

Gestione della memoria

- Introduzione
- Swapping
- Allocazione contigua
- Paginazione

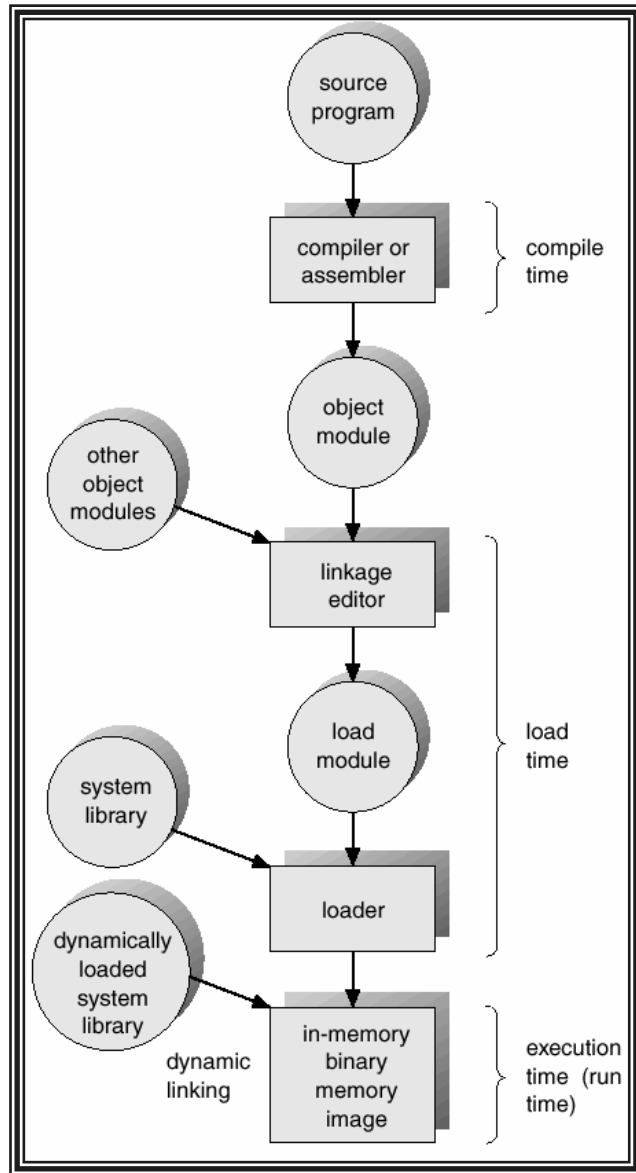
Introduzione

- In un sistema monoprogrammato la memoria centrale è divisa in due parti: una per il sistema operativo, l'altra per il programma che è in quel momento in esecuzione.
- In un sistema multiprogrammato, la parte "utente" deve essere ulteriormente suddivisa per ospitare processi multipli.
- Tale suddivisione è effettuata dinamicamente dal sistema operativo ed è detta **gestione della memoria**.
- Una gestione efficiente della memoria è vitale: se sono presenti in memoria solo pochi processi, per la maggior parte del tempo essi saranno in attesa di I/O.
- Affinché il processore sia sempre occupato, la memoria deve essere allocata efficientemente per immettervi il maggior numero di processi possibile.

Associazione degli indirizzi

- Un programma per essere eseguito deve essere caricato in memoria centrale e inserito all'interno di un processo; durante la sua esecuzione può essere trasferito dalla memoria al disco e viceversa.
- *Coda di input* — collezione di processi su disco in attesa di essere caricati in memoria per essere eseguiti.
- Un processo può risiedere in una qualsiasi parte della memoria fisica, e un processo non è detto che venga sempre caricato a partire dal primo indirizzo.
- I programmi utente passano attraverso stati diversi prima di essere eseguiti e in questi vari stati la rappresentazione degli indirizzi può essere differente.

Associazione degli indirizzi



■ L'associazione – **binding** – di istruzioni e dati a indirizzi di memoria può avvenire durante la fase di...

- ◆ **Compilazione:** se la posizione in memoria del processo è nota a priori, può essere generato un codice assoluto; se la locazione iniziale cambia, è necessario ricompilare il codice;
- ◆ **Caricamento:** se la posizione in memoria non è nota in fase di compilazione, è necessario generare codice *rilocabile*;
- ◆ **Esecuzione:** se il processo può essere spostato *a run-time* da un segmento di memoria all'altro, il binding viene rimandato fino al momento dell'esecuzione. È necessario un opportuno supporto hardware per mappare gli indirizzi (ad esempio attraverso registri *base* e *limite*).

