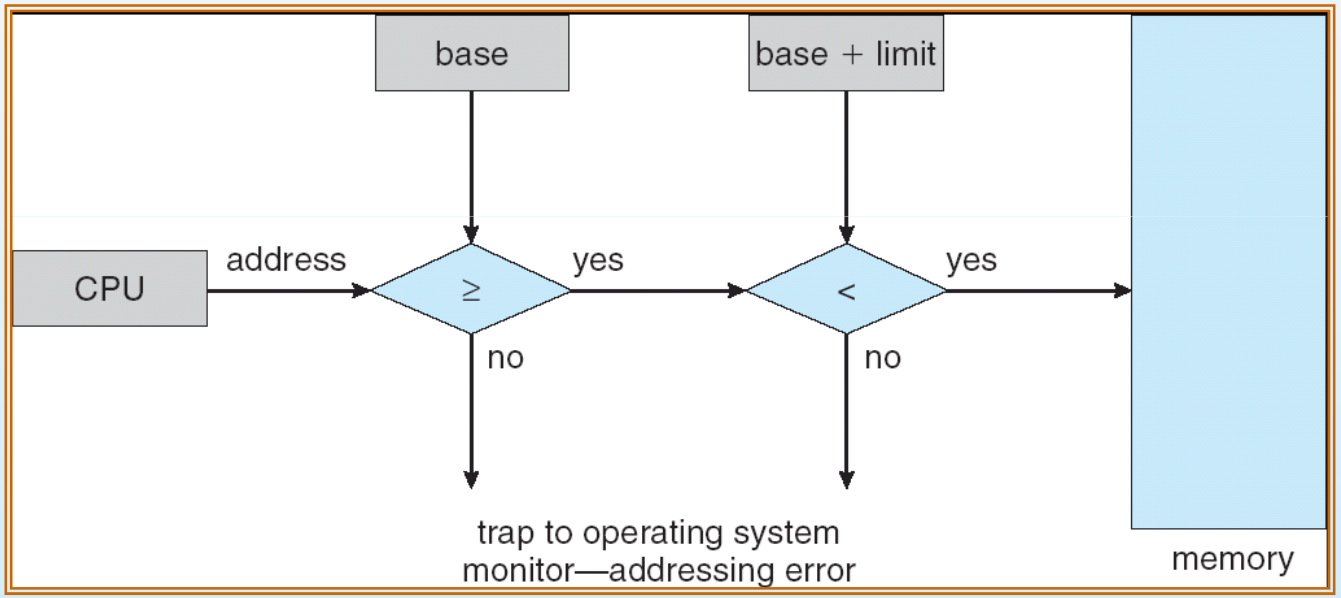
- Un programma in genere risiede su disco in forma di un file binario eseguibile e deve essere portato (dal disco) in memoria e “inserito” all’interno di un processo per essere eseguito
- La memoria principale e i registri sono le uniche memorie che la CPU può accedere direttamente
  - I dati devono essere portati in memoria prima che la CPU possa operare su di loro
- I registri sono accessibili dalla CPU nell’arco di un ciclo di clock
- L’accesso alla memoria centrale può richiedere diversi cicli
- La cache si trova tra la memoria centrale e i registri di CPU e tenta di conciliare le diverse velocità tra CPU e memoria

Registri base e limite

- Un meccanismo di protezione della memoria è richiesto per assicurare una corretta modalità di funzionamento
- Una coppia di registri **base** e **limit** definiscono lo spazio di indirizzamento logico di un processo



Protezione HW degli indirizzi tramite registri base e limit

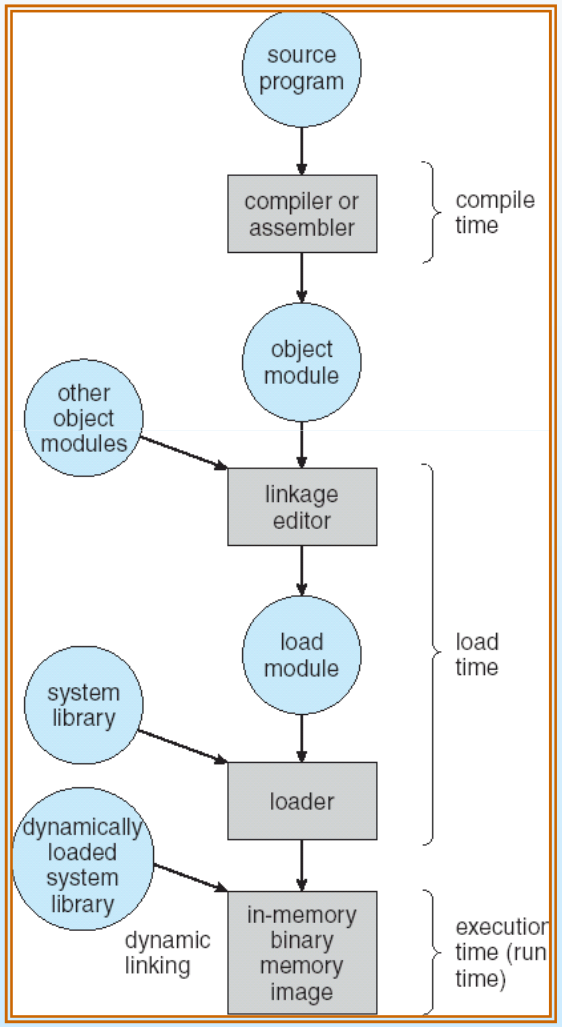


Generazione degli indirizzi di memoria

Il collegamento delle istruzioni e dei dati agli indirizzi di memoria può essere effettuato in:

- **Fase (o tempo) di compilazione**: se in fase di compilazione si conosce dove il processo risiederà in memoria, allora si può generare un codice assoluto.
- **Fase di caricamento**: il compilatore genera un codice rilocabile. Il caricatore associa indirizzi rilocabili ad indirizzi di memoria.
- **Fase (o tempo) di esecuzione**: se il processo può venire spostato, durante l'esecuzione, da un segmento di memoria a un altro, allora il collegamento deve essere ritardato fino al momento dell'esecuzione. Necessita di supporto hardware.

