Gestione della memoria

4.1 Introduzione alla gestione della memoria

- **Analogie tra la gestione della CPU e la gestione della memoria**

Tecnica di virtualizzazione delle risorse

Memoria virtuale

4.1 Introduzione alla gestione della memoria

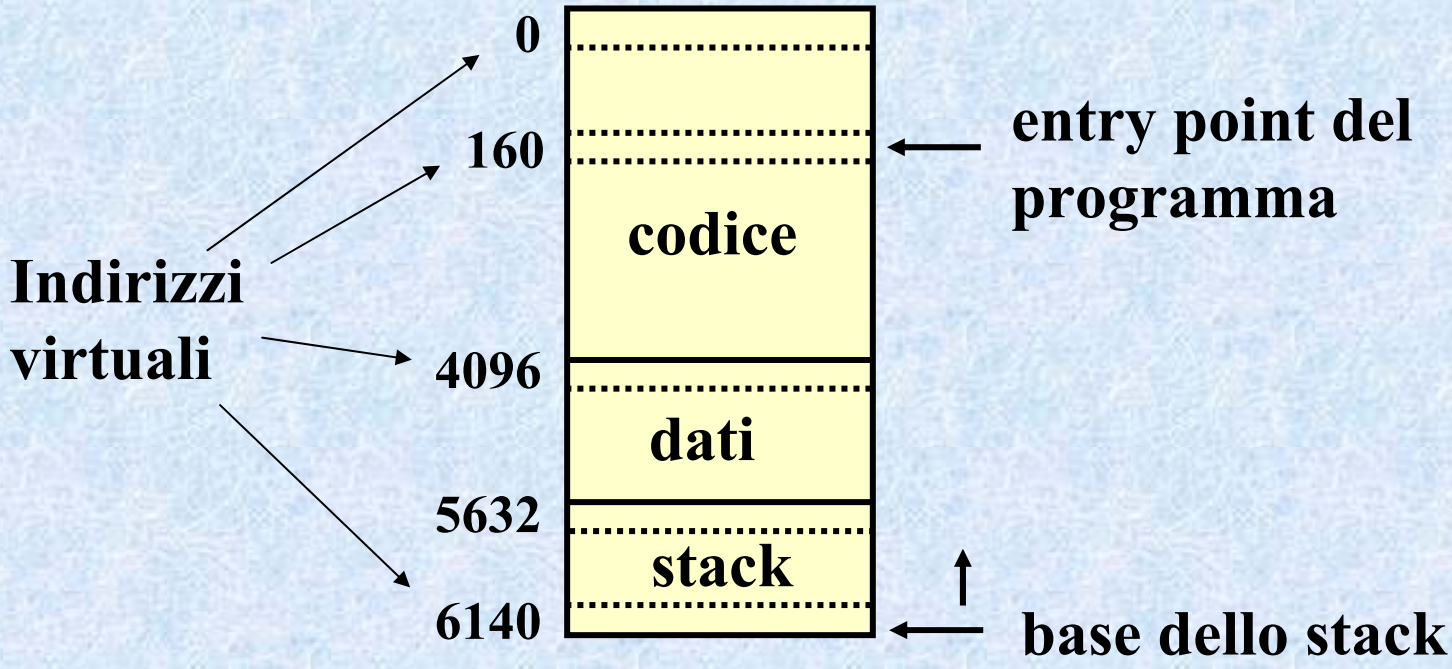
- **Differenze tra la gestione della CPU e la gestione della memoria**

Partizioni diverse della memoria possono essere contemporaneamente allocate a processi diversi

- ✓ **Necessità di meccanismi di protezione**
- ✓ **Opportunità di condivisione**

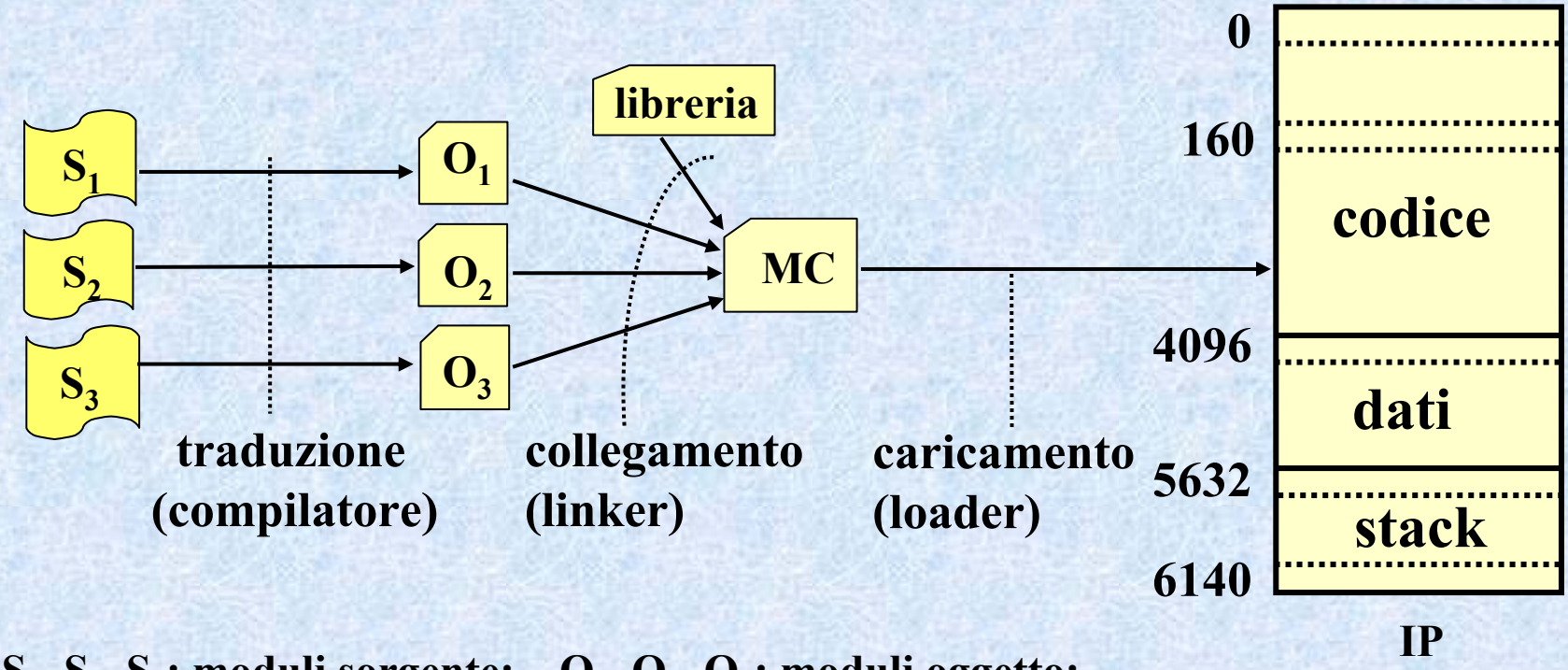
4.2 Aspetti caratterizzanti la gestione della memoria

- **Immagine di un processo**



4.2 Aspetti caratterizzanti la gestione della memoria

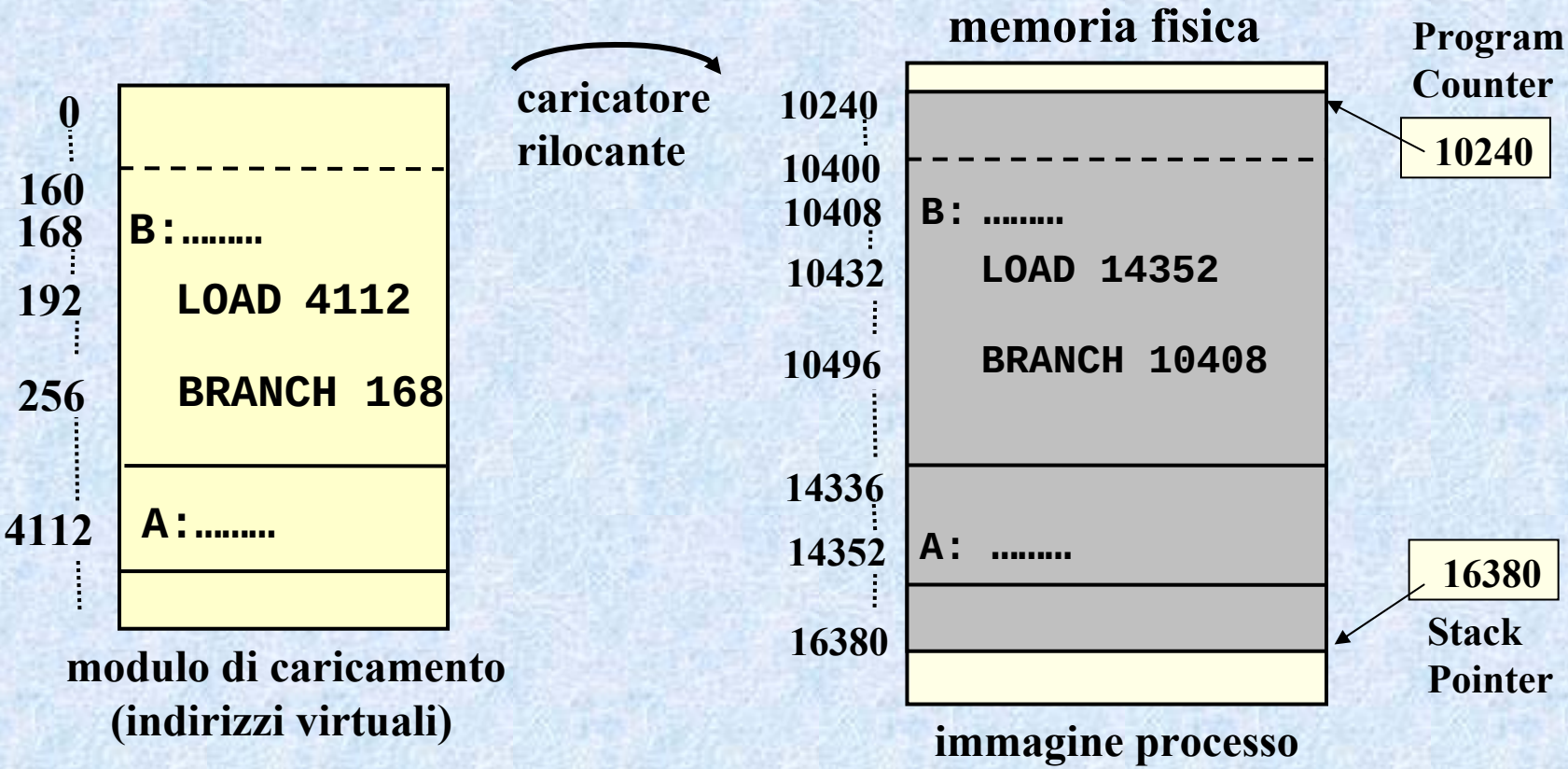
- **Preparazione di un programma per l'esecuzione**



S_1, S_2, S_3: moduli sorgente; O_1, O_2, O_3: moduli oggetto;
MC: modulo di caricamento (file eseguibile);
IP: immagine del processo

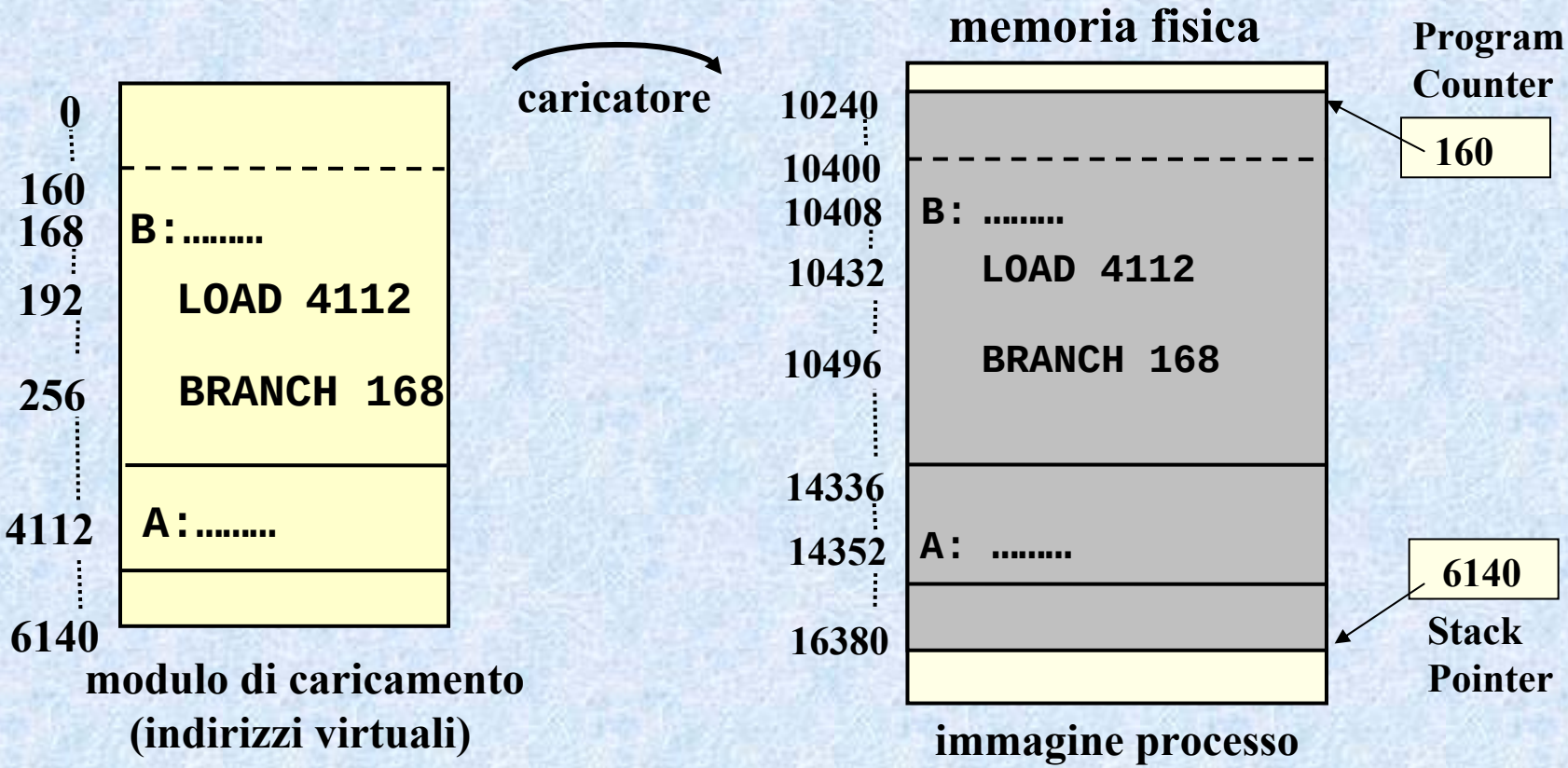
4.2 Aspetti caratterizzanti la gestione della memoria