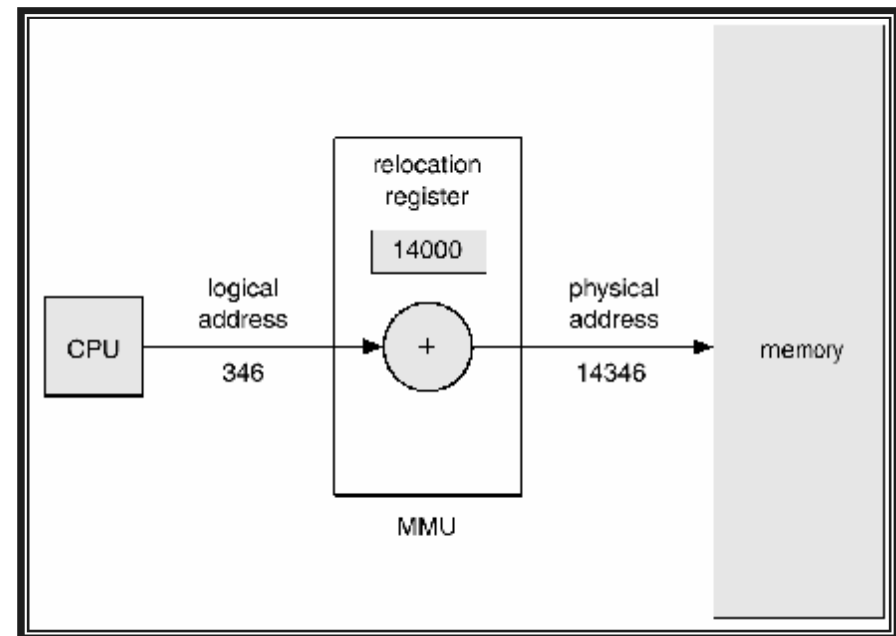
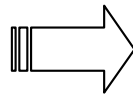
Spazio di indirizzi logici e fisici

- Il concetto di uno *spazio di indirizzi logici* nettamente distinto dallo *spazio degli indirizzi fisici* è centrale per la gestione della memoria.
 - ◆ ***Indirizzi logici*** — generati dalla CPU; chiamati altrimenti *indirizzi virtuali*.
 - ◆ ***Indirizzi fisici*** — indirizzi utilizzati nell'unità di memoria.
- Gli indirizzi logici corrispondono agli indirizzi fisici negli schemi di binding in fase di compilazione e caricamento, mentre differiscono in quelli in fase di esecuzione.

Memory–Management Unit (MMU)

- Dispositivo hardware che mappa indirizzi virtuali su indirizzi fisici.
- Nello schema MMU, il valore contenuto nel *registro di rilocalizzazione* viene sommato ad ogni indirizzo generato dai processi utente nel momento stesso in cui l'indirizzo viene inviato alla memoria.
- Il programma utente *ragiona* in termini di indirizzi virtuali, né mai è (deve essere) *conscio* della loro mappatura fisica.

Rilocalizzazione dinamica
ottenuta tramite registro
di rilocalizzazione



Caricamento dinamico

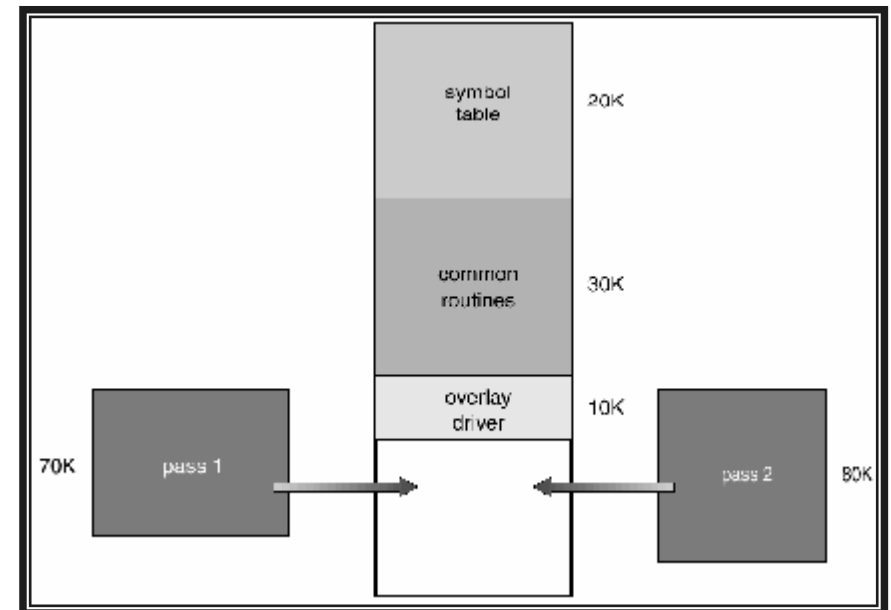
- I sottoprogrammi non vengono caricati in memoria fino a quando non vengono richiamati.
- Si ha un miglior impiego della memoria: sottoprogrammi non utilizzati non vengono mai caricati.
- Utile quando si richiedono grandi quantità di codice per gestire situazioni che avvengono raramente (condizioni di eccezione).
- Al SO non è richiesto alcun supporto speciale. Il caricamento dinamico viene implementato attraverso un opportuno progetto del software.