Programma utente



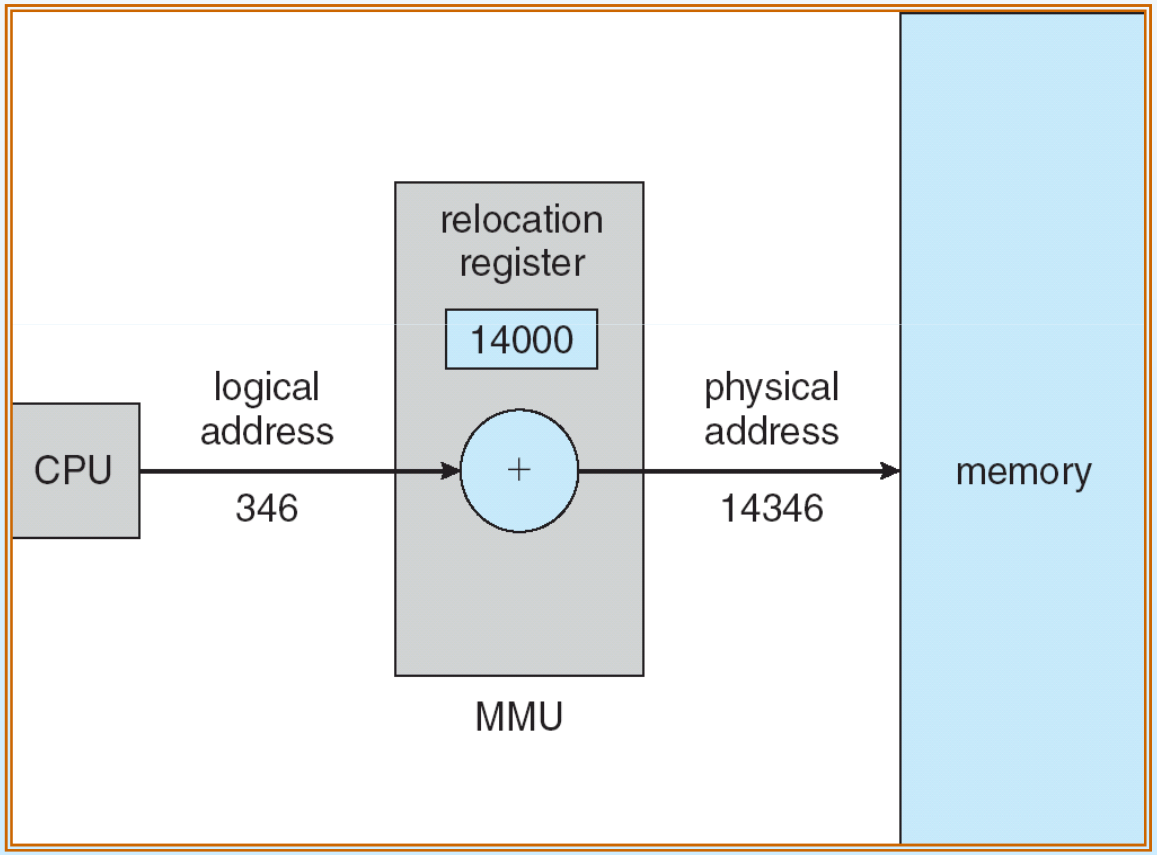
Spazio di indirizzamento logico e spazio di indirizzamento fisico

- Il concetto di *spazio di indirizzamento logico* collegato ad un separato spazio di indirizzamento fisico è basilare per un'adeguata gestione della memoria.
  - Indirizzo logico – indirizzo generato dalla CPU; anche definito come indirizzo virtuale.
  - Indirizzo fisico – indirizzo visto dalla memoria.
- Fase di compilazione e di caricamento: indirizzi logici e fisici identici.
- Fase di esecuzione: indirizzi logici e fisici diversi.

Unità di gestione della memoria centrale
(Memory Management Unit)

- Dispositivo hardware che realizza la trasformazione degli indirizzi virtuali in indirizzi fisici in fase di esecuzione.
- La MMU, ad ogni indirizzo generato da un processo, aggiunge il valore contenuto nel registro di rilocazione (= registro base) nel momento in cui l'indirizzo è trasmesso alla memoria.
- Il programma utente interagisce con gli indirizzi logici; non vede mai gli indirizzi fisici reali.

Rilocazione Dinamica tramite un registro di rilocazione



Caricamento dinamico

- Una procedura non è caricata finché non è chiamata.
  - Vengono mantenute in memoria secondaria in un formato rilocabile
- Migliore utilizzo dello spazio di memoria; una procedura inutilizzata non viene mai caricata.
- Utile quando sono necessarie grandi quantità di codice per gestire situazioni che si presentano raramente. Gestione errori.
- Non richiede un supporto speciale da parte del sistema operativo. Procedure di libreria.

Collegamento dinamico

- Il collegamento è posposto fino al tempo di esecuzione.
- Una piccola parte di codice (stub) specifica come individuare la procedura desiderata residente in memoria.
- Lo stub è rimpiazzato con l'indirizzo della procedura che viene eseguita.
- Il sistema operativo deve controllare che la procedura necessaria sia nello spazio di indirizzamento del processo.
- Il collegamento dinamico è particolarmente utile per l'implementazione delle librerie dei linguaggi.

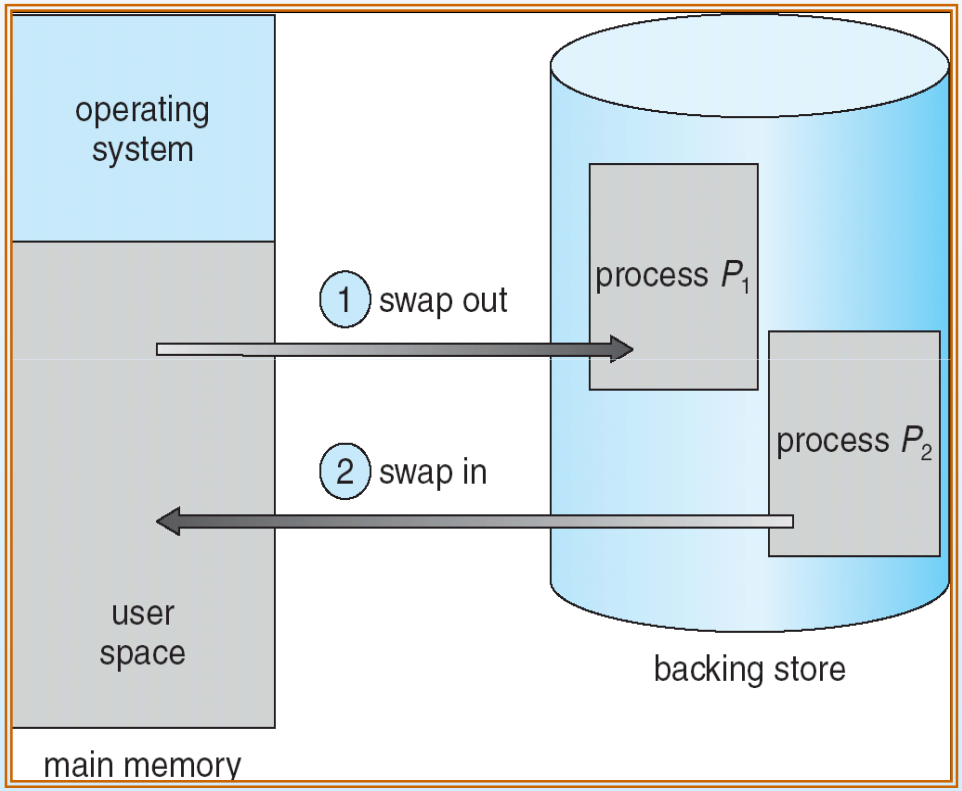
Swapping

- Un processo può essere temporaneamente spostato e poi riportato in memoria centrale (swap out, swap in).
- Memoria temporanea (disco): memorizza le copie delle immagini di memoria centrale per i processi, e fornisce accesso diretto a queste immagini.
- Roll out, roll in – variante dello swapping usata per algoritmi di schedulazione basati su priorità.
- La maggior parte del tempo di swap è tempo di trasferimento; il tempo totale di trasferimento è direttamente proporzionale alla quantità di memoria interessata.
- Versioni modificate di swapping si trovano in molti sistemi operativi (ad esempio UNIX, Linux, e Windows).