- **Rilocazione degli indirizzi: rilocazione statica**

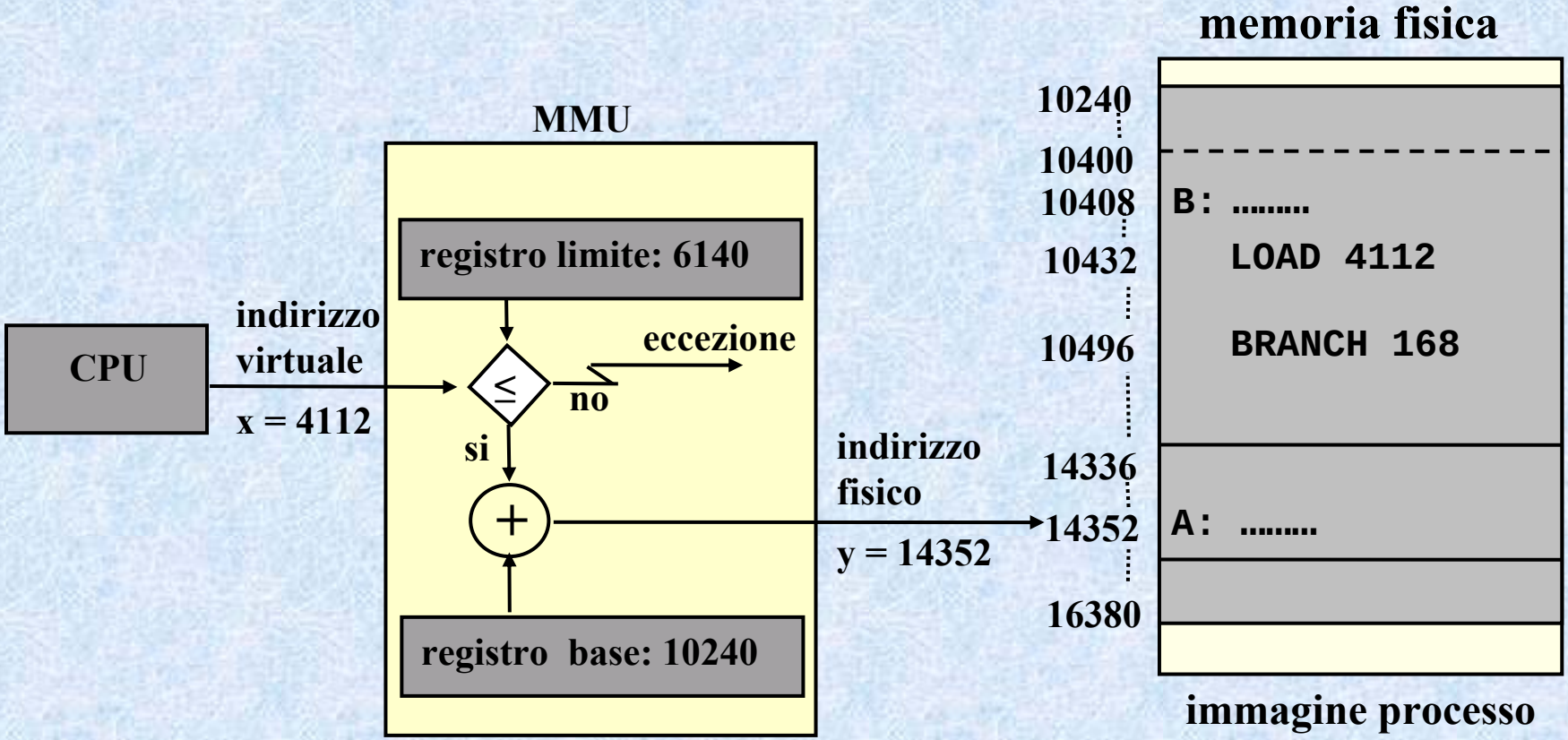


4.2 Aspetti caratterizzanti la gestione della memoria

- **Rilocazione degli indirizzi: rilocazione dinamica**



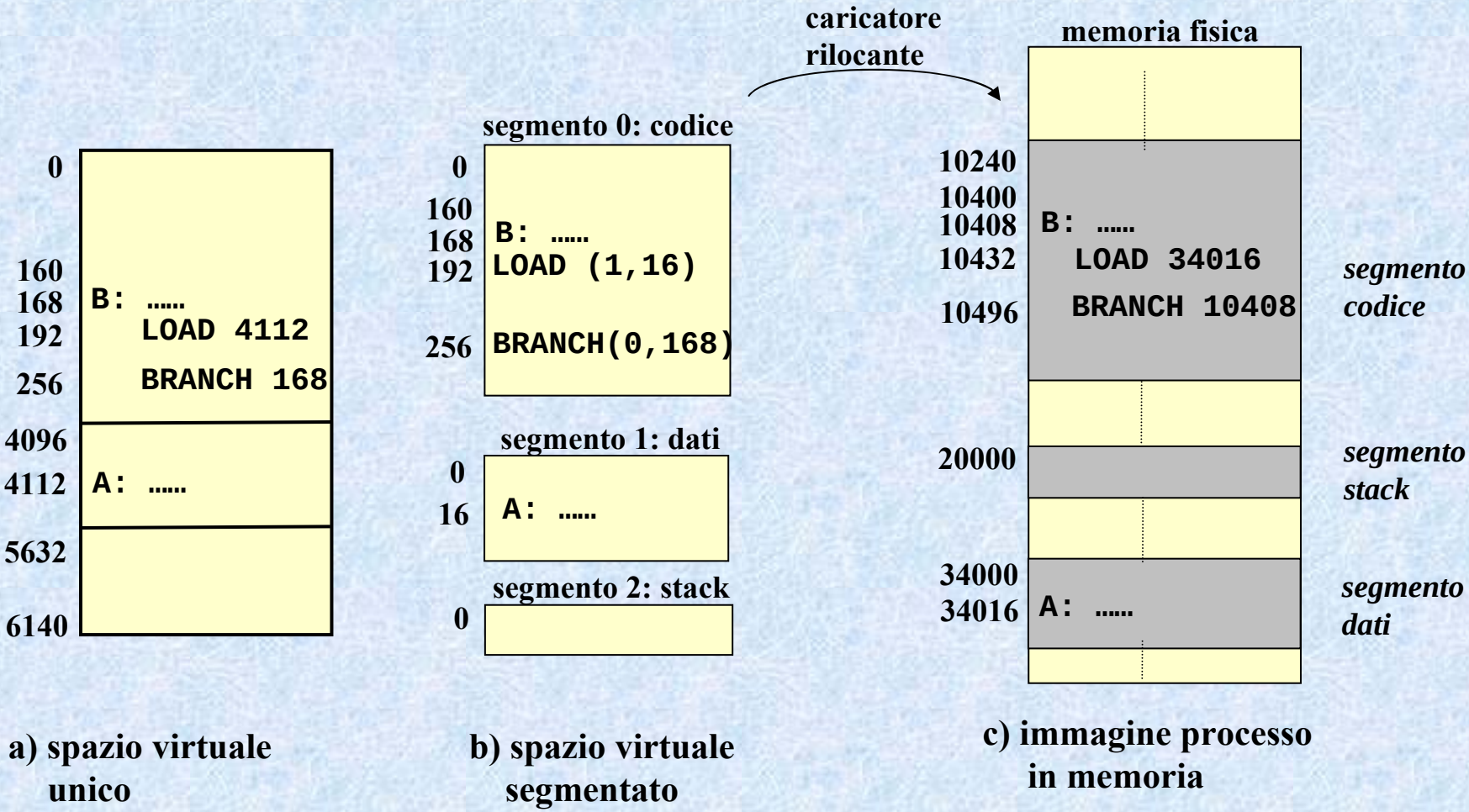
Rilocazione dinamica: Memory Management Unit



4.2 Aspetti caratterizzanti la gestione della memoria

- **Organizzazione della memoria virtuale**
  - ✓ **unica**
  - ✓ **segmentata**

Spazio virtuale segmentato



a) spazio virtuale unico

b) spazio virtuale segmentato

c) immagine processo in memoria

4.2 Aspetti caratterizzanti la gestione della memoria

- **Allocazione della memoria fisica**
  - ✓ **contigua**
  - ✓ **non contigua**

4.2 Aspetti caratterizzanti la gestione della memoria

- **Caricamento nella memoria fisica**
  - ✓ **unico**
  - ✓ **a domanda**

==> Dimensione della memoria virtuale

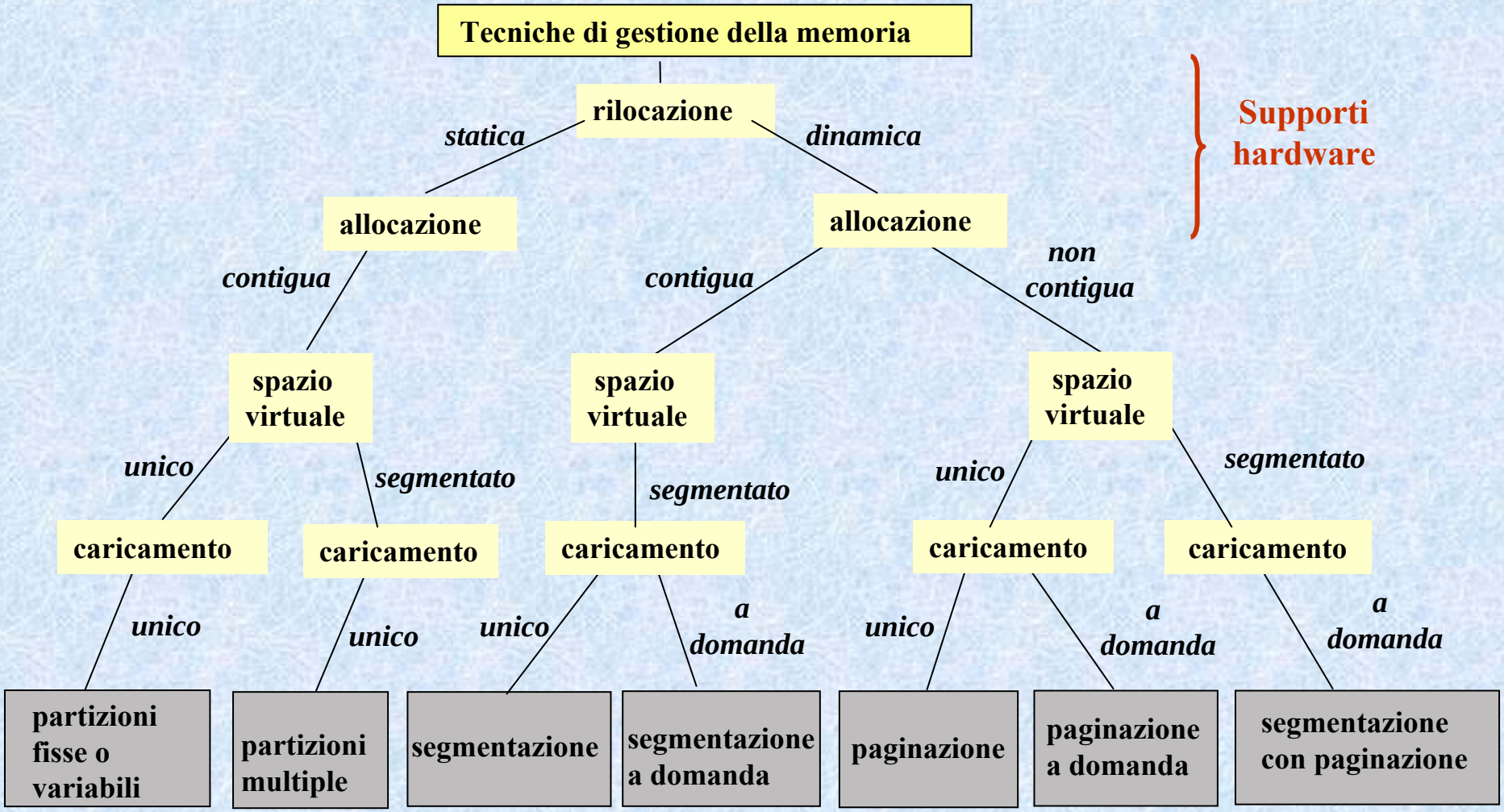
1) minore o uguale a quella della memoria fisica: possibile il caricamento unico

2) superiore a quella della memoria fisica: necessario il caricamento a domanda

4.2 Aspetti caratterizzanti la gestione della memoria

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
• statica • dinamica	• contigua • non contigua	• unico • segmentato	• unico • a domanda

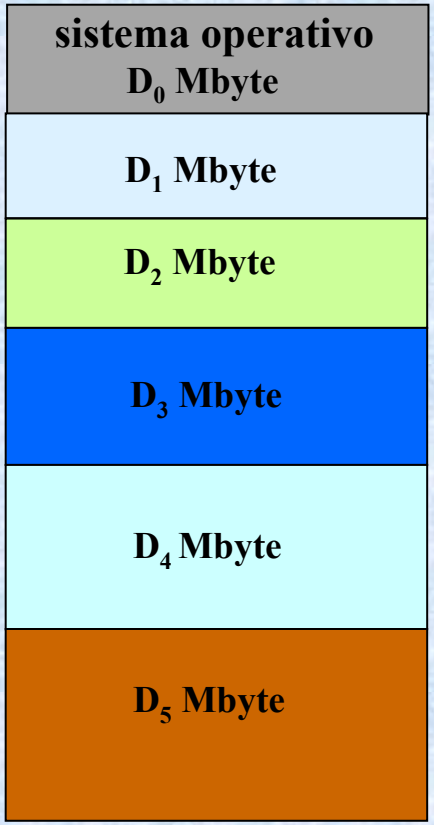
4.3 Tecniche di gestione della memoria



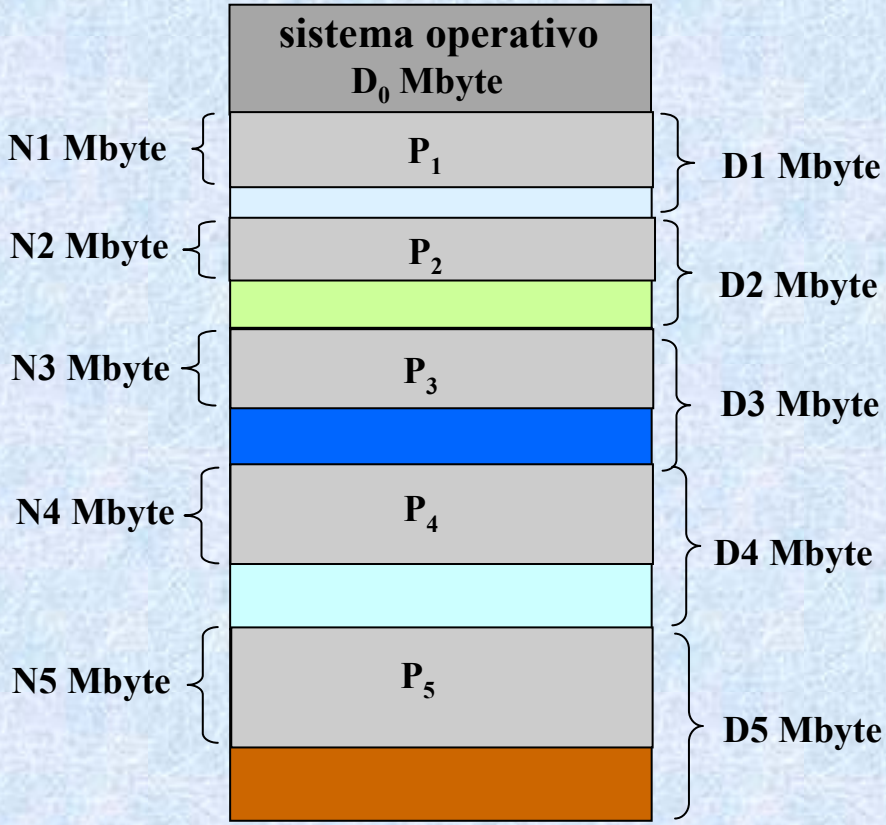
Partizioni fisse e variabili

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
STATICA	CONTIGUA	UNICO	UNICO