Link dinamico delle librerie

- Il linking viene rinviato fino al momento dell'esecuzione.
- Questa caratteristica si usa soprattutto con le librerie di sistema, ad esempio quelle del linguaggio.
- In questo modo si evita che gli tutti eseguibili dispongano di una copia delle procedure a cui fanno riferimento.
- Porzioni di codice (dette *stub*) vengono impiegate per localizzare la routine appropriata nella libreria residente in memoria: lo stub viene rimpiazzato con l'indirizzo della routine.
- Se le librerie caricate in memoria possono essere utilizzate da più processi (*librerie condivise*), il SO deve gestirne la condivisione.

Overlay

- Si mantengono in memoria solo istruzioni e dati necessari in un certo istante. Se occorrono altre istruzioni, vengono caricate nello spazio che era occupato dalle istruzioni che non vengono più utilizzate.
- È richiesto quando un processo è più grande della memoria allocatagli.
- L'esecuzione è rallentata a causa dell'operazione di I/O necessaria per caricare l'overlay.
- Viene implementato dall'utente, non viene richiesto alcun supporto speciale da parte del SO. Il progetto di software con overlay è complesso.



Overlay per un assembler a due passi

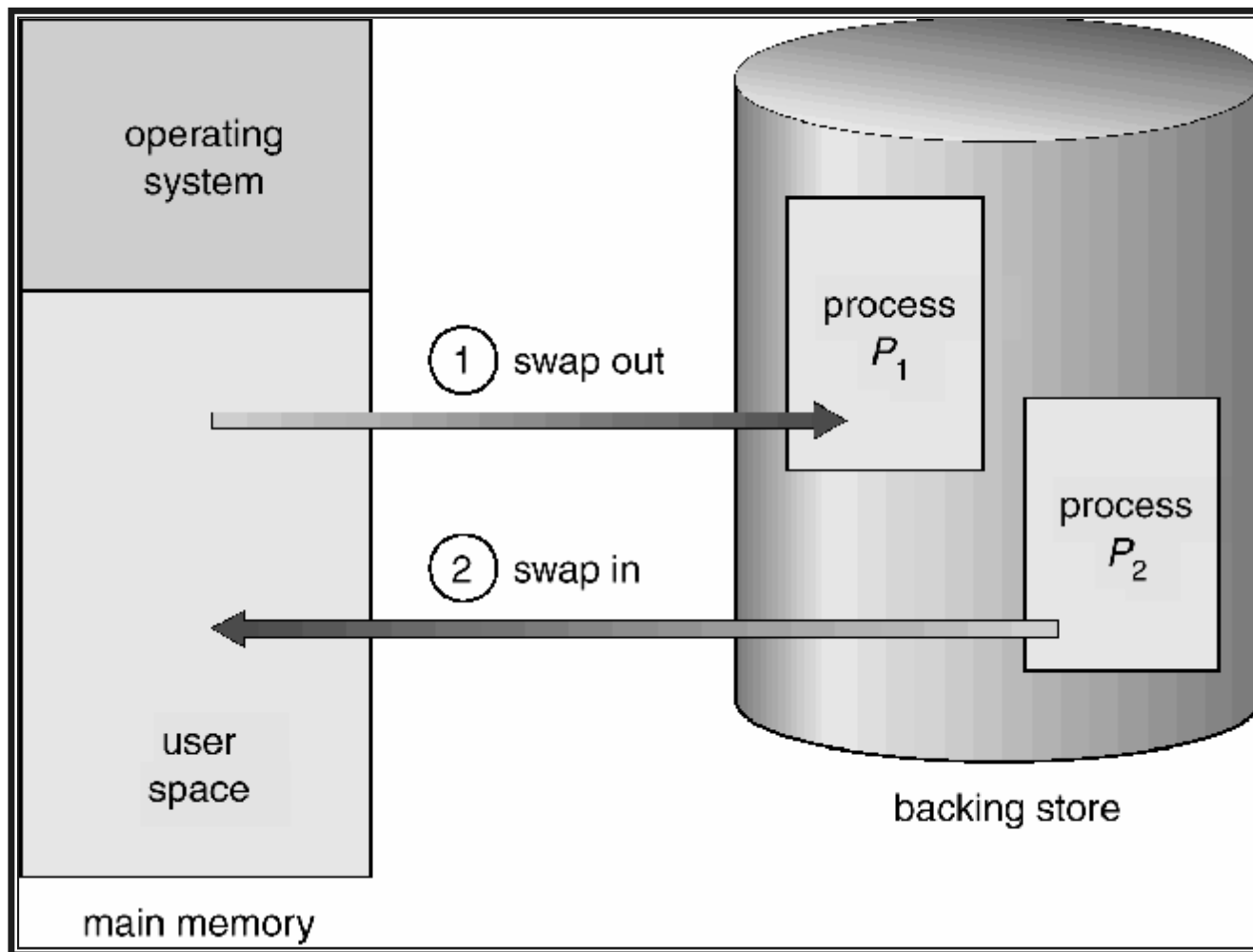
Swapping

- Un processo per essere eseguito deve trovarsi nella memoria centrale. ma può venir temporaneamente riversato (**swapped**) nella **memoria ausiliaria *backing store*** da cui, in seguito, viene prelevato per proseguire l'esecuzione.
- ***Backing store*** — È un disco veloce, sufficientemente capiente da accogliere copie di tutte le immagini di memoria per tutti gli utenti; deve garantire accesso diretto a tali immagini. Può essere una partizione dedicata del disco (pochi movimenti della testina).
- ***Roll out, roll in*** — È una variante dello swapping impiegata per algoritmi di scheduling basati su priorità; un processo a priorità più bassa è riversato sulla memoria di massa, per permettere a un processo a maggior priorità di essere caricato ed eseguito.