Veduta schematica dello Swapping

Svantaggi :

- Tempo di trasferimento
- Problemi con I/O
- Riposizionamento nello spazio di mem



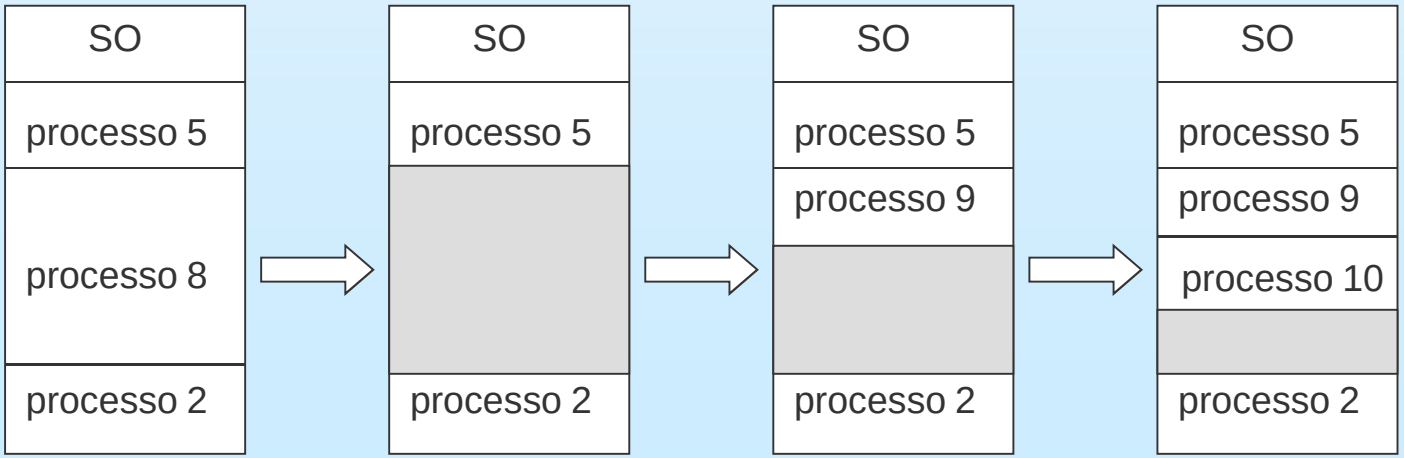
Allocazione contigua di memoria

- La memoria centrale è normalmente divisa in due partizioni:
  - Sistema operativo residente, collocato nella memoria bassa con il vettore degli interrupt.
  - Processi dell'utente collocati nella memoria alta.
- Con l'allocazione contigua, ciascun processo è contenuto in una singola sezione contigua di memoria

Allocazione contigua di memoria

Allocazione con partizioni multiple:

- blocchi di memoria libera di varie dimensioni (*hole*) sono sparsi nella memoria centrale.
- Quando un processo arriva, il sistema cerca nell'insieme un blocco libero abbastanza grande per questo processo.
- Il sistema operativo controlla le informazioni su:
  - a) partizioni allocate
  - b) partizioni libere (hole)



Problema dell'allocazione dinamica della memoria centrale

Come soddisfare una richiesta di dimensione n da una lista di blocchi liberi?

- **First-fit**: assegna il primo blocco libero abbastanza grande per contenere lo spazio richiesto.
- **Best-fit**: assegna il più piccolo blocco libero abbastanza grande.
- **Worst-fit**: assegna il più grande blocco libero.

Sia il metodo first-fit sia quello best-fit sono migliori del metodo worst-fit in termini di tempo e di utilizzo della memoria centrale.

Frammentazione

- **Frammentazione esterna** – c'è abbastanza spazio totale di memoria centrale per soddisfare una richiesta, ma gli spazi disponibili non sono contigui

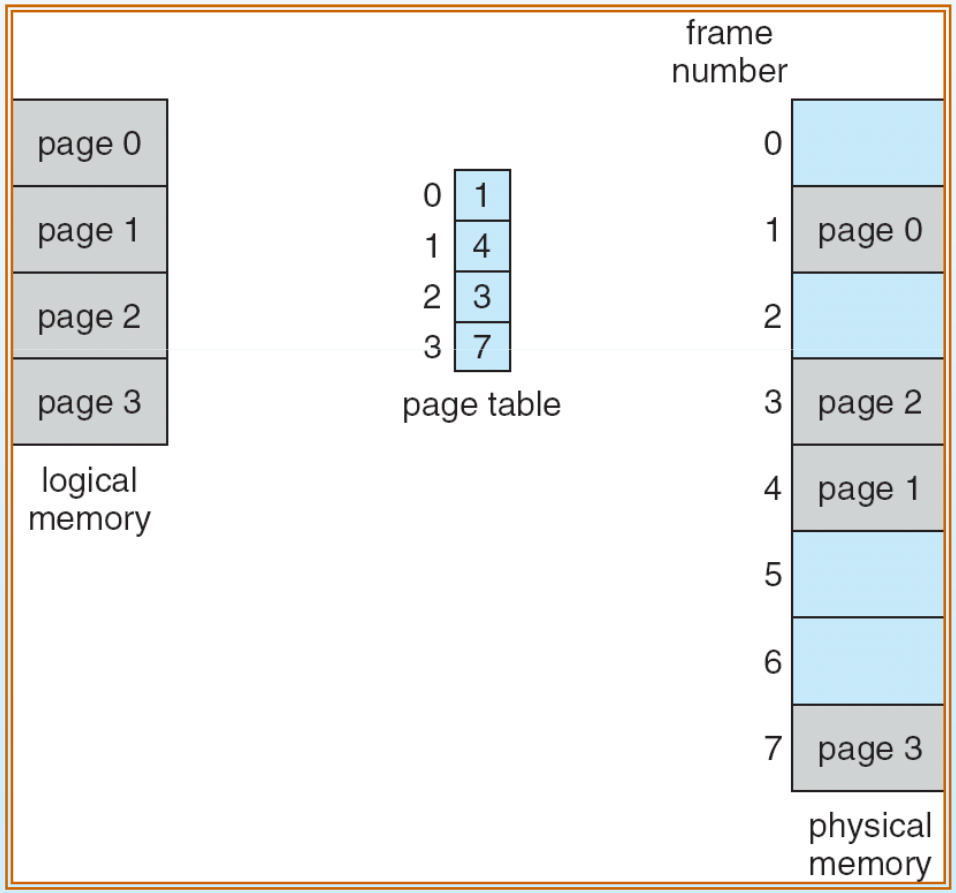
- **Frammentazione interna** – la memoria centrare allocata ad un processo può essere un po' più grande di quella richiesta.

- Ridurre la frammentazione esterna attraverso la compattazione:
  - Fondere i contenuti della memoria centrale per avere tutta la memoria centrale libera in un grande blocco.
  - La compattazione è possibile *solo se* la rilocazione è dinamica ed è fatta al momento dell'esecuzione.
  - Problema dei dispositivi di I/O:
    - Un processo è fermo in memoria mentre compie I/O.
    - Fare I/O soltanto in buffer del SO.

Paginazione

- Lo spazio degli indirizzi fisici può essere non contiguo.
- Suddivide la memoria fisica in blocchi (**frame**) (la dimensione è una potenza di 2, e.g., 512 byte, 8192 byte …).
- Divide la memoria logica in blocchi delle stesse dimensioni chiamati **pagine**.
- Mantiene traccia di tutti i frame liberi.
- Per eseguire un processo di n pagine, bisogna trovare n frame liberi e caricare le pagine nei frame.
- Impostare una tabella delle pagine per tradurre gli indirizzi logici in indirizzi fisici.