Partizioni fisse

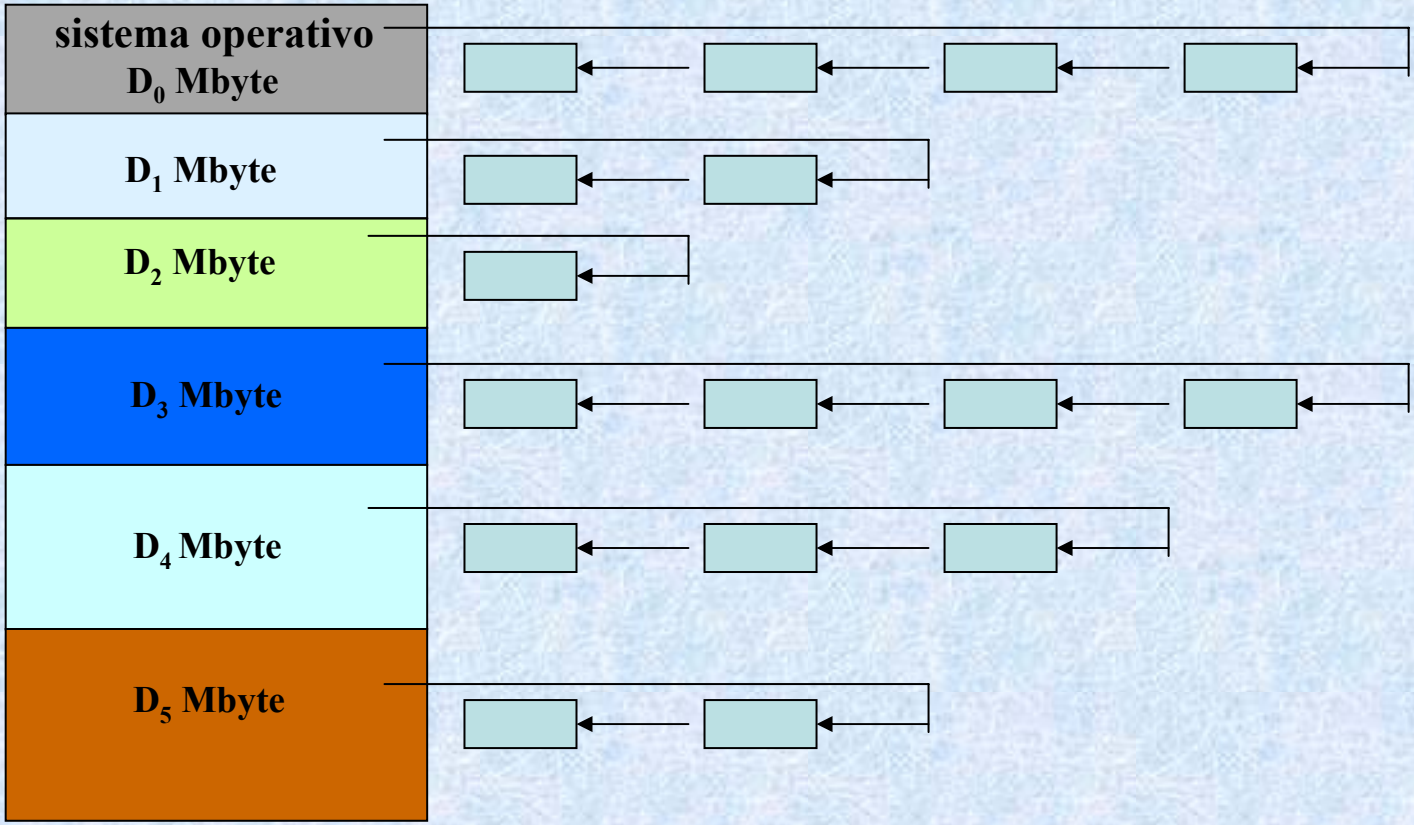


a) partizioni libere

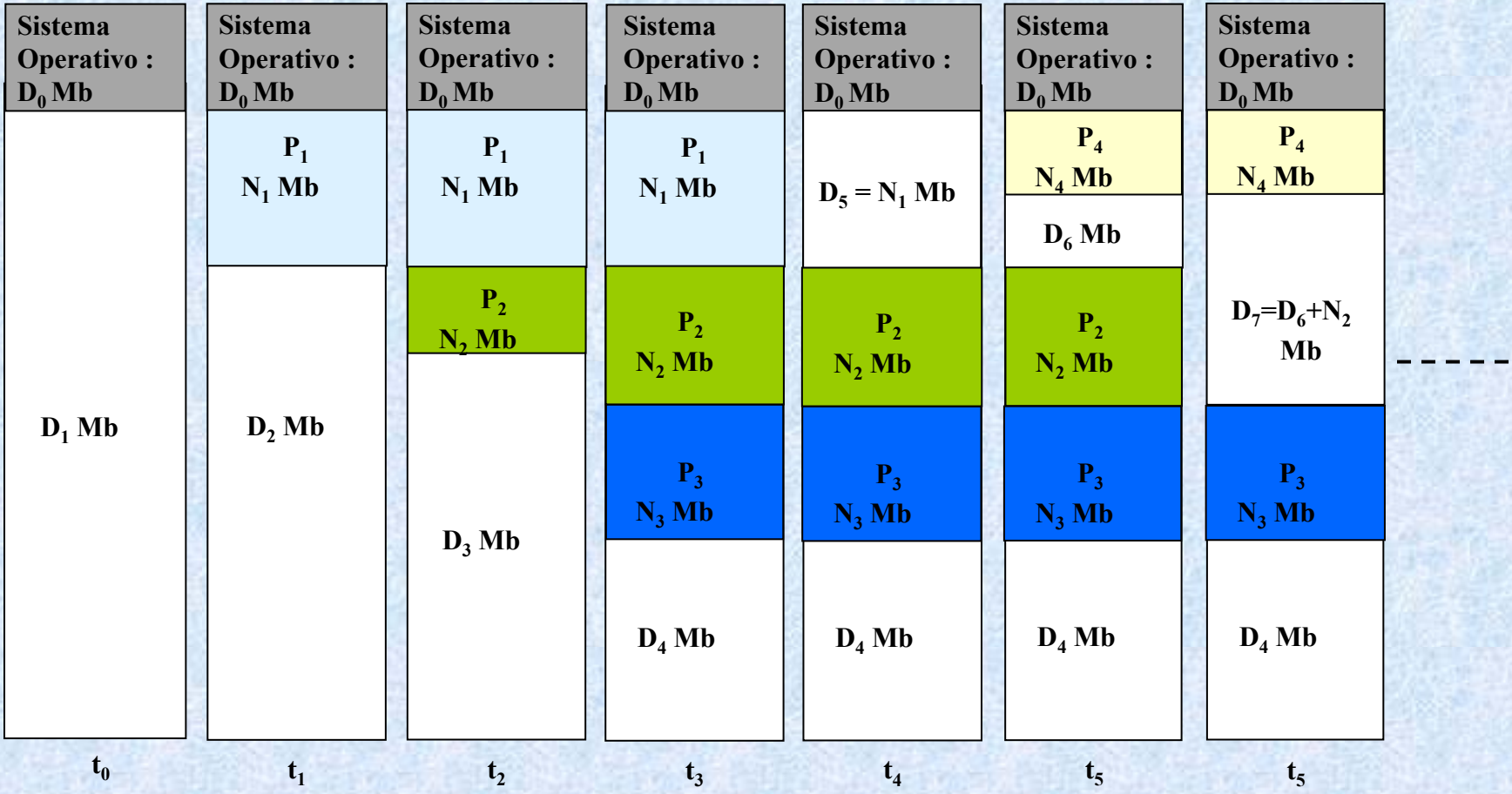


b) frammentazione interna =
$(D_1-N_1)+(D_2-N_2)+(D_3-N_3)+(D_4-N_4)+(D_5-N_5)$

Partizioni fisse: caricamento dinamico



Partizioni variabili



✓ **scomposizione, ricomposizione**

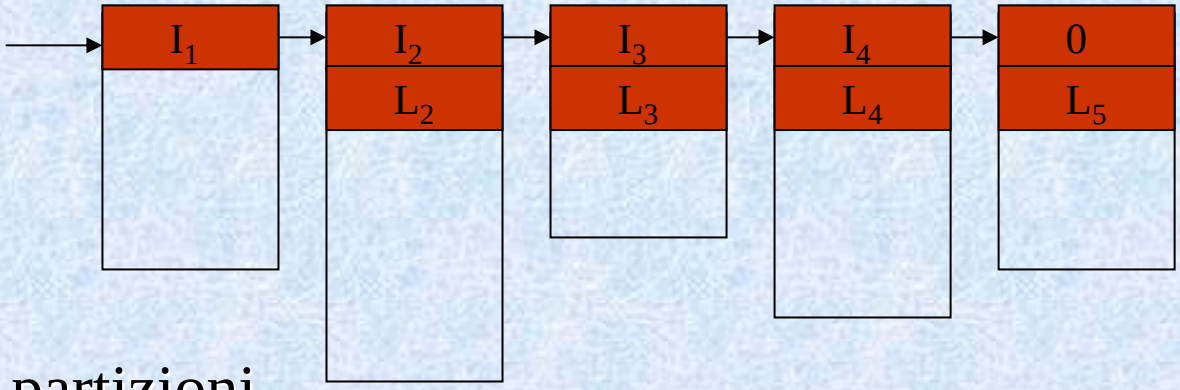
Criteri di allocazione delle partizioni libere

- **First-fit**

  fra tutte le partizioni libere di dimensioni sufficienti a soddisfare la richiesta viene scelta quella di indirizzo minimo.

- **Best-fit**

  fra tutte le partizioni libere di dimensioni sufficienti a soddisfare la richiesta viene scelta quella di lunghezza minima



Lista di partizioni

Partizioni e frammentazione

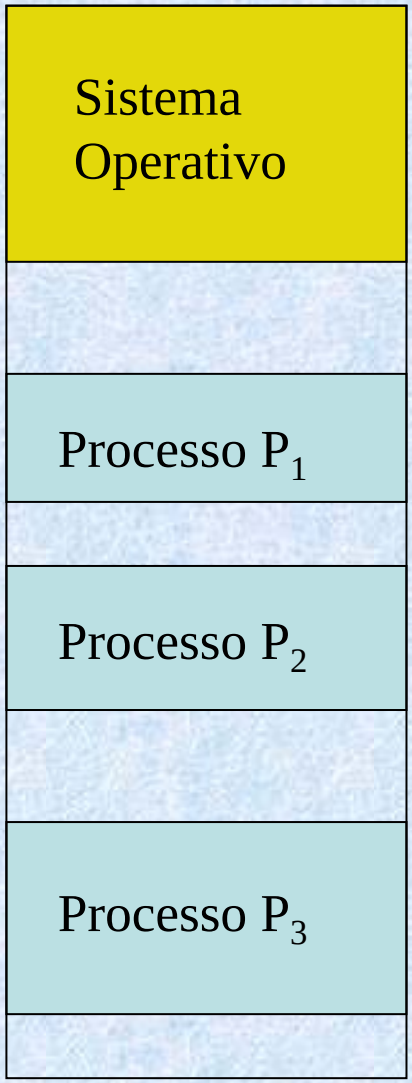
- **Frammentazione interna**

  frazione della memoria non utilizzata nella tecnica delle partizioni fisse: corrisponde alla differenza tra la somma delle dimensioni delle partizioni allocate e la somma delle dimensioni delle partizioni richieste.

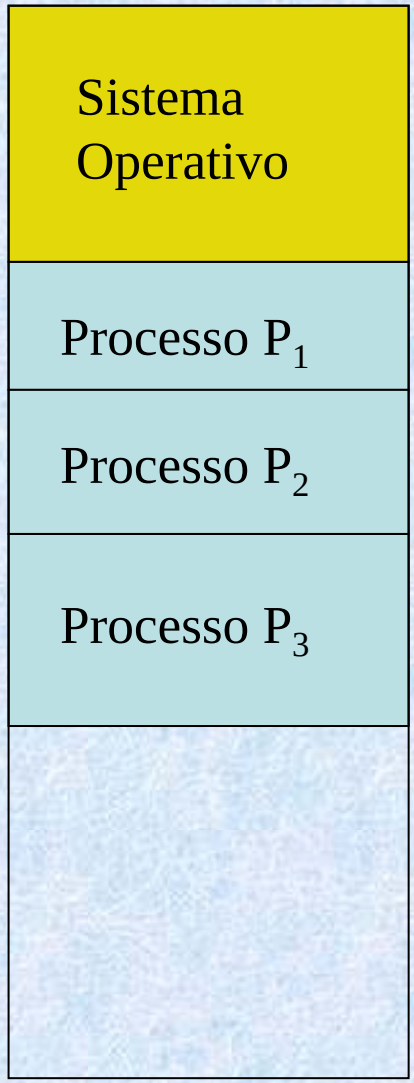
- **Frammentazione esterna**

  frazione della memoria non utilizzata nella tecnica delle partizioni variabili: corrisponde alla somma delle dimensioni delle partizioni libere quando ciascuna di esse non è da sola sufficiente a soddisfare una richiesta di memoria.

Compattamento



Memoria frammentata

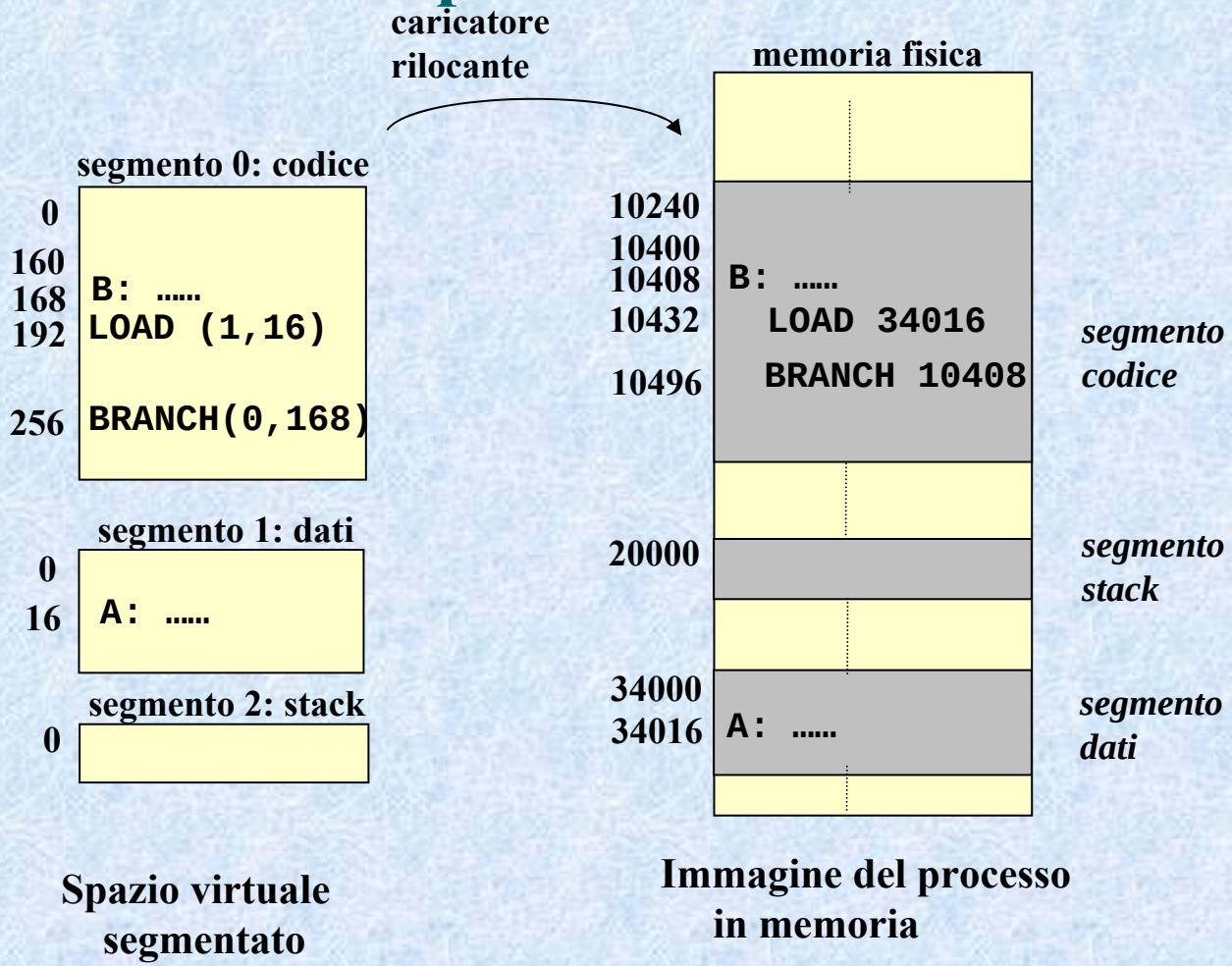


Memoria compattata

Partizioni multiple

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
STATICA	CONTIGUA	SEGMENTATO	UNICO