Swapping

- Se l'associazione tra indirizzi logici e fisici della memoria è effettuata in fase di assemblaggio o caricamento, il processo non può essere caricato in posizioni differenti.
- La maggior parte del tempo di swap è impiegata per il trasferimento di dati; il tempo totale di trasferimento è direttamente proporzionale alla quantità di memoria riversata.
- Se il binding viene effettuato in fase d'esecuzione, il processo sottoposto a swapping può successivamente essere riversato in uno spazio di memoria diverso.
- Molti SO attuali supportano lo swapping, ad esempio **UNIX** e **MS Windows**.

Swapping

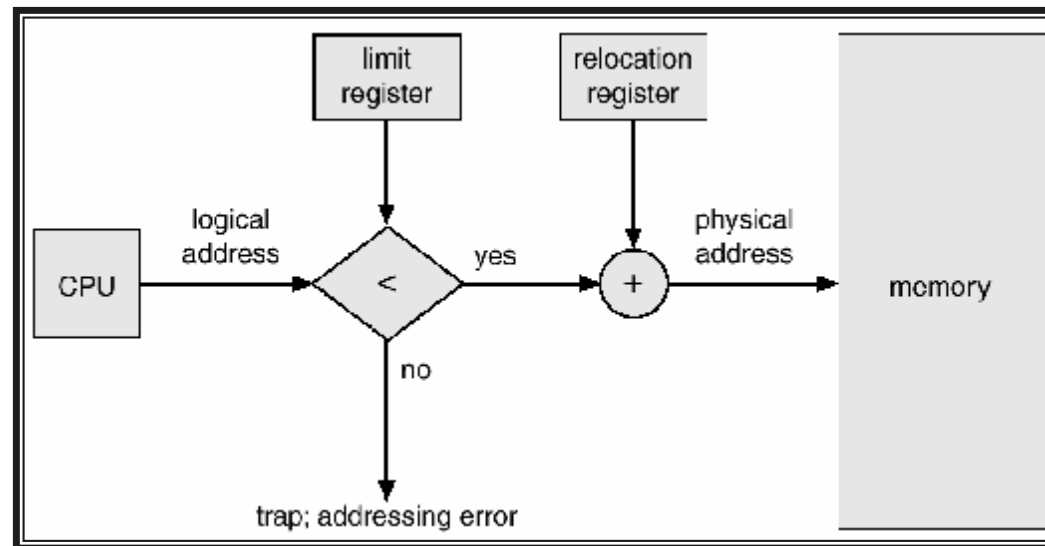


Swapping di due processi utilizzando il disco come backing store

Allocazione contigua

- La memoria principale viene suddivisa in due partizioni:
 - ◆ La parte residente del SO è generalmente memorizzata nella memoria bassa, insieme al *vettore degli interrupt*.
 - ◆ I processi utente sono memorizzati nella memoria alta.
- **Allocazione con partizione singola**
 - ◆ Si impiega uno schema basato su registri base e limite per proteggere programmi e dati del SO e per proteggere reciprocamente i programmi utente.
 - ◆ Il registro di rilocazione contiene il valore del più piccolo indirizzo fisico di memoria allocata al processo; il registro limite contiene l'intervallo degli indirizzi logici: ciascun indirizzo logico deve essere inferiore al valore del registro limite.