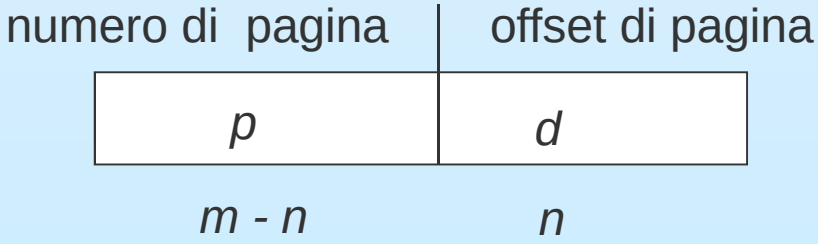
Paginazione: pagine e frame



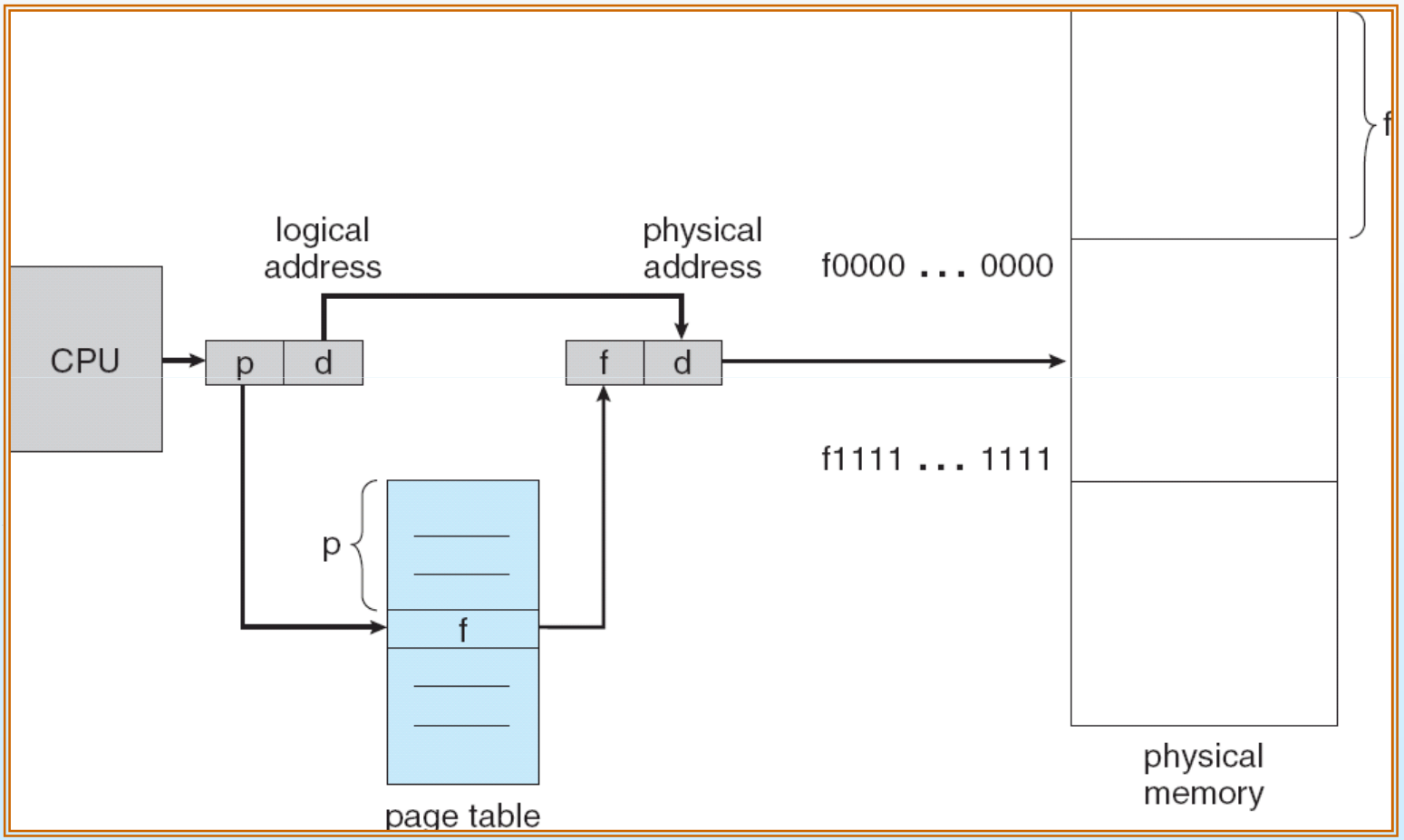
Schema di traduzione dell'indirizzo

Ogni indirizzo generato dalla CPU è diviso in due parti:

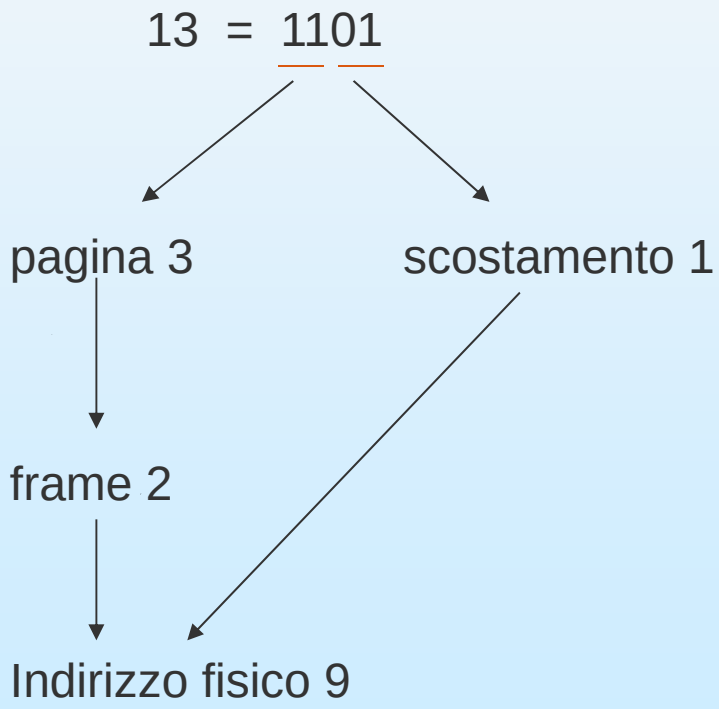
- Numero di pagina (p) – usato come indice nella tabella delle pagine che contiene l'indirizzo di base di ogni frame nella memoria fisica.
- Spiazzamento nella pagina (d) – combinato con l'indirizzo di base per calcolare l'indirizzo di memoria fisica che viene mandato all'unità di memoria centrale.