Partizioni multiple con rilocazione statica

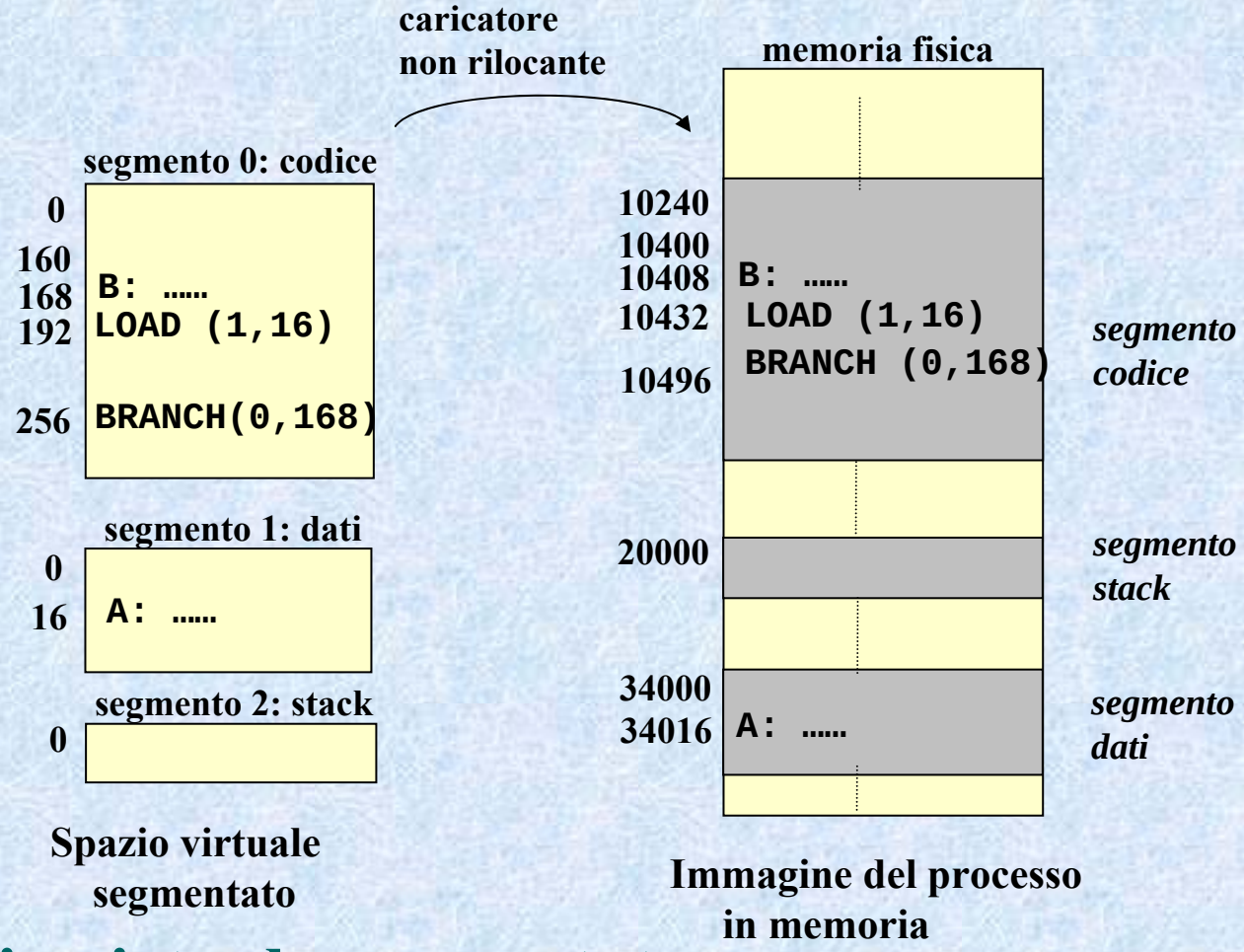


Spazio virtuale segmentato

Segmentazione

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
DINAMICA	CONTIGUA	SEGMENTATO	UNICO

Segmentazione

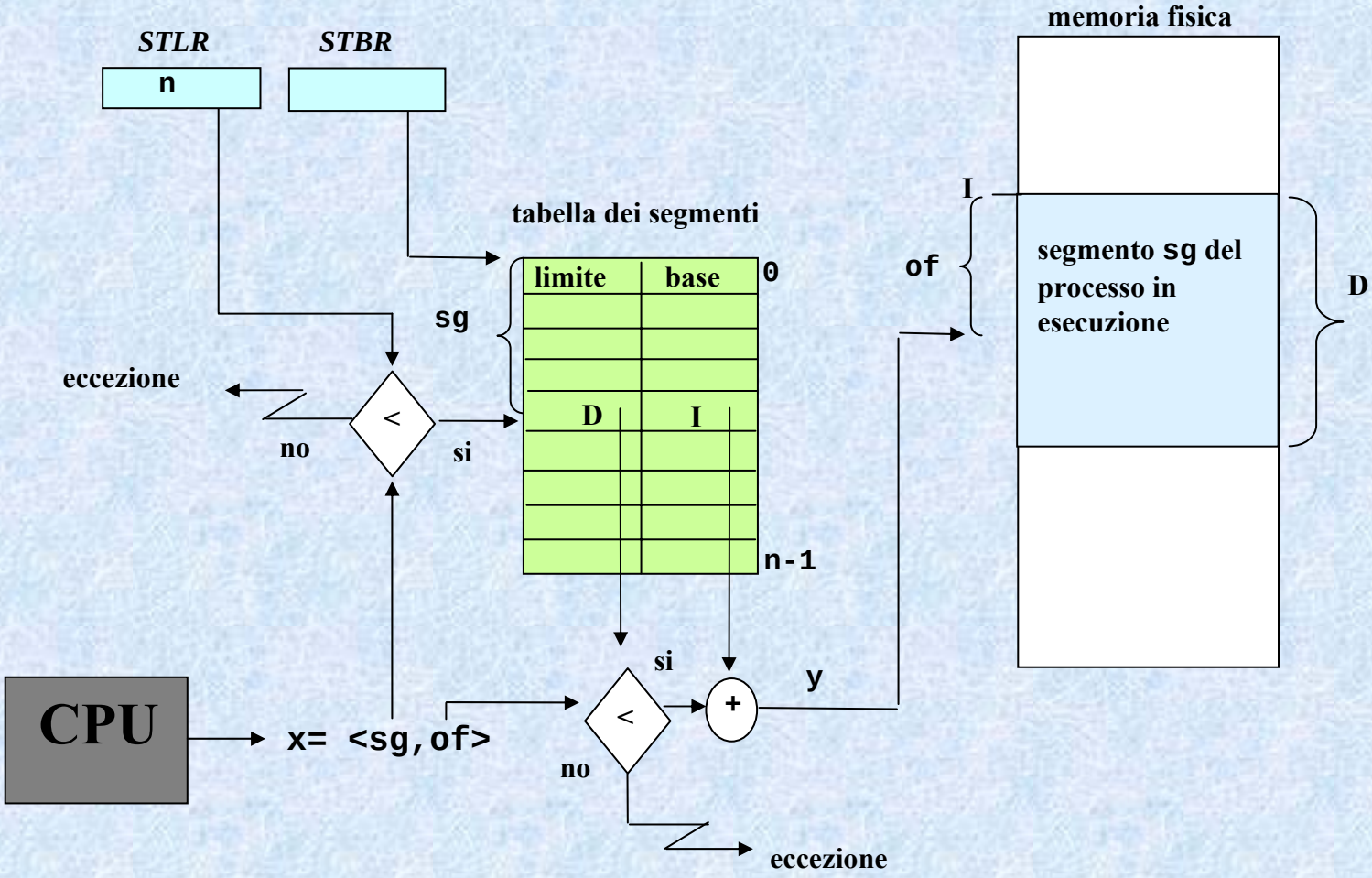


Spazio virtuale segmentato

Segmentazione

- Indirizzi virtuali a due dimensioni: x = <sg, of>.
- Un tabella dei segmenti per ogni processo: per ogni segmento contiene l'indirizzo a partire dal quale il segmento è allocato in memoria (**base**) e le sue dimensioni (**limite**).
- Il registro di macchina ***STBR*** (Segment Table Base Register) contiene l'indirizzo della tabella dei segmenti del processo in esecuzione.
- Il registro di macchina ***STLR*** (Segment Table Length Register) contiene le dimensioni della tabella dei segmenti del processo in esecuzione.

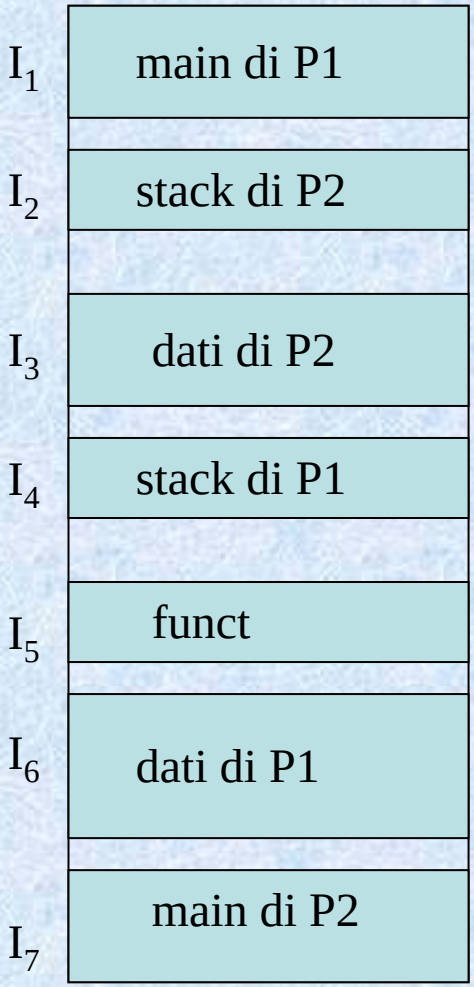
Traduzione degli indirizzi segmentati



Condivisione di segmenti

	n°	base	limite
main	0	I_1	L_1
funct	1	I_5	L_5
dati	2	I_6	L_6
stack	3	I_4	L_4

Processo P_1



	n°	base	limite
main	0	I_7	L_7
funct	1	I_5	L_5
dati	2	I_3	L_3
stack	3	I_2	L_2

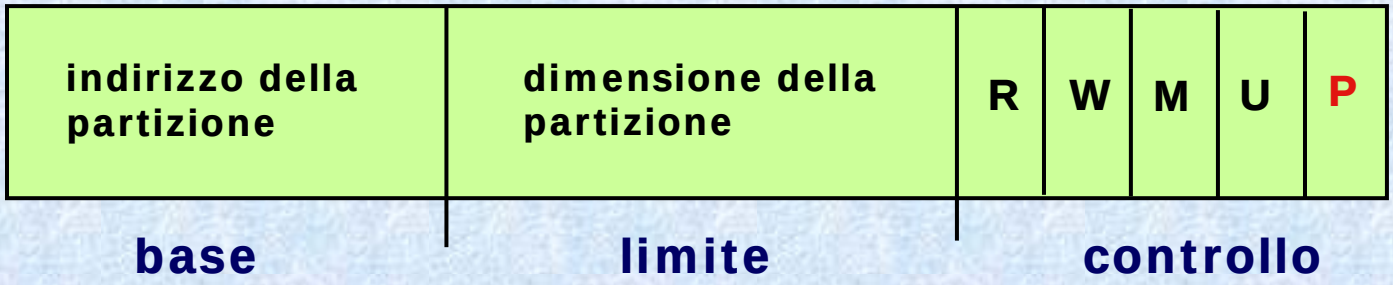
Processo P_2

Segmentazione a domanda

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
DINAMICA	CONTIGUA	SEGMENTATO	A DOMANDA

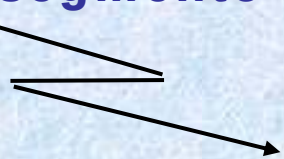
Segmentazione a domanda

descrittore di segmento



- R e W: diritti di accesso in lettura e scrittura
- M e U: bit di modifica e di uso (per gli algoritmi di sostituzione)
- P: bit di presenza

  - P = 1: semento presente in memoria
  - P = 0: segmento non presente in memoria



segment fault

Paginazione

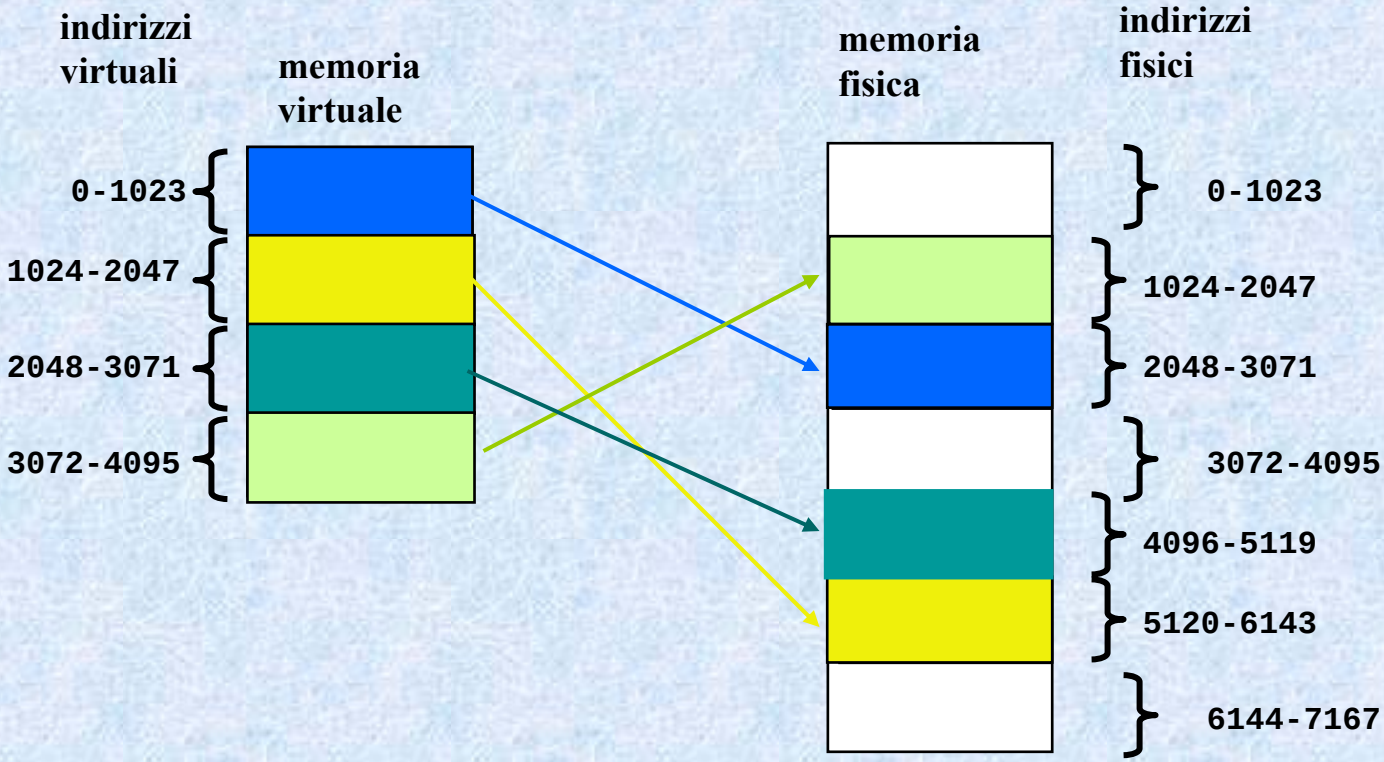
rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
DINAMICA	NON CONTIGUA	UNICO	UNICO