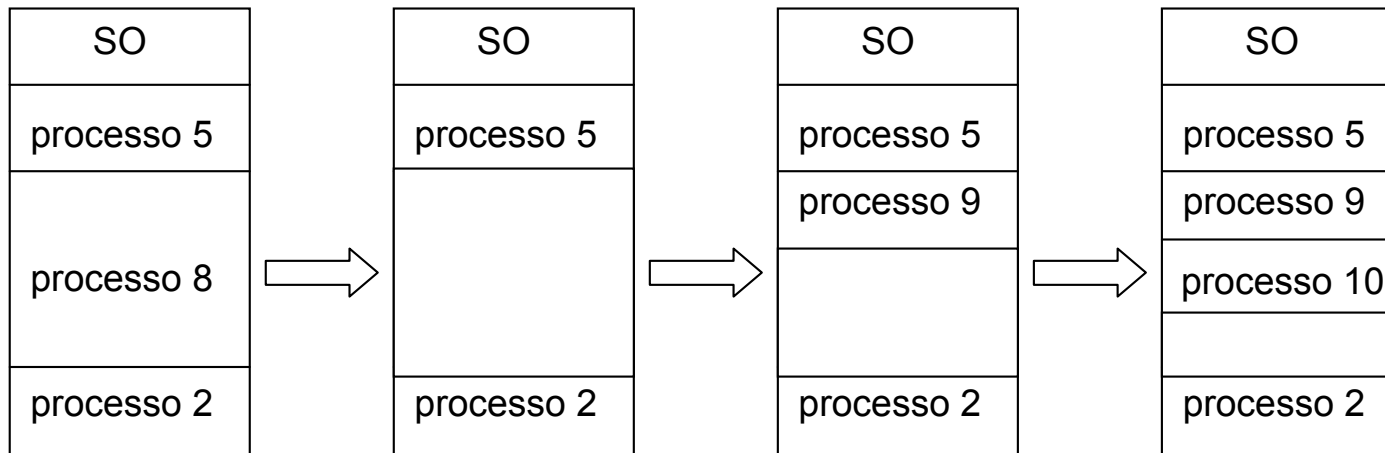
Supporto hardware ai registri base e limite.



Allocazione contigua

■ Allocazione con partizioni multiple

- ◆ Nei primi SO si avevano partizioni di dimensioni fisse.
- ◆ Un *bucco* (*hole*) è un blocco di memoria disponibile; nella memoria sono sparsi buchi di varie dimensioni.
- ◆ Quando viene caricato un nuovo processo, gli viene allocato un buco grande abbastanza da contenere il processo.
- ◆ Il SO conserva informazioni su:
 - a) *Partizioni allocate* b) *Partizioni libere* (buchi)



Problemi di allocazione dinamica della memoria

- Come soddisfare una richiesta di dimensione n a partire da un insieme di buchi?
- In ogni momento è presente un insieme di buchi di diverse dimensioni sparsi per la memoria.
 - ◆ **First-fit**: Viene allocato il primo buco grande abbastanza.
 - ◆ **Best-fit**: Viene allocato il buco più piccolo capace di contenere il processo. È necessario scandire tutta la lista dei buchi. Si produce il più piccolo buco residuo.
 - ◆ **Worst-fit**: Viene allocato il buco più grande. È ancora necessario ricercare in tutta la lista. Si produce il più grande buco residuo.
- First-fit e Best-fit sono mediamente migliori di Worst-fit, rispettivamente in termini di velocità e impiego di memoria.

Frammentazione

- **Frammentazione esterna** — È disponibile lo spazio totale per soddisfare una richiesta, ma non è contiguo.
- **Frammentazione interna** — La memoria allocata può essere leggermente maggiore della memoria richiesta (pochi byte di differenza). La differenza di dimensioni è memoria interna ad una partizione che non viene impiegata.
- Si può ridurre la frammentazione esterna con la compattazione
 - ◆ Si postano i contenuti della memoria per avere tutta la memoria libera contigua a formare un grande blocco.
 - ◆ La compattazione è possibile *solo* con la rilocalizzazione dinamica e viene effettuata in fase d'esecuzione.
 - ◆ Si possono spostare tutti i processi verso un'estremità della memoria.