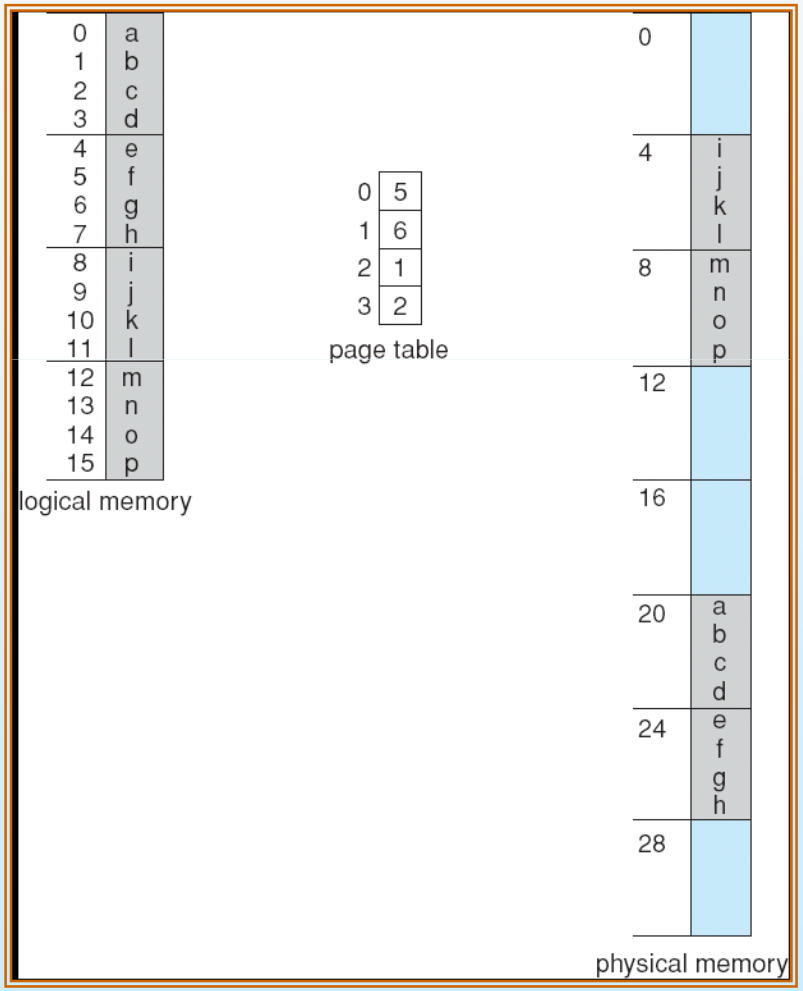


Hardware per la paginazione

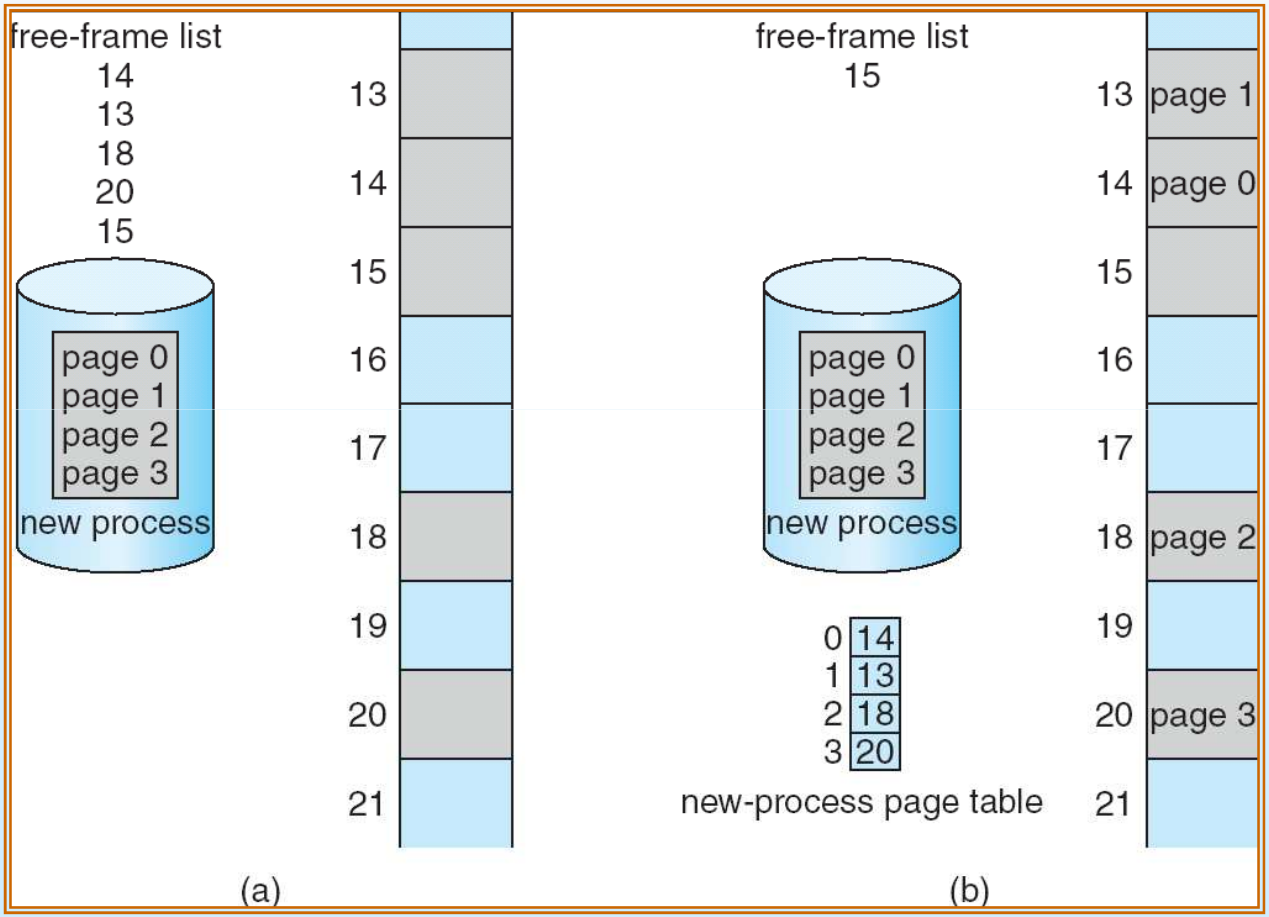


Esempio di paginazione per una memoria centrale di 32 byte con pagine di 4 byte





Frame liberi



Prima dell' allocazione

Dopo l'allocazione

Implementazione di una tabella di pagine

- La tabella delle pagine è mantenuta nella memoria centrale.
- Il registro base della tabella delle pagine (PTBR) punta alla tabella.
  - Quando vi è un cambio di contesto basta cambiare il contenuto del PTBR
- Ogni accesso a dati/istruzioni richiede due accessi alla memoria: uno per la tabella delle pagine e uno per dati/istruzioni.
- Il problema dei due accessi alla memoria può essere risolto attraverso l'uso di una speciale piccola cache per l'indicizzazione veloce detta **memoria associativa** (o *translation look-aside buffer* – TLB).