Paginazione

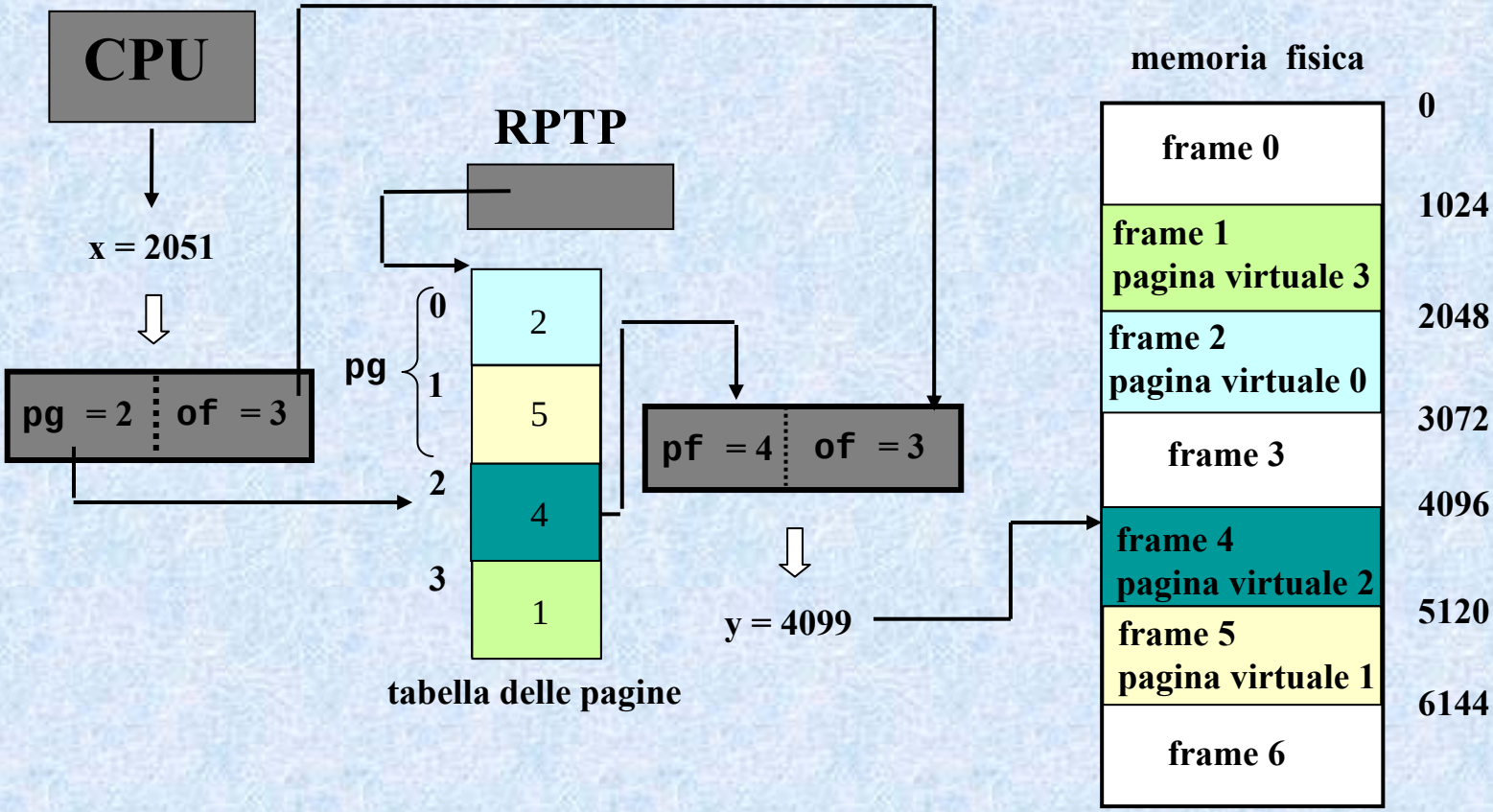
- Lo spazio virtuale è suddiviso in pagine di dimensione prestabilita;
- Lo spazio fisico è suddiviso in blocchi (o frames) di dimensione uguale a quella della pagina;
- Ogni pagina viene allocata in un blocco; pagine consecutive possono essere allocate in blocchi non consecutivi;
- Per tradurre un indirizzo virtuale nel corrispondente indirizzo fisico è necessaria una tabella (tabella delle pagine) che registra la corrispondenza tra pagine e blocchi.

Paginazione

Esempio con dimensione della pagina di 1024 locazioni



Traduzione degli indirizzi



RPTP: registro puntatore alla tabella delle pagine

Condivisione di pagine

Pagina	Frame
0	1
1	4
2	2
3	6

Memoria virtuale del processo P_1

0	
1	Codice condiviso
2	Dati privati di P1
3	Stack privato di P2
4	Codice condiviso
5	Dati privati di P2
6	Stack privato di P1

Memoria fisica

Pagina	Frame
0	1
1	4
2	5
3	3

Memoria virtuale del processo P_2

Paginazione a domanda

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
DINAMICA	NON CONTIGUA	UNICO	A DOMANDA

Paginazione a domanda

campo pagina fisica	campo controllo				
indice della pagina fisica se P=1; indirizzo su disco se P=0	R	W	U	M	P

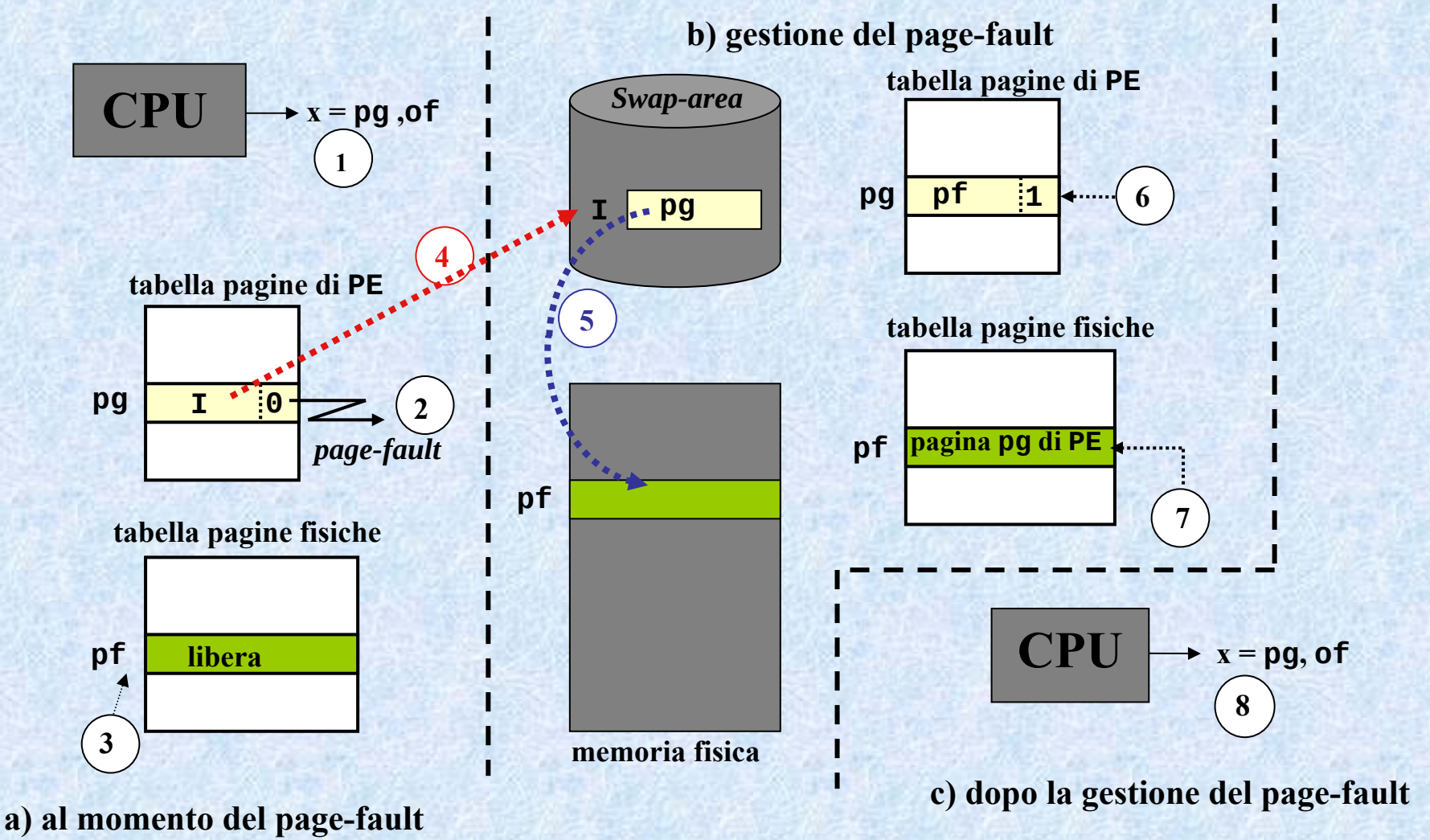
elemento della tabella delle pagine

- R e W: diritti di accesso in lettura e scrittura
- M e U: bit di modifica e di uso (per gli algoritmi di sostituzione)
- P: bit di presenza

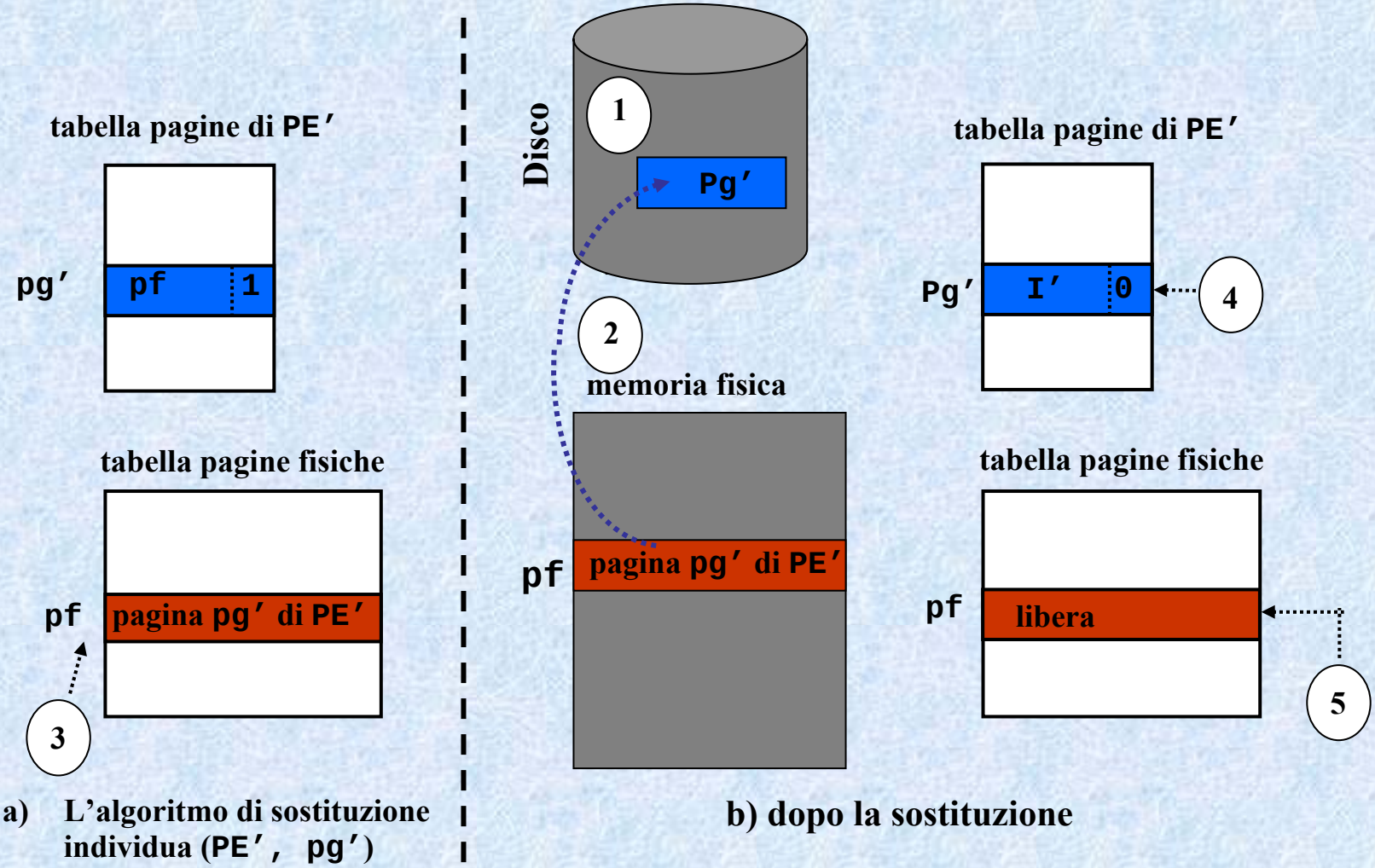
  - P = 1: pagina presente in memoria
  - P = 0: pagina non presente in memoria

→ **page-fault**

Gestione di un page-fault



Sostituzione di pagine



Algoritmi di sostituzione

Problema: tra le pagine caricate in memoria, selezionarne una per lo scaricamento

Obiettivo:

1) minimizzare il numero di errori di pagina (page faults)

- Se la pagina scaricata sarà riferita in futuro, lo scaricamento determinerà un errore di pagina
- Possibilità di *thrashing*

2) a parità di altre condizioni, selezionare una pagina non modificata durante la residenza in memoria

- Lo scaricamento di pagine modificate è più costoso, perchè è necessario il salvataggio su disco

Algoritmi di sostituzione

Algoritmo ottimo:
seleziona la pagina la cui distanza futura è massima

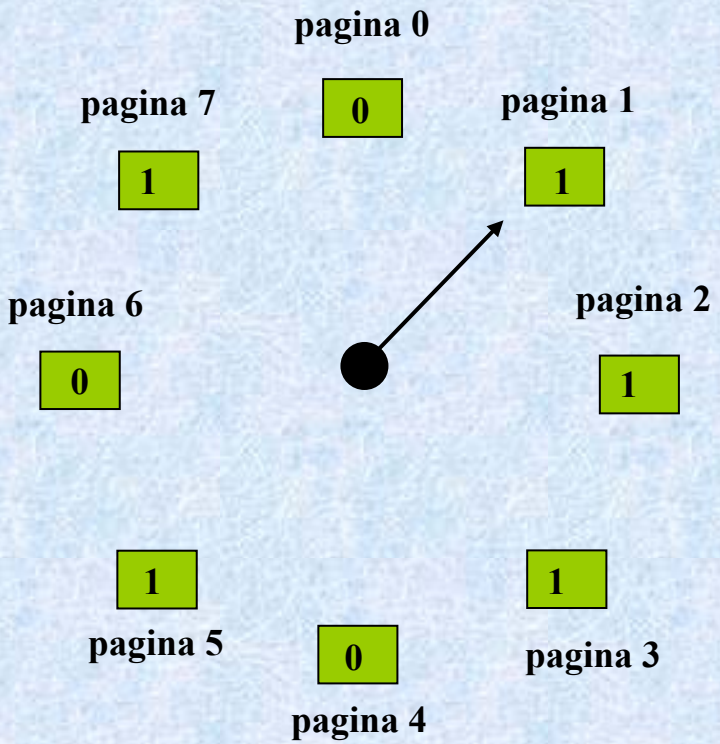
- Distanza futura: tempo che intercorrerà tra quello attuale e quello del primo riferimento futuro
- Ovviamente irrealizzabile