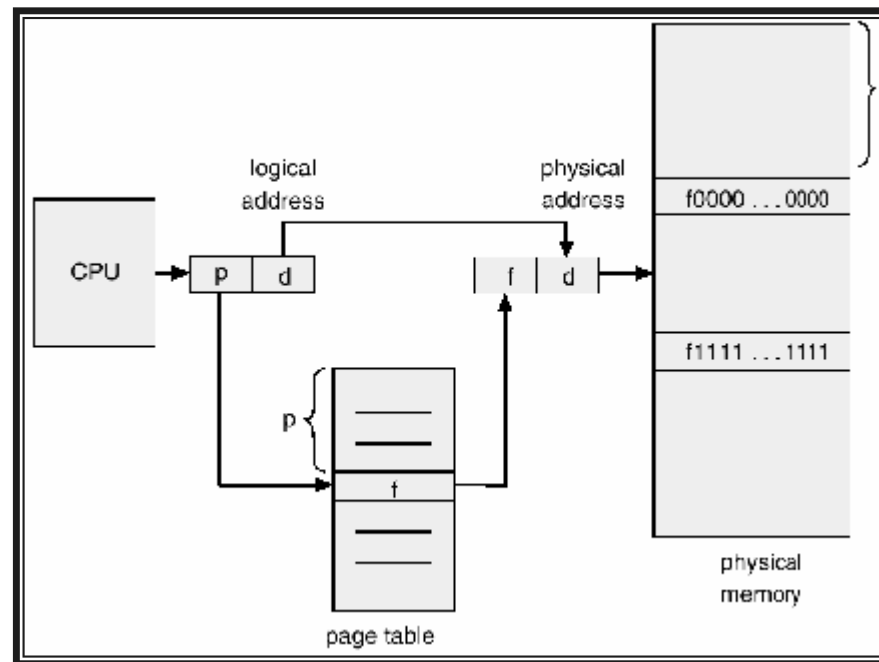
Paginazione

- Un'altra soluzione alla frammentazione esterna è ottenuta consentendo la non contiguità degli indirizzi fisici: allocazione della memoria fisica ai processi ovunque essa sia disponibile.
- Si divide la memoria fisica in blocchi di dimensione fissa chiamati **frame** (la dimensione è una potenza del 2, compresa fra 512 e 8192 byte).
- Si divide la memoria logica in blocchi della stessa dimensione chiamati **pagine**.
- Si tiene traccia di tutti i frame liberi.
- Per eseguire un programma di dimensione n pagine, è necessario trovare n frame liberi prima di caricare il programma.
- Si impiega una **tabella delle pagine** per tradurre gli indirizzi logici negli indirizzi fisici.
- Si ha solo frammentazione interna (relativa all'ultimo frame).

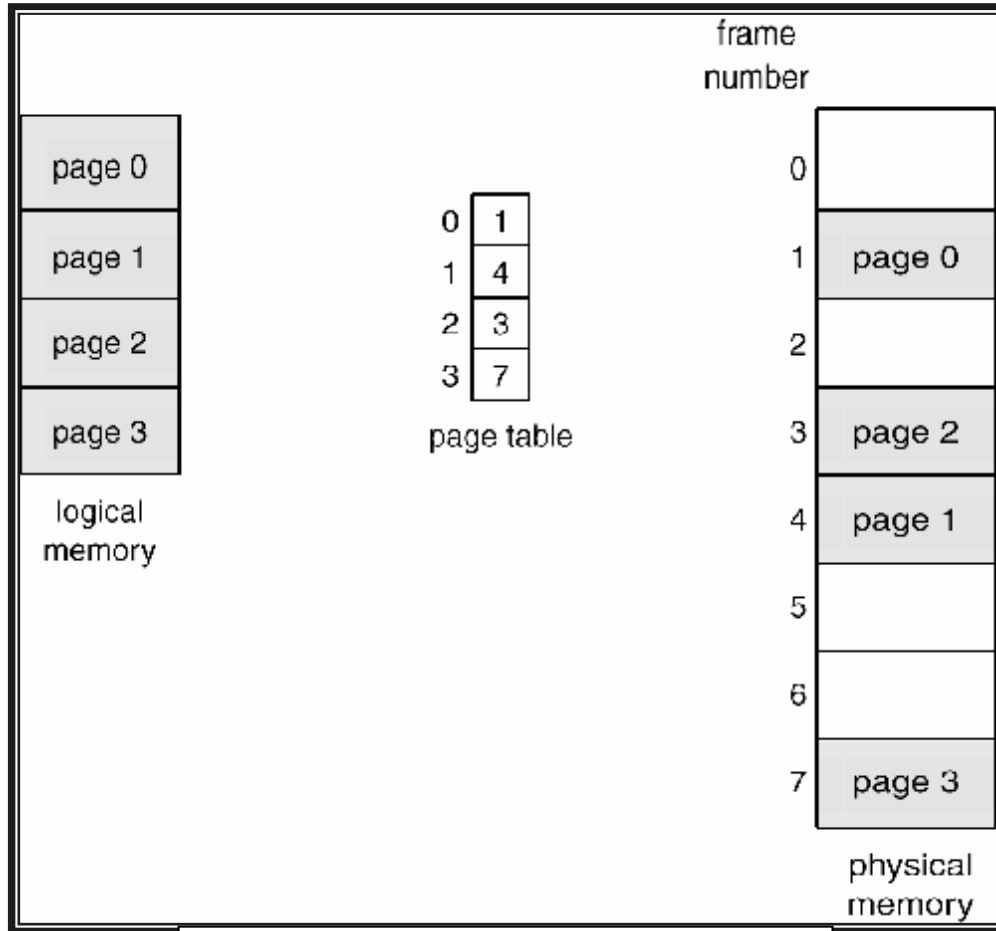
Schema di traduzione degli indirizzi

- L'indirizzo generato dalla CPU viene suddiviso in:
 - ◆ **Numero di pagina (p)** — impiegato come indice in una tabella di pagine che contiene l'indirizzo base di ciascuna pagina nella memoria fisica.
 - ◆ **Offset nella pagina (d)** — combinato con l'indirizzo base per definire l'indirizzo fisico di memoria che viene inviato all'unità di memoria.

Supporto hardware alla traduzione degli indirizzi.