TLB

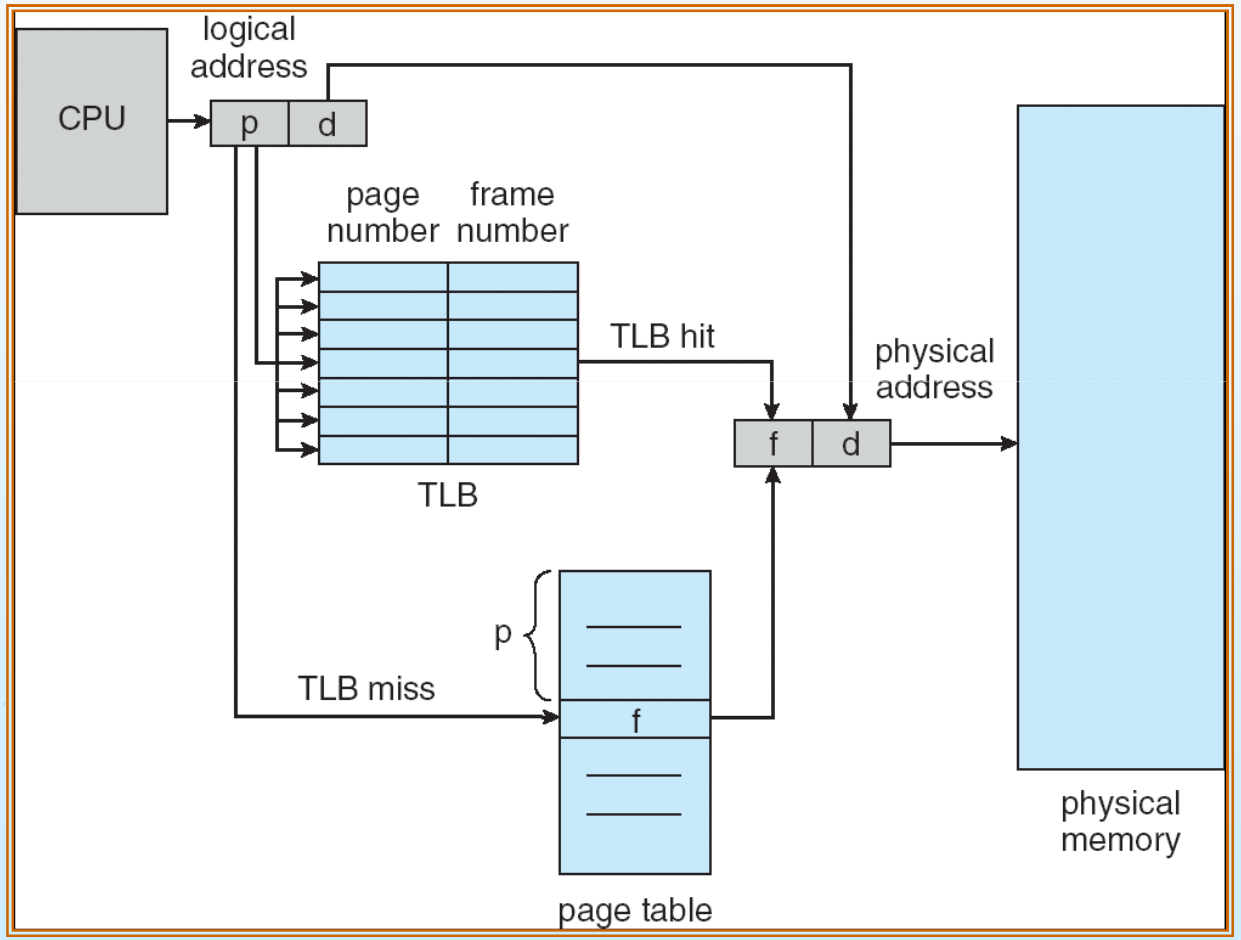
Memoria associativa – ricerca parallela

Pagina #	Frame #

Traduzione dell'indirizzo (P, F)

- Se P si trova nella memoria associativa, ottiene il numero del frame F.
- Altrimenti ottiene il numero del frame dalla tabella in memoria centrale.
- Uso dell' ASID (address-space identifier) in alcune TLB

Hardware per paginazione con TLB



Tempo di accesso effettivo

- Tempo di accesso alla TLB (*associative lookup*) = ε unità di tempo
- Tasso di accesso con successo (*hit ratio*) – frequenza delle volte in cui un particolare numero di pagina viene trovato nella TLB = α
- Tempo di accesso effettivo (EAT)

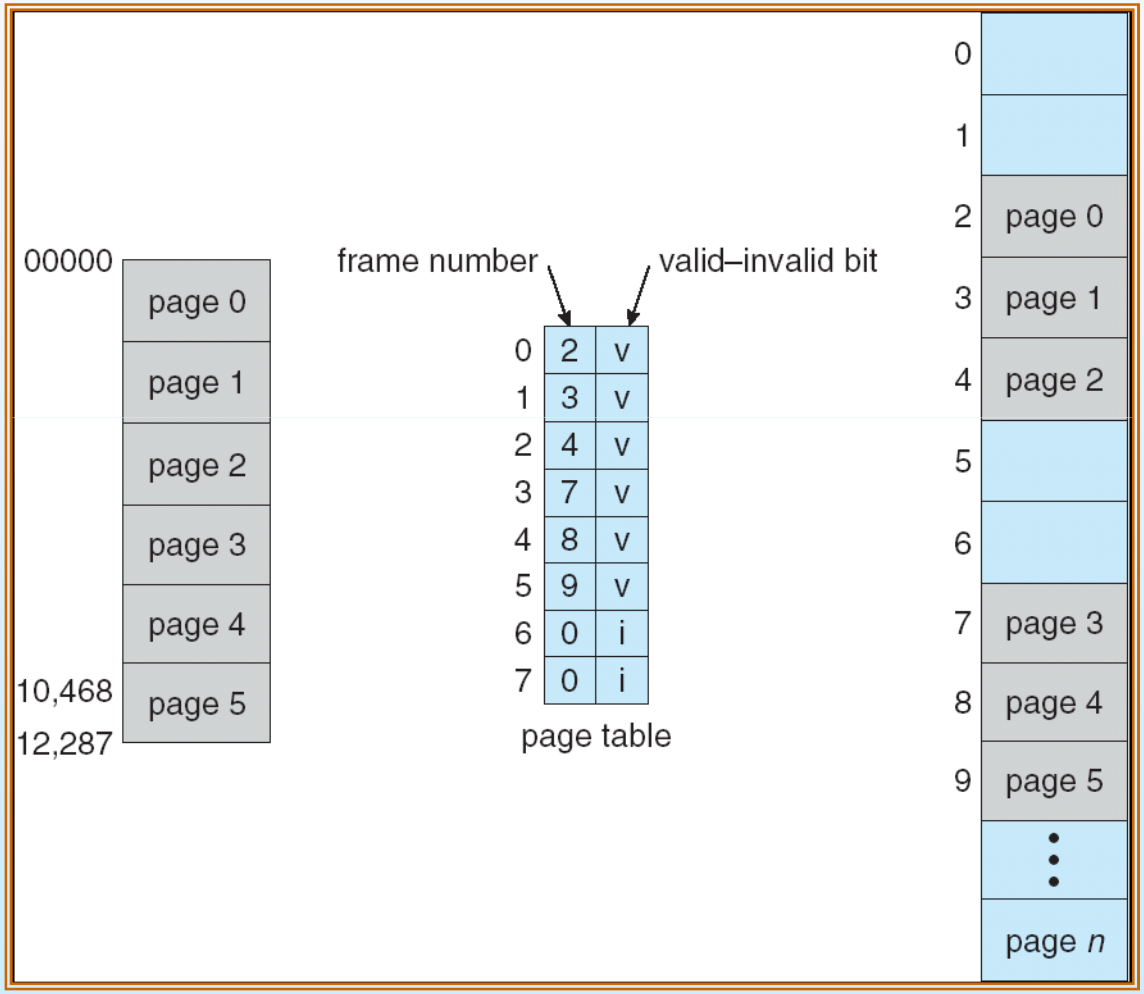
$$EAT = (1 + \varepsilon)\,\alpha + (2 + \varepsilon)(1 - \alpha)$$

$$= 2 + \varepsilon - \alpha$$

Protezione della memoria centrale

- La protezione della memoria centrale in ambiente paginato è ottenuta mediante bit di protezione associati ai frame.
- Un bit di validità/invalidità è associato ad ogni elemento della tabella:
  - "valido" indica che la pagina associata è nello spazio degli indirizzi logici del processo ed è quindi una pagina legale.
  - "non valido" indica che la pagina non è nello spazio degli indirizzi logici del processo.