Algoritmo *Least Recently Used* (LRU):
seleziona la pagina la cui distanza passata è massima

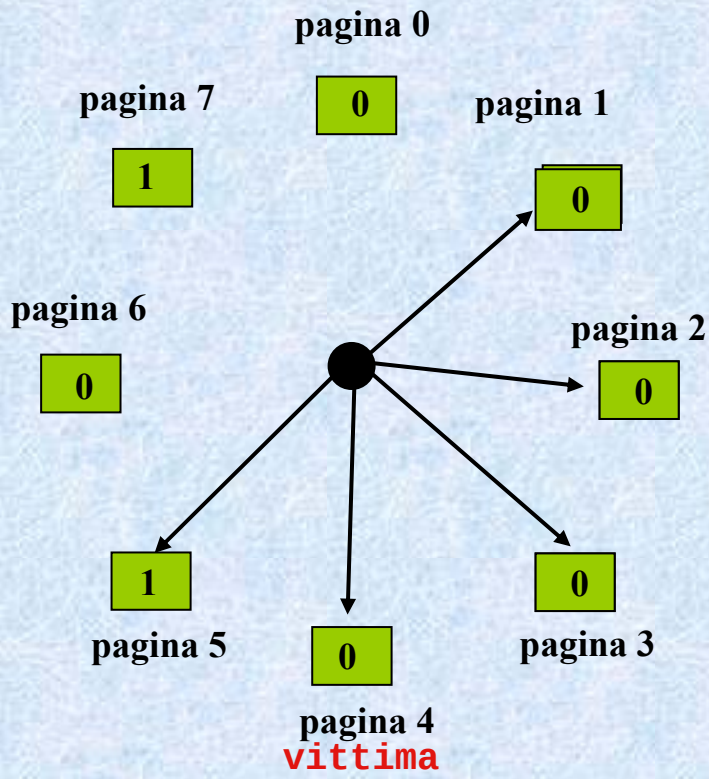
- Distanza passata: tempo intercorso tra l'ultimo riferimento passato e quello attuale
- Approssima probabilisticamente l'algoritmo ottimo sulla base del *principio di località*
- Realizzazione relativamente costosa

Algoritmo di sostituzione "*second-chance*"

- **Approssimazione dell'algoritmo LRU**



a) all'inizio dell'algoritmo



b) Come procede l'algoritmo

Algoritmi di sostituzione globali o locali

Algoritmi globali:

la pagina è selezionata tra tutte quelle residenti in memoria, indipendentemente dal processo di appartenenza

- Distanza passata definita come tempo assoluto
- Provocano *thrashing* per i processi lenti

Algoritmi locali

la pagina è selezionata tra quelle di uno specifico processo

==> normalmente il processo che commette l'errore di pagina

- Distanza passata definita come tempo *virtuale*
- Obiettivo: mantenere in memoria il *working set* del processo, definito come l'insieme di pagine che il processo ha riferito nel passato recente
- Evitano il thrashing per tutti i processi

Gestione degli errori di pagina con algoritmi di sostituzione locali

Per ogni processo:

- *Insieme di lavoro (working set):* insieme delle di pagine che il processo ha riferito nel passato recente, di cardinalità *#(Insieme di lavoro)*
- *Insieme residente*: insieme delle pagine del processo caricate in memoria, di cardinalità #(*Insieme residente*)

Gestione elementare con working set statico

Per ogni processo *P*, assegnata staticamente *#(Insieme di lavoro)*

Quando *P* commette un errore di pagina:

- se #(*Insieme residente*)< #(*Insieme di lavoro)*, la pagina riferita è caricata in memoria
- altrimenti viene scaricata dalla memoria una pagina di *P* selezionata dall'algoritmo di sostituzione locale e al suo posto viene caricata la pagina riferita con errore di pagina.

Modalità evolute di gestione

- eliminazione preventiva di pagine dall'insieme residente
- definizione dinamica di #(*Insieme residente*) in base alla frequenza di errori di pagina

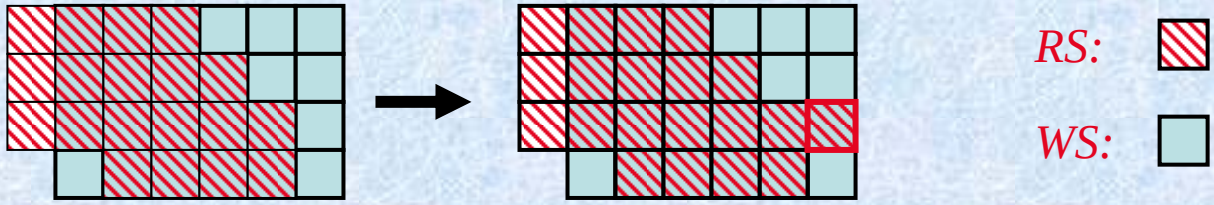
Gestione degli errori di pagina con algoritmi di sostituzione locali

Gestione elementare con *#WS statica*

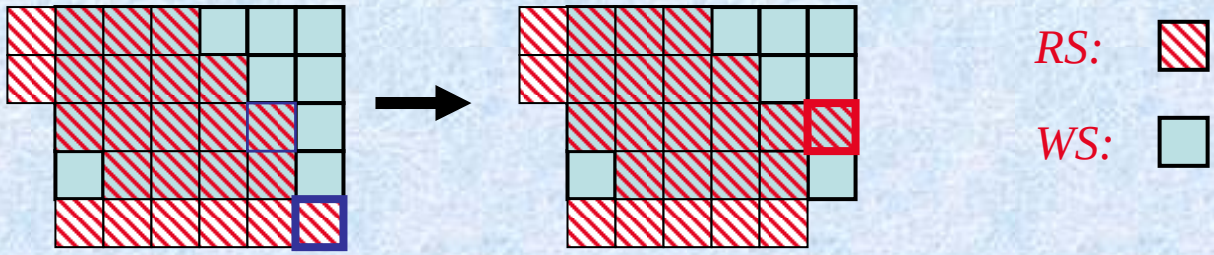
Per ogni processo *P*, assegna staticamente *#WS*

Quando *P* commette un errore di pagina:

- se *#RS* < *#WS* si assegna un blocco, dove si carica la pagina riferita *(#RS= #RS+1)*



- altrimenti si scarica una pagina di *P* selezionata dall'algoritmo di sostituzione locale e nel blocco di *RS reso disponibile* si carica la pagina riferita *(#RS invariata).*



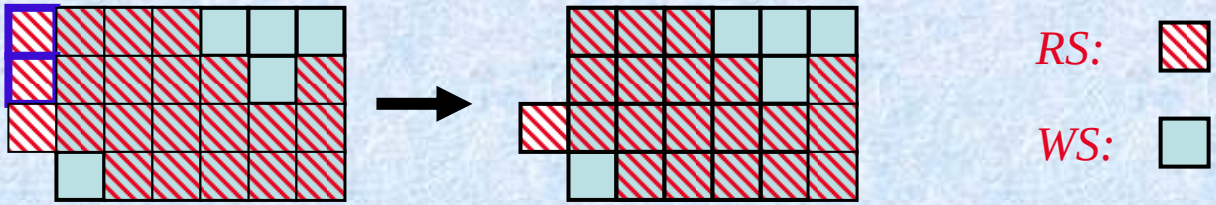
--> l'algoritmo di sostituzione dovrebbe scegliere una pagina in RS-WS

Gestione degli errori di pagina con algoritmi di sostituzione locali

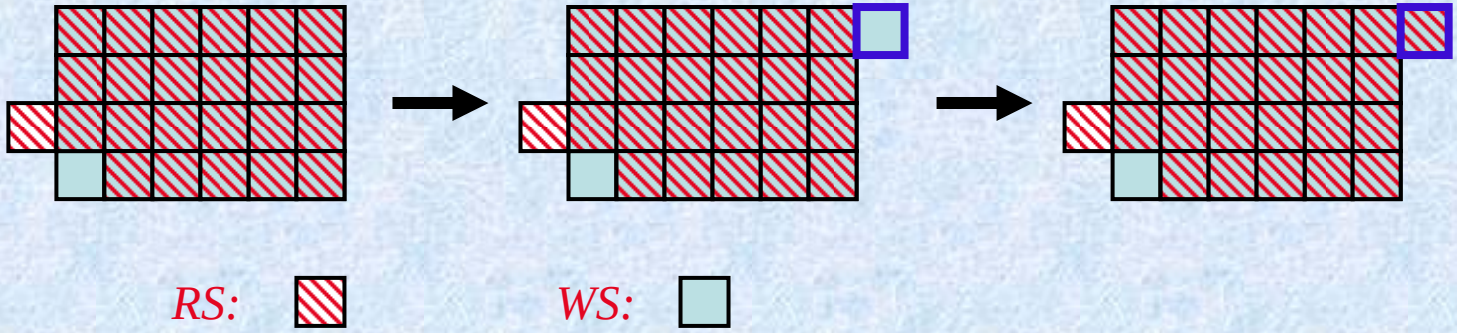
Modalità evolute di gestione

- *#WS statica con eliminazione preventiva di pagine dall'insieme residente*

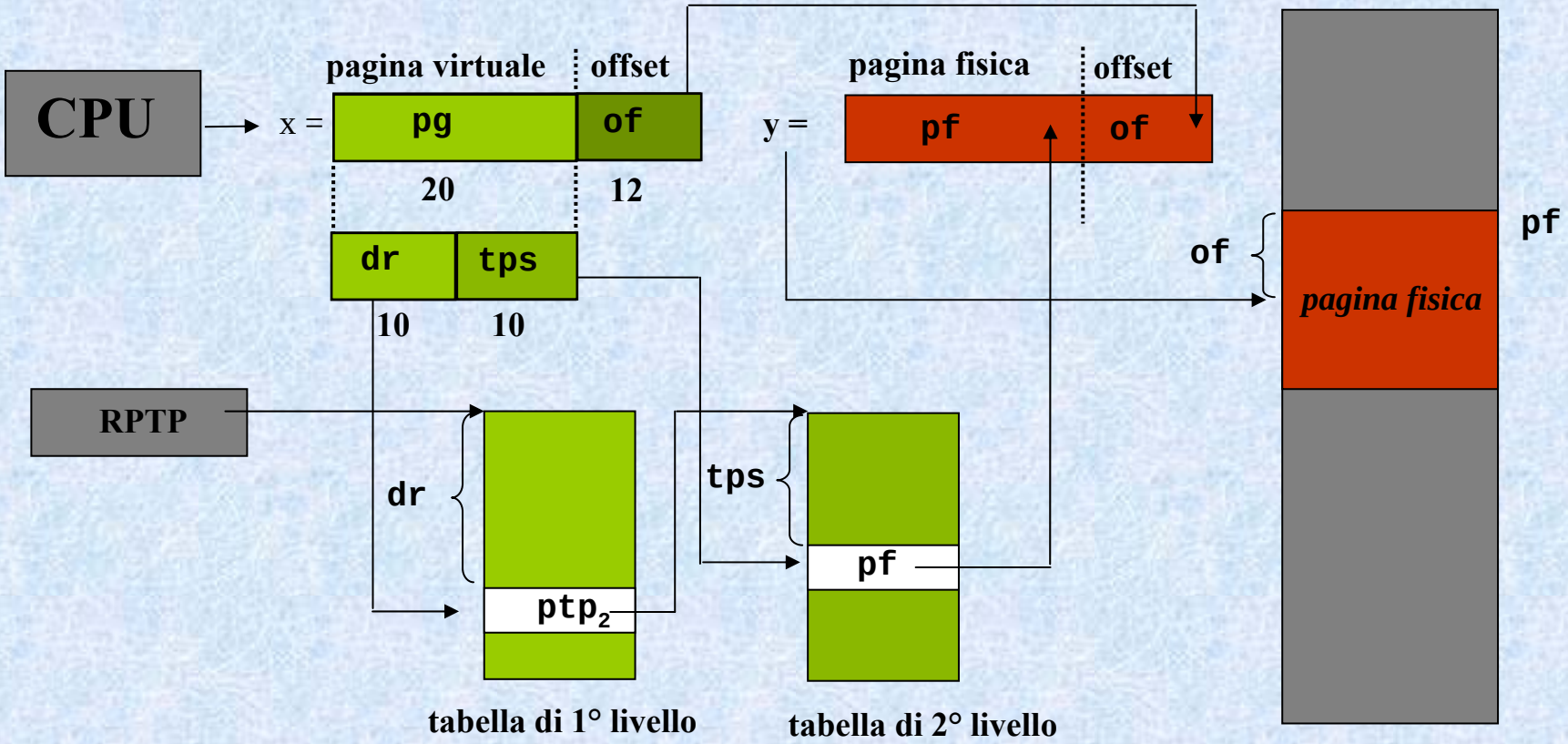
  --> l'algoritmo di sostituzione dovrebbe scegliere pagine in RS-WS



- *definizione dinamica di #WS in base alla frequenza degli errori di pagina*



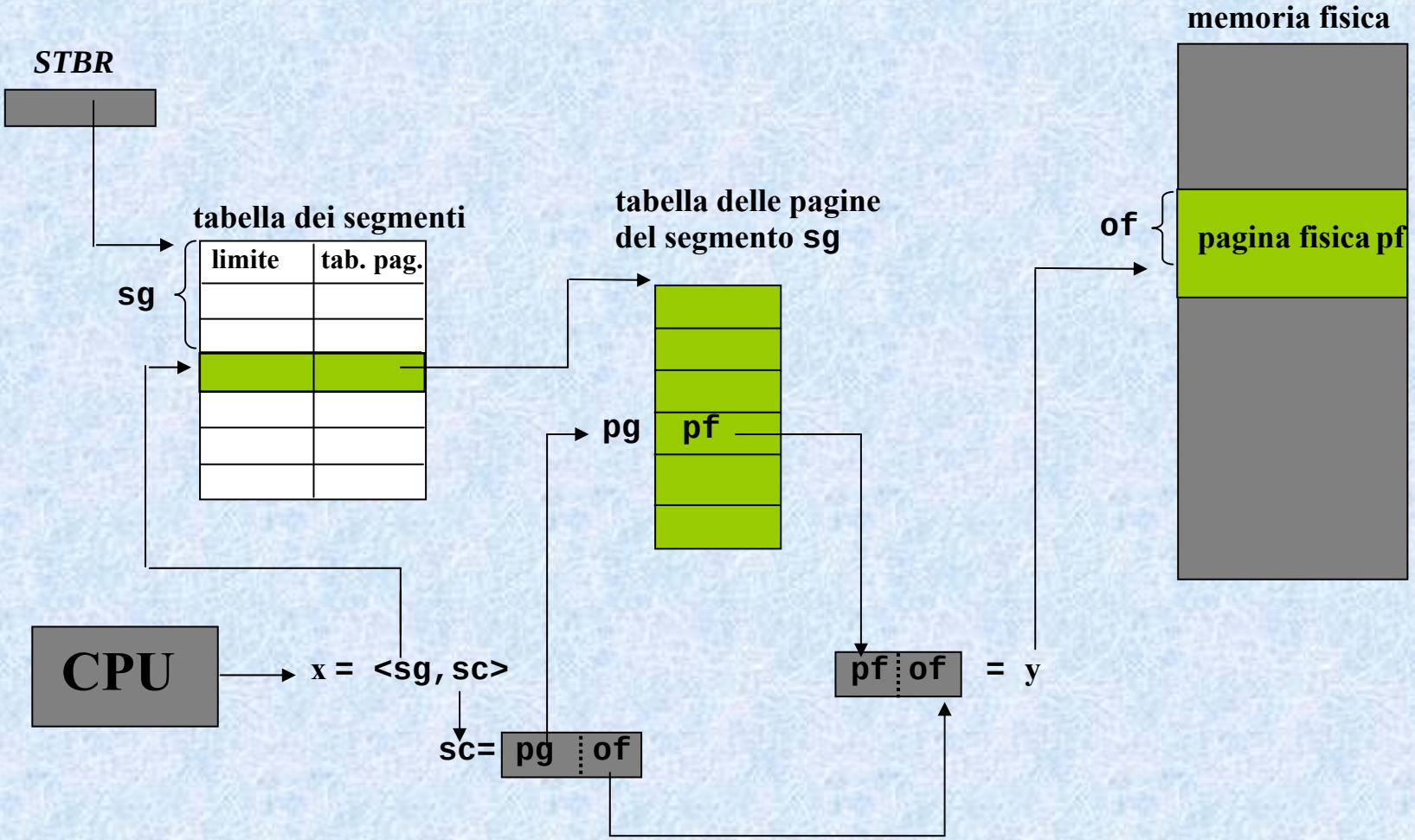
Paginazione a due livelli



Segmentazione con paginazione

rilocazione degli indirizzi	allocazione della memoria	spazio virtuale	caricamento
DINAMICA	NON CONTIGUA	SEGMENTATO	A DOMANDA