Modello di paginazione

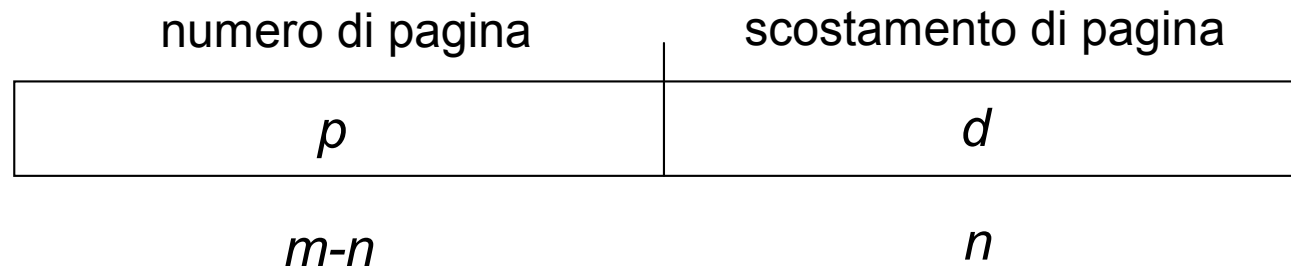


Modello di paginazione di memoria logica e memoria fisica

- Il numero di pagina serve come indice per la **tabella delle pagine**.
- Questa contiene l'indirizzo di base nella memoria fisica di ogni pagina.
- L'indirizzo di base si appaia con lo scostamento di pagina per definire l'indirizzo della memoria fisica.

Dimensione delle pagine

- La dimensione di una pagina, così come quella di un frame, è definita dall'architettura del calcolatore ed è tipicamente una potenza di 2 compresa tra 512 byte e 16 MB.
- Le dimensioni sono potenza di 2 perché ciò facilita la traduzione degli logici in numero di pagina e scostamento.
- Se lo spazio degli indirizzi logici è 2^m e la dimensione di una pagina è 2^n , allora:
 - ◆ gli $m-n$ bit più significativi di un indirizzo logico indicano il **numero di pagina**,
 - ◆ gli n bit meno significativi indicano lo **scostamento**.



Esempio di paginazione

■ Esempio: pagine di 4 byte e memoria fisica di 32 byte (8 pagine).