Bit di validità (v) o di invalidità (i) in una tabella delle pagine



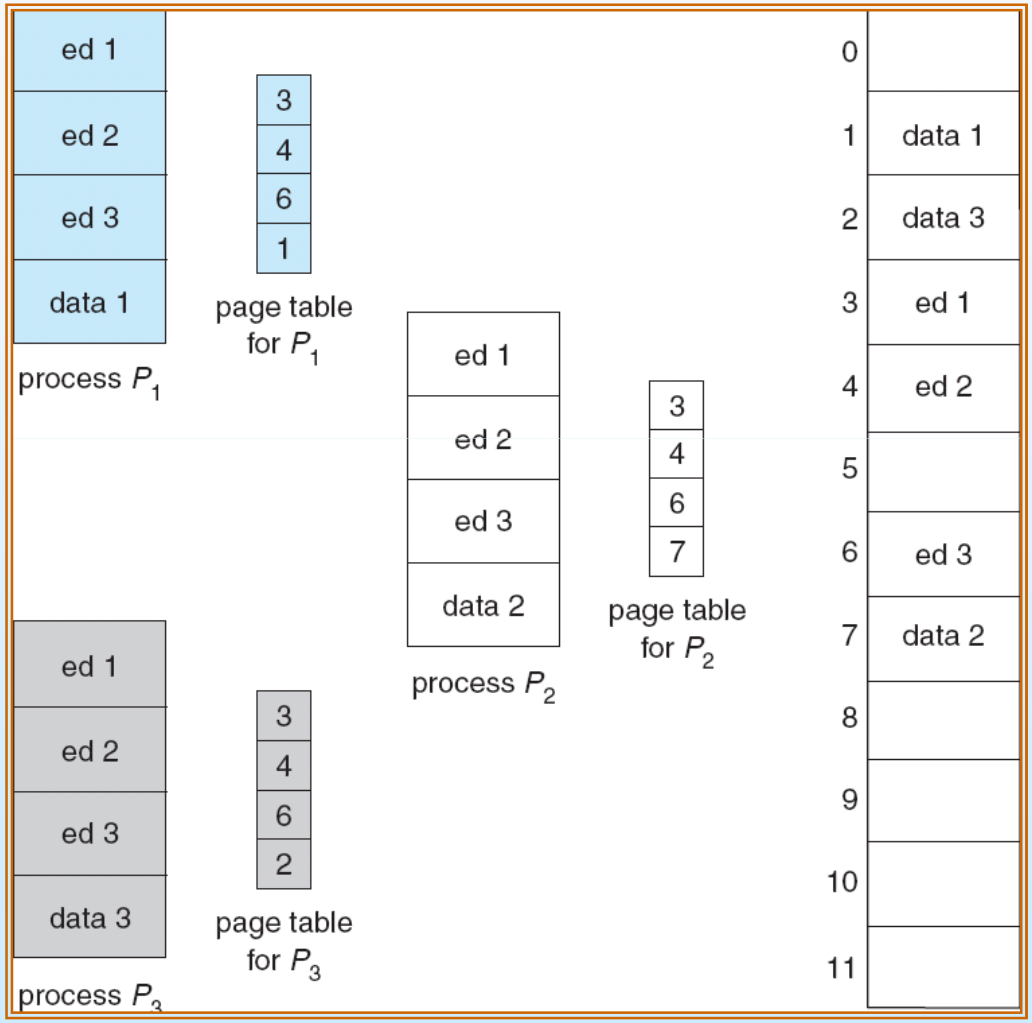
Pagine valide di un processo

- Il registro della lunghezza della tabella delle pagine (PTLR) indica la dimensione della tabella delle pagine del processo.
- Permette di evitare grosse tabelle delle pagine con molte entrate associate a pagine invalide.

Pagine condivise

- Codice condiviso.
  - Una copia di sola lettura di codice rientrante (i.e., non automodificante = non cambia durante l'esecuzione) condiviso fra processi (ad esempio editor, basi di dati).
  - Le tabelle delle pagine dei processi associano le pagine logiche del codice agli stessi frame fisici.
- Codice privato e dati.
  - Ogni processo possiede una copia separata del codice e dei dati.
  - Le pagine per il codice privato ed i dati possono apparire ovunque nello spazio di indirizzo logico.

Esempio di pagine condivise



Struttura della tabella delle pagine

Problema: tabella delle pagine troppo grande.
Allocata in modo contiguo in memoria

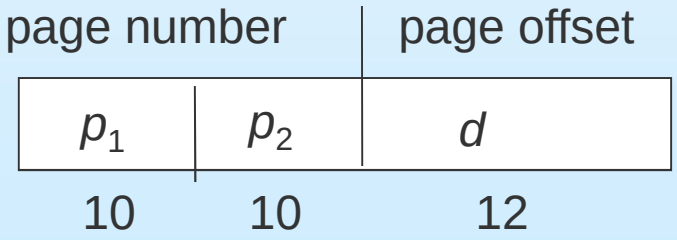
- Paginazione gerarchica.
- Tabelle delle pagine con hashing.
- Tabella delle pagine invertita.

Paginazione gerarchica

- Suddivide lo spazio degli indirizzi logici in più tabelle di pagine.
- Una tecnica semplice è la tabella delle pagine a due livelli.

Esempio di paginazione a due livelli

- Un indirizzo logico (su una macchina a 32-bit con pagine di 4K) è diviso in:
  - un numero di pagina di 20 bit;
  - uno spiazzamento della pagina di 12 bit.
- Il numero di pagina è ulteriormente diviso in :
  - un numero di pagina da 10 bit;
  - uno spiazzamento nella pagina da 10 bit.



dove p1 è un indice per la tabella esterna, e p2 rappresenta lo spostamento nella pagina della tabella interna.

Schema di traduzione dell'indirizzo

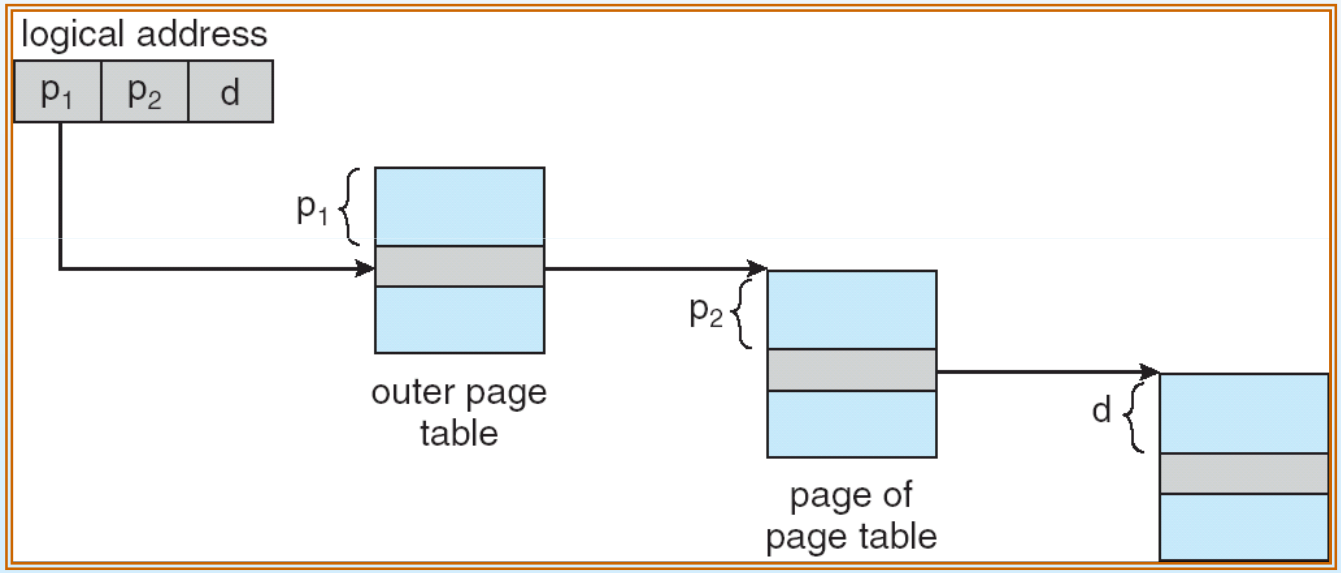
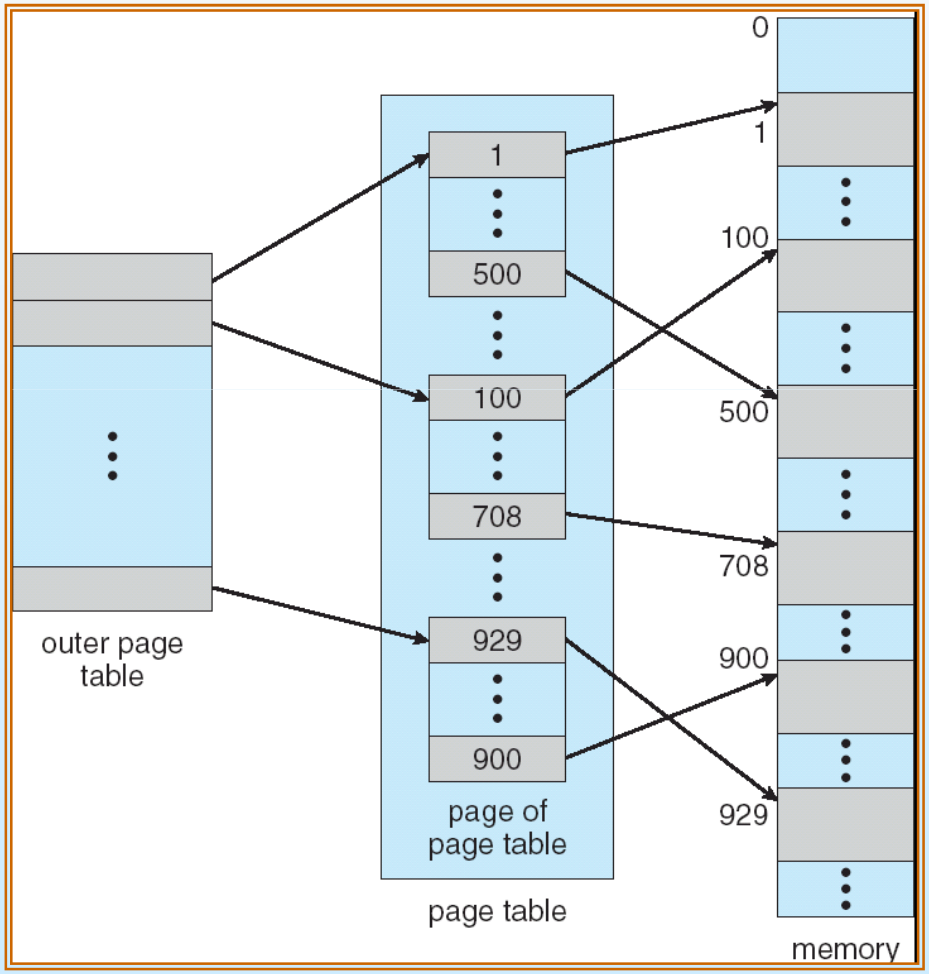