Segmentazione con paginazione: traduzione degli indirizzi



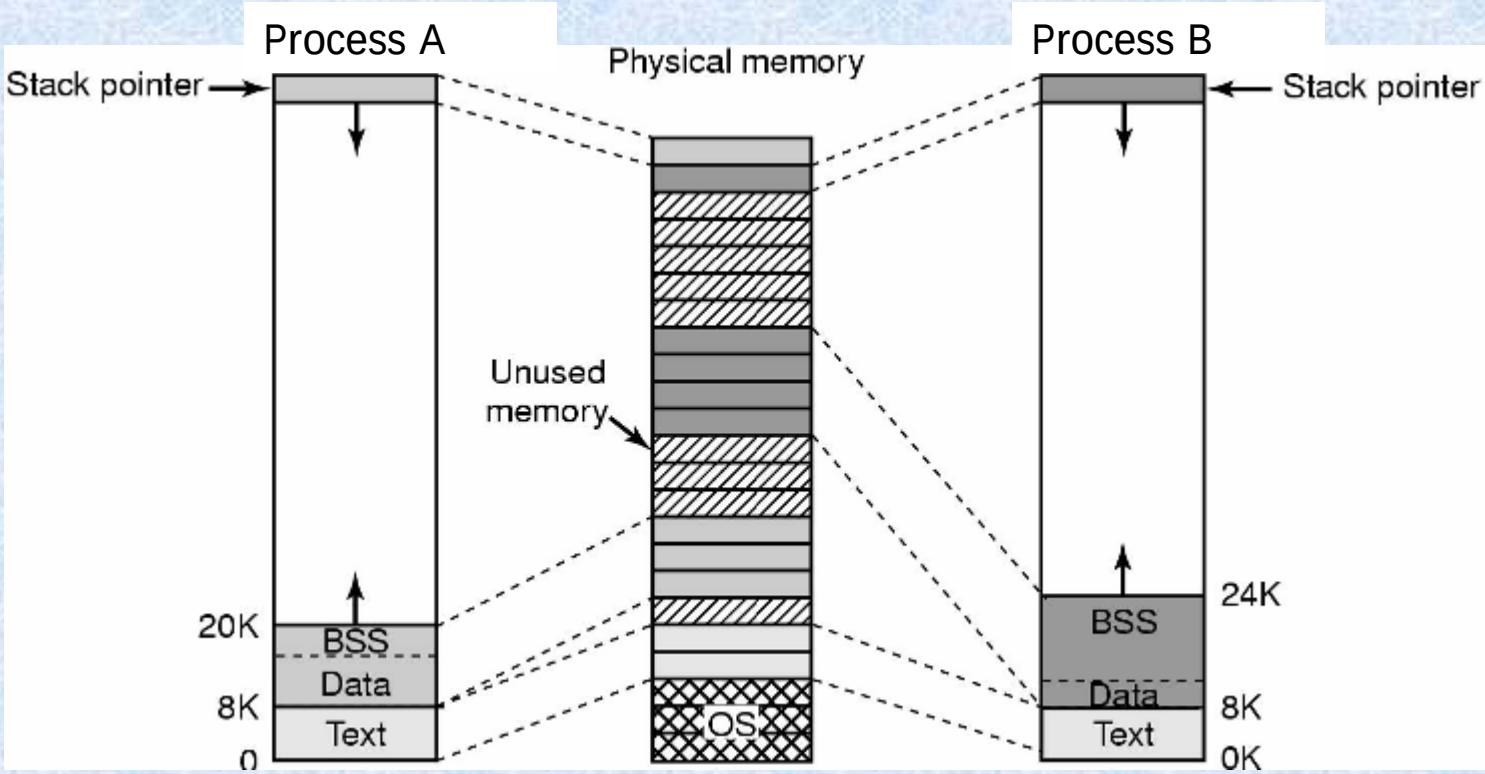
Gestione della memoria in UNIX

- Da BSD v.3 in poi:
  - segmentazione paginata
  - **memoria virtuale** con paginazione a domanda

Paginazione a domanda :

- ***Core map*:** struttura dati interna al kernel che descrive lo stato di allocazione dei blocchi e che viene consultata in caso di page fault.
- **Sostituzione delle pagine:** algoritmo Second Chance

UNIX: organizzazione della Memoria



UNIX: files mappati in memoria e condivisi

UNIX: system calls per la gestione della memoria

System call	Description
s = brk(addr)	Change data segment size
a = mmap(addr, len, prot, flags, fd, offset)	Map a file in
s = unmap(addr, len)	Unmap a file

- **s** è un codice di errore
- **b** e **addr** sono indirizzi di memoria
- **len** è una lunghezza
- **prot** controlla la protezione
- **flags** bit vari
- **fd** è un descrittore di file
- **offset** è un offset su un file

1) Core Map + Tabelle delle pagine

SO	SO	SO	SO	SO	SO		A,1 2	B,0 10	C,1 3		B,6 5	C,7 8		C,3 6	A,5 9	C,5 7	B,2 1	A,7 4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Processo, pagina
Tempo ultimo riferimento

Blocco

Pagina	Blocco
0	-
1	7
2	-
3	-
4	-
5	15
6	-
7	18

Processo A

Pagina	Blocco
0	8
1	-
2	17
3	-
4	-
5	-
6	11
7	-

Processo B

Pagina	Blocco
0	-
1	9
2	-
3	14
4	-
5	16
6	-
7	12

Processo C

2) Core Map = Tabella delle pagine inversa

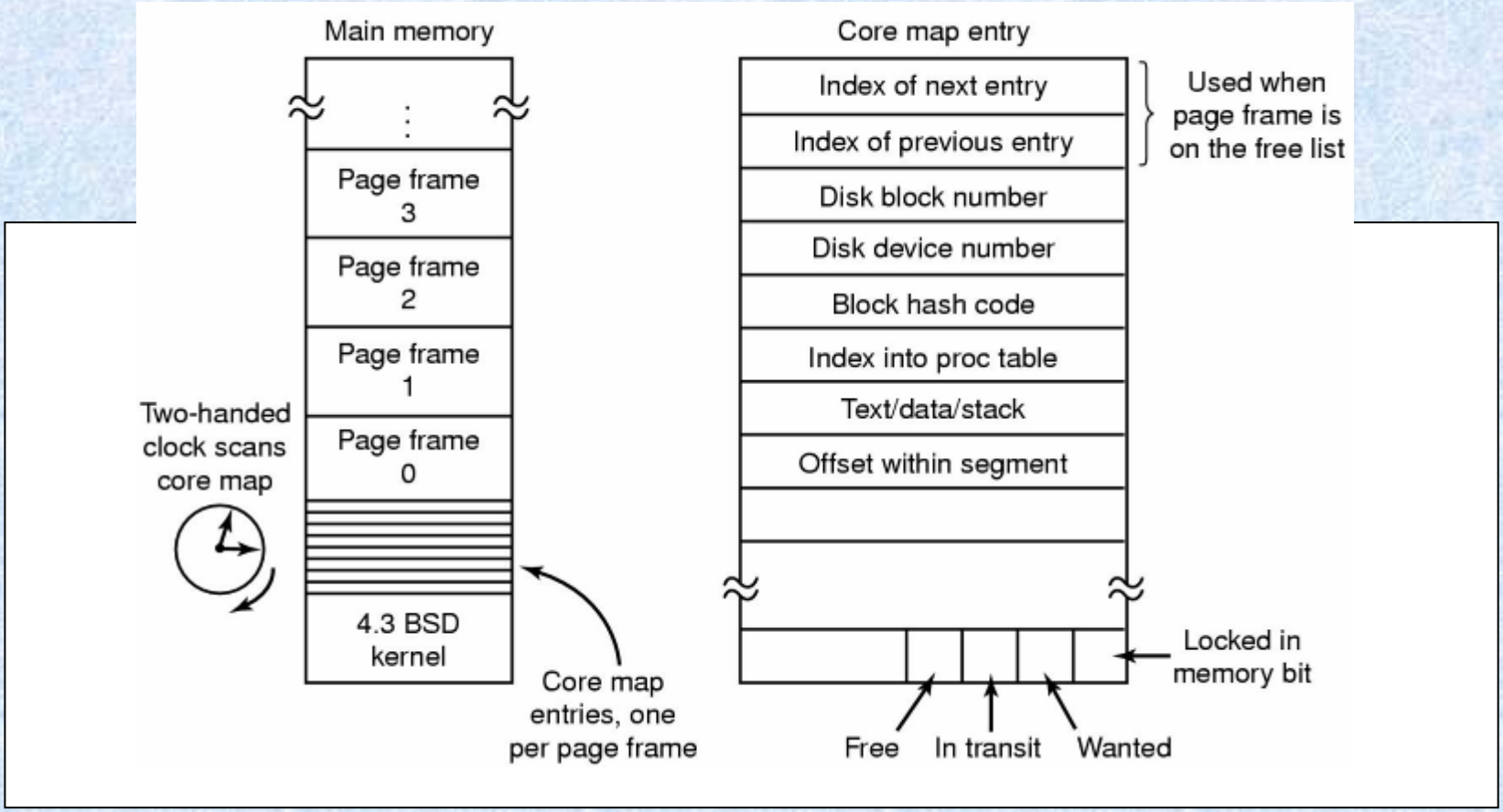
SO	SO	SO	SO	SO	SO		A,1 2	B,0 10	C,1 3		B,6 5	C,7 8		C,3 6	A,5 9	C,5 7	B,2 1	A,7 4
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Processo, pagina
Tempo ultimo riferimento

Blocco

Paginazione in UNIX

- Usa una tabella delle pagine inversa *(core map)*



Sostituzione delle pagine in UNIX (BSD)

Per selezionare pagine da scaricare:
- algoritmo *SecondChance* (globale)
- o la sua variante *Two-handed Clock Algorithm*

Eseguito da *PageDaemon*
PageDaemon:
- utilizza parametri *lotsfree, desfree, minfree con lotsfree > desfree > minfree*
- interviene periodicamente

Algoritmo di *PageDaemon* (esistono diverse varianti nelle diverse versioni di UNIX)
- **se** *NumeroBlocchiLiberi* ≥ *lotsfree* ritorna senza scaricare pagine
- **se** *minfree*≤*NumeroBlocchiLiberi*< *lotsfree*
  oppure *NumeroBlocchiLiberi*< *minfree* **e** *media(NumeroBlocchiLiberi,* ΔT*)*≥ *desfree*
  scarica pagine fino ad ottenere *NumeroBlocchiLiberi*= *lotsfree*+ *k*, con k≥0
- **se** *NumeroBlocchiLiberi*< *minfree* e *media(NumeroBlocchiLiberi,* ΔT*)* < *desfree*
  esegue swapout di uno o più processi

Sostituzione delle pagine in UNIX (BSD): relazione con la teoria del Working Set

Se *NumeroBlocchiLiberi< minfree*

==> elevata frequenza di errori di pagina nell'ultimo periodo di attivazione di *Page Daemon*

==> esistono processi con *#RS < #WS:* provocano thrashing

Se *media(NumeroBlocchiLiberi, ΔT)< desfree*

==> il fenomeno persiste da alcuni periodi

- per effetto dell'algoritmo di sostituzione globale, quando si carica una pagina di un processo che ha commesso errore di pagina si tende a ridurre *#RS* per altri processi, aggravando il thrashing

Eseguendo *swapout* si liberano risorse di memoria

==> *i processi con #RS < #WS,* per effetto dei propri errori di pagina, possono incrementare *#RS*

Swapout e Swapin dei processi in UNIX (BSD):

Swapout

Se *NumeroBlocchiLiberi*< *minfree* e *NumeroMedioBlocchiLiberi*< *desfree* (media calcolata in un opportuno intervallo di tempo),

PageDaemon:

- Seleziona processi candidati allo scaricamento con criterio basato su:
  - tempo trascorso senza andare in esecuzione
  - quantità di memoria necessaria
- Esegue Swapout di uno o più processi fino a ottenere *NumeroBlocchiLiberi*≥ *lotsfree*

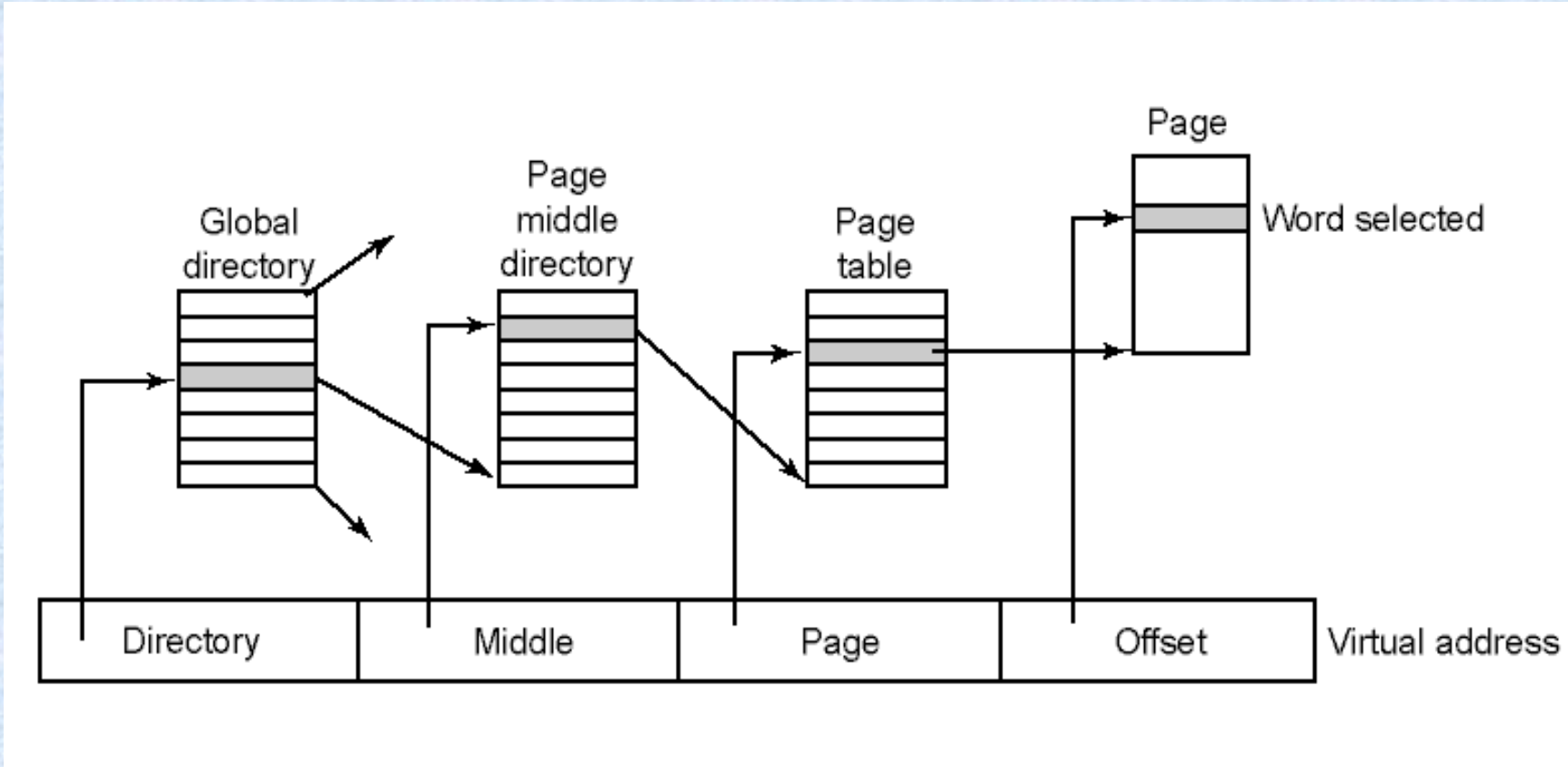
Swapin

Se *NumeroBlocchiLiberi* sufficientemente grande, *PageDaemon:*

- Seleziona uno o più processi in stato *swapped,* con criteri basati su:
  - tempo trascorso in stato *swapped*
  - quantità di memoria necessaria
- Esegue swapin di uno o più processi, rispettando la condizione *NumeroBlocchiLiberi*≥ *lotsfree*