■ Indirizzo logico 0 → Pagina 0 + scostamento 0

◆ Pagina 0 → Blocco 5

◆ Indirizzo logico 0 → Indirizzo fisico 20

■ Indirizzo logico 3 → Pagina 0 + scostamento 3

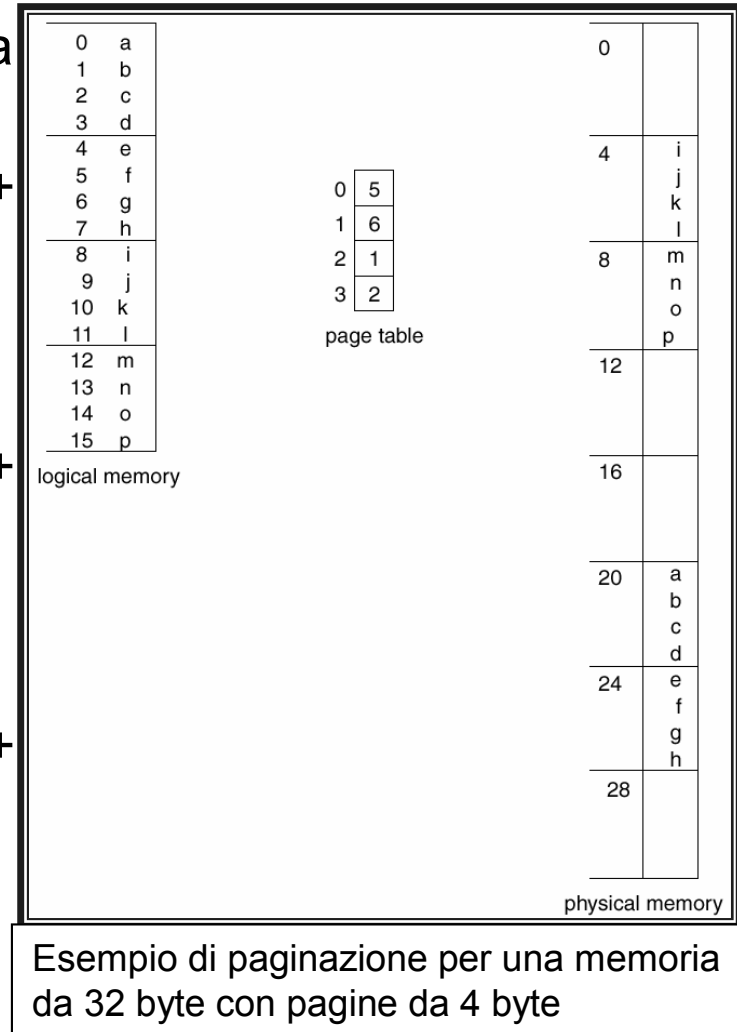
◆ Pagina 0 → Blocco 5

◆ Indirizzo logico 3 → Indirizzo fisico 23

■ Indirizzo logico 4 → Pagina 1 + scostamento 0

◆ Pagina 1 → Blocco 6

◆ Indirizzo logico 4 → Indirizzo fisico 24

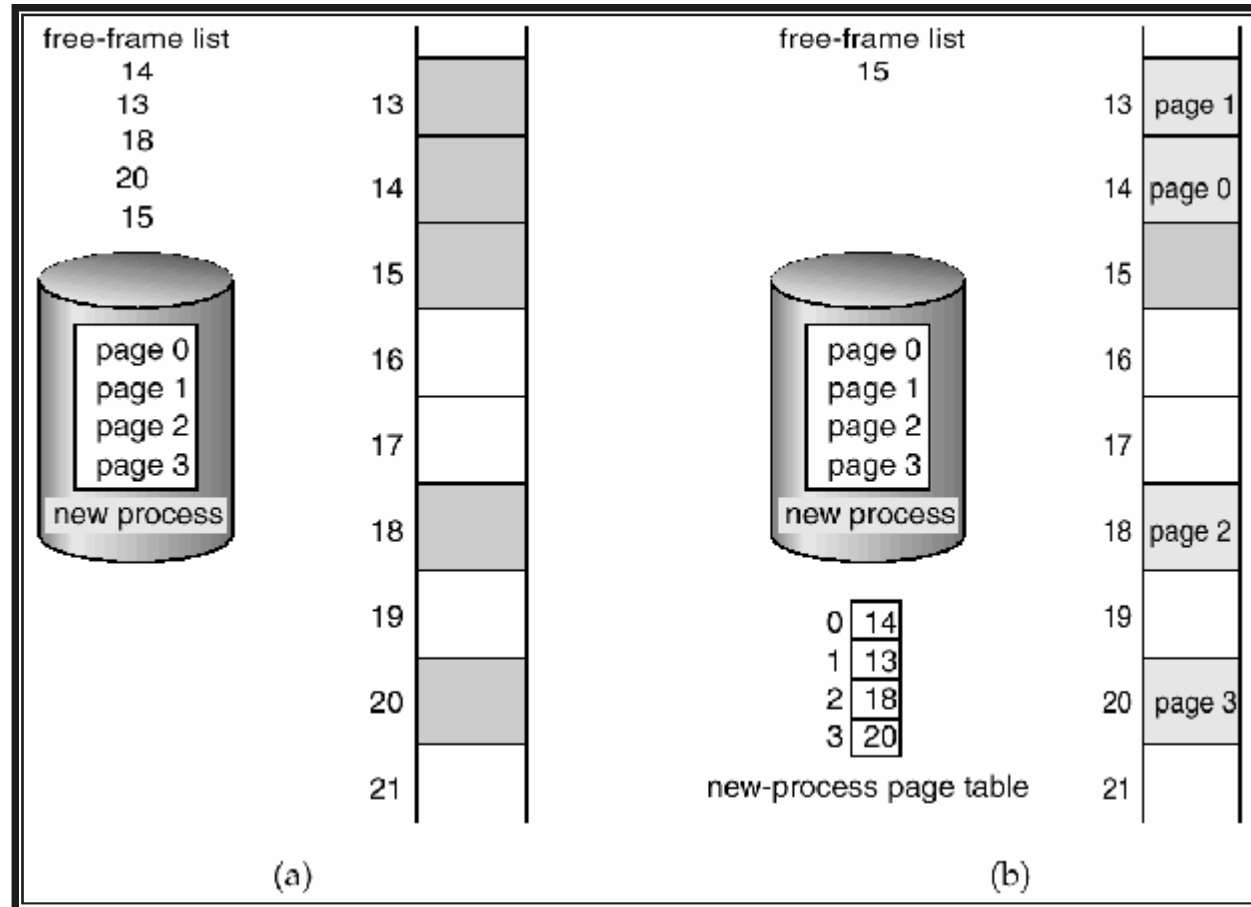


Paginazione vs rilocazione

- La paginazione è una forma di rilocazione dinamica: ad ogni indirizzo logico l'architettura di paginazione fa corrispondere un indirizzo fisico.
- Ciascuna riga della tabella delle pagine è analoga ad un registro di rilocazione.
- Con la paginazione si evita la frammentazione esterna, ma non la paginazione interna, che mediamente è pari alla metà di un blocco.
- Blocchi piccoli diminuiscono la frammentazione interna, blocchi grandi facilitano il trasferimento dalla memoria secondaria.
- Alcune CPU e kernel permettono l'utilizzo di pagine di dimensione variabile (Solaris)

Esempi di paginazione

Frame liberi



Prima dell'allocazione

Dopo l'allocazione

Tabella dei blocchi di memoria

- Nella paginazione vi è una netta differenza tra memoria fisica e memoria logica: per il programma utente la memoria è un unico spazio contiguo.
- In realtà il programma è sparso nella memoria fisica, che contiene anche altri programmi.
- La differenza tra memoria fisica e memoria logica è contenuta nell'architettura di traduzione degli indirizzi, che fa corrispondere indirizzi fisici e logici.
- Queste operazioni sono controllate dal sistema operativo, che mantiene le informazioni sull'assegnazione in una **tabella dei blocchi di memoria**.