Tabella delle pagine a due livelli



Schema di paginazione a tre livelli

outer page	inner page	offset
p_1	p_2	d
42	10	12

2nd outer page	outer page	inner page	offset
p_1	p_2	p_3	d
32	10	10	12

Più aumentano i livelli della gerarchia più aumentano il numero di accesso necessari alla memoria per recuperare il dato.

Tabella delle pagine con hashing

- Comune per trattare gli spazi di indirizzamento più grandi di 32 bit. Ottima per spazi “sparsi”.
- Il numero di pagina virtuale p è l’input di una funzione hash. Il valore hash prodotto viene usato come indice nella tabella. Ogni elemento della tabella contiene una lista di coppie (numero pagina, frame).
- Il numero di pagina p viene confrontato con i numeri di pagina di questa lista, cercando una corrispondenza. Se viene trovata, il numero di frame fisico associato al numero di pagina viene estratto.

Tabella delle pagine con hashing

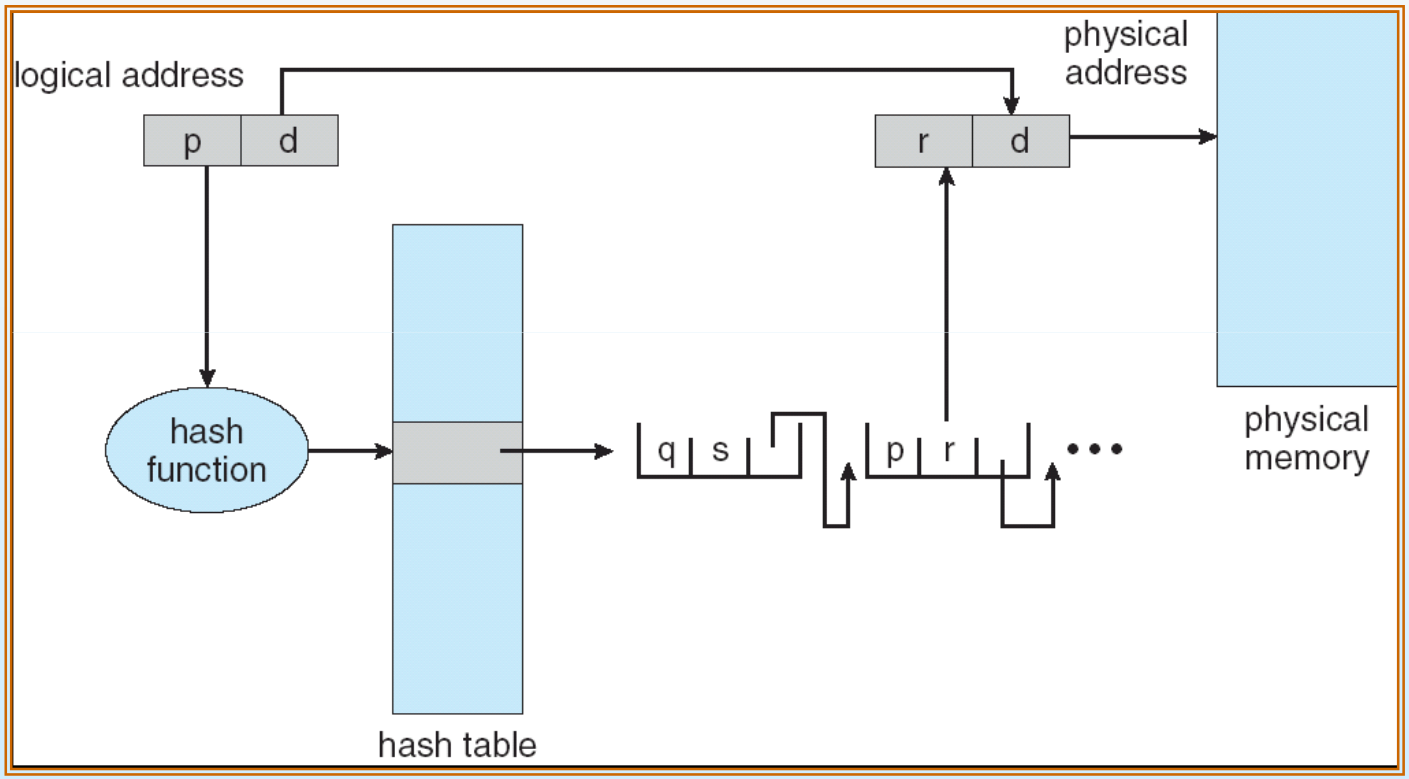


Tabella delle pagine invertita

- Un elemento per ogni frame della memoria centrale.
- Ciascun elemento contiene il numero della pagina logica memorizzata in quel frame fisico, con informazioni sul processo cui appartiene quella pagina.
- Questo schema permette di diminuire la quantità di memoria centrale necessaria per memorizzare le tabelle ma aumenta il tempo necessario per cercare nella tabella quando si verifica un riferimento alla pagina.
- Usare una tabella con hashing per limitare la ricerca a uno o al più a pochi elementi nella tabella delle pagine.

Architettura della tabella delle pagine invertita