Paginazione in Linux

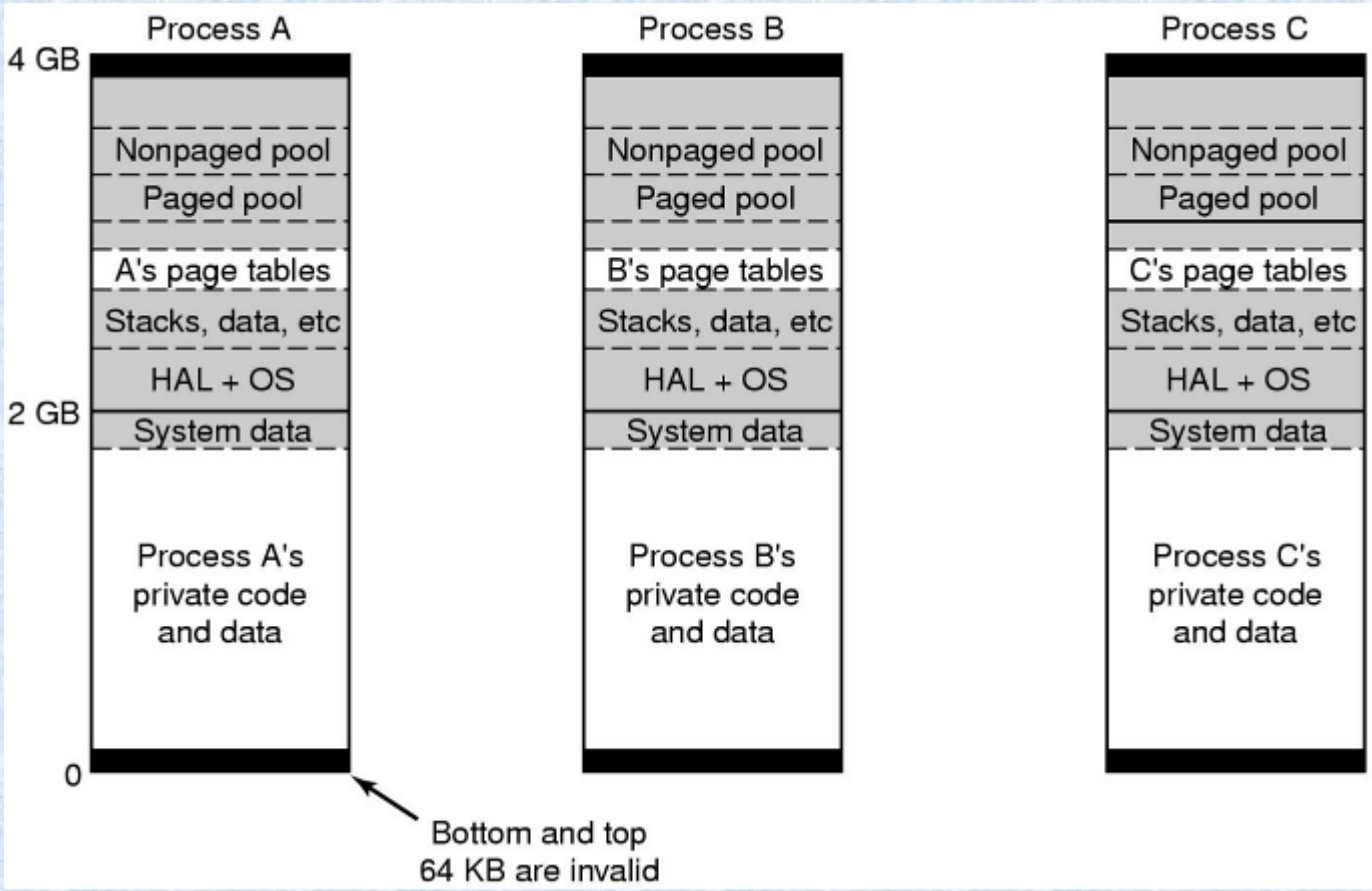
- tabelle delle pagine a tre livelli



Gestione della memoria in Windows

- Dimensione della memoria virtuale: 4 gigabyte (indirizzo virtuale di 32 bit).
- Memoria virtuale paginata (paginazione a domanda) con pagine di dimensioni fisse (le dimensioni della pagina dipendono dalla particolare macchina fisica).
- Spazio virtuale suddiviso in due sottospazi di 2 gigabyte ciascuno: il sottospazio inferiore è privato di ogni processo mentre il sottospazio superiore è condiviso tra tutti i processi e contiene il sistema operativo.

Struttura della memoria virtuale in Windows



Le aree bianche sono private; le aree scure sono condivise

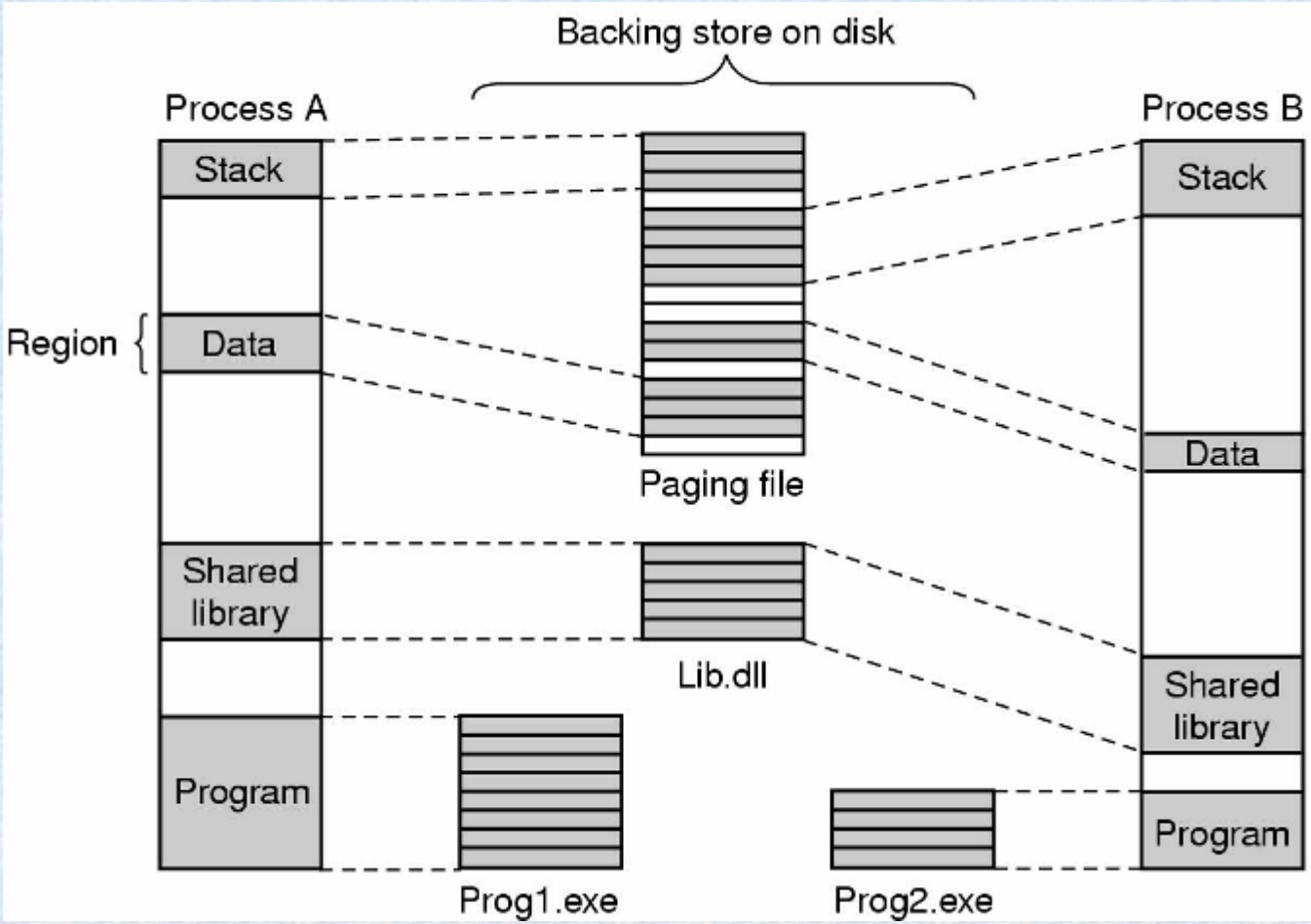
Memoria Virtuale in Windows

Spazio virtuale unico, suddiviso in *regioni*

Ogni pagina logica può essere :

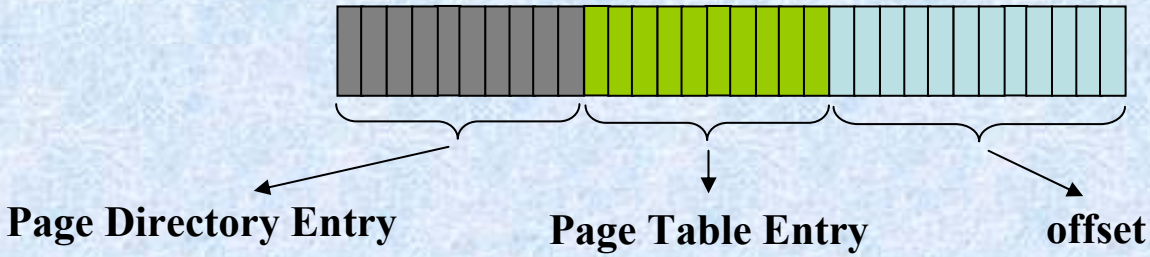
- *free* : se non è assegnata a nessuna regione
  - un accesso a una pagina free determina page fault non gestibile
- *reserved* : è una pagina non ancora in uso ma che è stata riservata per espandere una regione
  - esempio: riservata per l'espansione dello stack
  - non utilizzabile per mappare nuove regioni
  - un accesso a una pagina reserved determina page fault gestibile
- *committed* : se appartiene alle pagine in uso di una regione già mappata
  - un accesso a una pagina *committed* risulta in un page fault che determina il caricamento della pagina solo se questa *non si trova in una lista di pagine eliminata dal working set*

Windows: caricamento dei processi in memoria



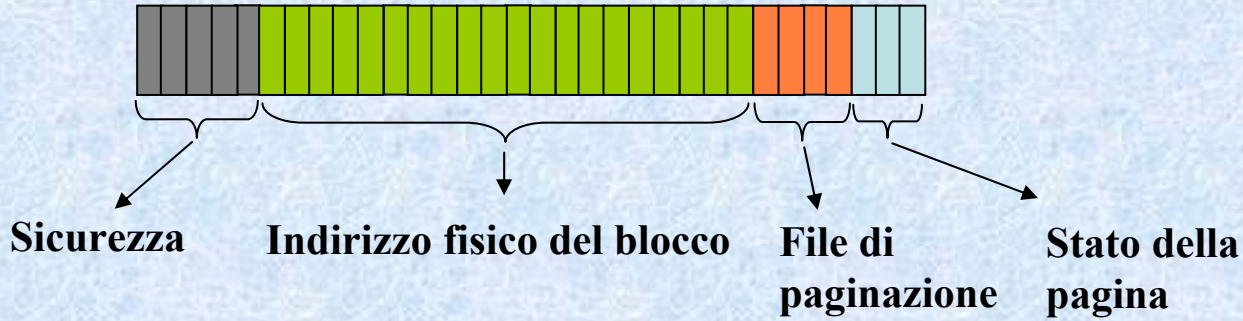
Gestione della memoria in Windows

- Meccanismo di paginazione a due livelli: la tabella di primo livello (*Page Directory*) contiene 1024 elementi (*Page Directory Entry*) ciascuno dei quali punta ad una tabella di secondo livello (*Page Table*) anch'essa di 1024 elementi (*Page Table Entry*).
- Indirizzo virtuale suddiviso in tre parti:



Gestione della memoria in Windows

- Ogni elemento di una tabella delle pagine è composto da 4 campi:



Windows: gestione degli erori di pagina (1)

Algoritmo di sostituzione *Working Set*

==> è un algoritmo (fondamentalmente) locale

==> *working set* inteso come *insieme residente del processo*

==> parametri *min* e *max*

- rispettivamente minima e massima ampiezza del working set
- valori iniziali uguali per tutti i processi, ma modificabili dal *memory manager*

Se *x* è l'ampiezza corrente del working set e si verifica un page fault, il *memory manager:*

- carica la pagina riferita in un blocco libero;
- se *x<max* assegna *x= x +1;*
- se *x=max* il numero di pagine effettivamente caricate supera *x*, *x* non viene incrementato e *l*e pagine in eccesso saranno scaricate dal *working set manager*

La disponibilità di un numero sufficiente di blocchi liberi è assicurata dai demoni *balance set manager* e *working set manager*.

Windows: gestione degli erori di pagina (2)

Balance Set Manager:

- Attivato periodicamente
- Se il numero di pagine libere è inferiore a una certa soglia, attiva *working set manager*

Working Set Manager

- considera i processi con $x>min$
- per ogni processo, considera per ogni pagina *pag* caricata in memoria il bit di uso *R*
  - se *R=1*, azzera *R* e il contatore *cont(pag)*
  - se *R=0* incrementa *cont(pag)*
- dopo la scansione dei processi, considera (globalmente) le pagine in ordine decrescente di *cont(pag)* e le rimuove (le pagine rimosse risultano non presenti in memoria)
- Le pagine rimosse sono incluse nella lista delle *pagine libere* o in quella delle *pagine azzerate*, dopo il passaggio nella lista delle *pagine in attesa* e/o in quella delle *pagine modificate*.

Se un processo riferisce una pagina presente nella lista delle *pagine in attesa* o in quella delle *pagine modificate*, recupera la pagina senza provocare un accesso al disco.

Windows: gestione degli erori di pagina (3)

Liste delle pagine e transizioni tra le liste