Memoria Virtuale

Corso di Laurea in Ingegneria Informatica

Università degli Studi di Napoli Federico II

Anno Accademico 2024/2025, Canale San Giovanni

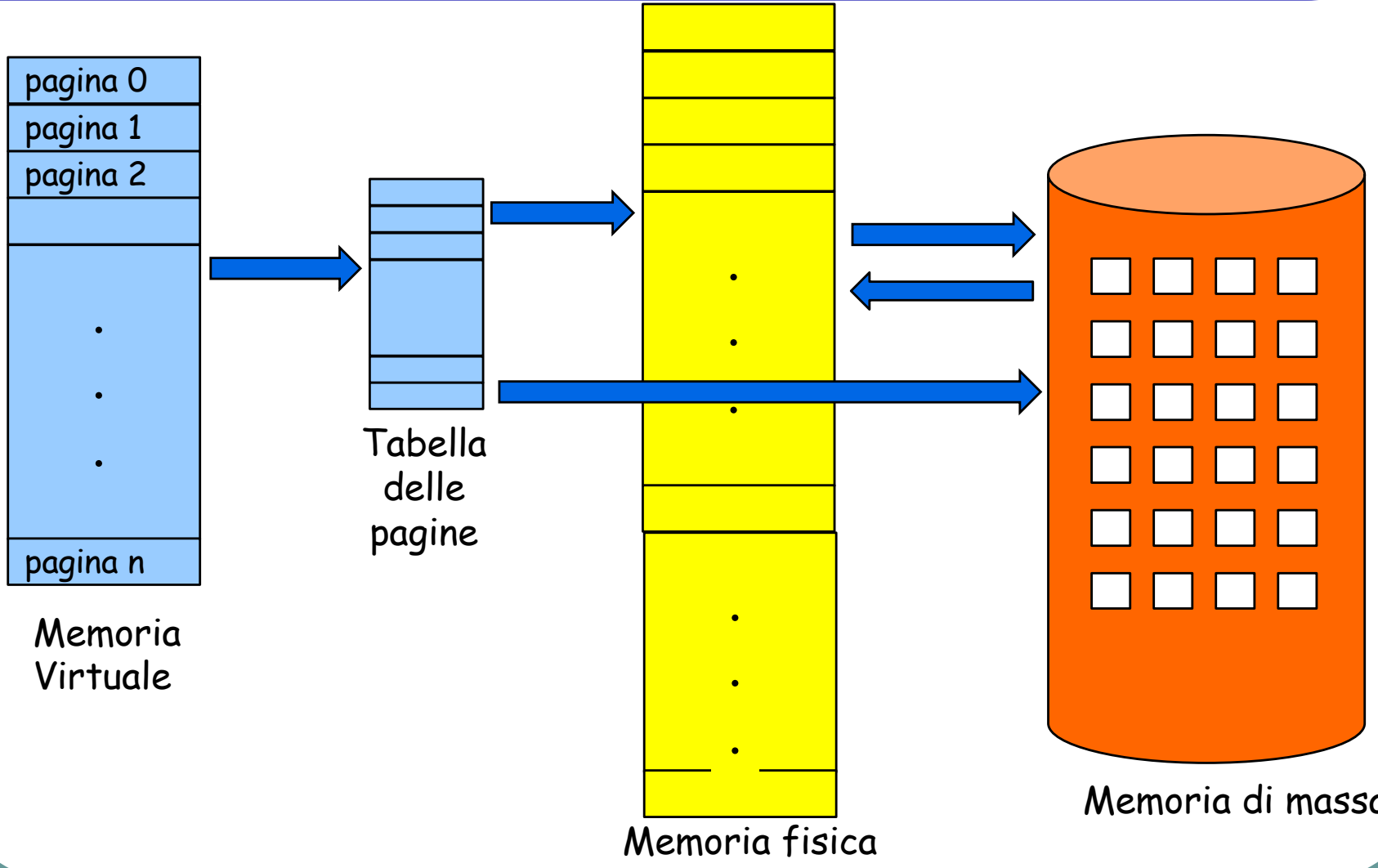
Memoria virtuale

- Sommario
  - Introduzione
  - Paginazione su richiesta
  - Sostituzione delle pagine
  - Attività di paginazione degenere (*thrashing*)
  - Modello del working set
- Riferimenti
  - P. Ancilotti, M.Boari, A. Ciampolini, G. Lipari, "Sistemi Operativi", Mc-Graw-Hill (Cap. 4, par. 4.3.3)
  - www.ostep.org, Capp. 21, 22
  - Dispensa "Sostituzione delle pagine"
  - Dispensa "Thrashing"

Introduzione

- La **memoria virtuale** separa la memoria fisica dalla memoria vista da un processo
- Può essere **più grande** della memoria fisica a disposizione
- Si realizza con la paginazione o segmentazione **"su richiesta"** (es. demand paging)

Schema di memoria virtuale



Paginazione su richiesta

Paginazione su richiesta (demand paging):
Solo le pagine effettivamente utilizzate sono caricate in memoria principale (pagine "residenti")

- Minore consumo di memoria fisica
- Più utenti e più processi possono eseguire sul sistema
- Caricamento dei processi più rapido

Bit di validità

- Si realizza usando il bit di validità (V) nella page table
- V=1 se queste condizioni sono entrambe vere:
  1. La pagina virtuale è stata allocata al processo (es. malloc())
  2. La pagina fisica corrispondente è residente in memoria
- V=0 se anche una sola condizione è falsa

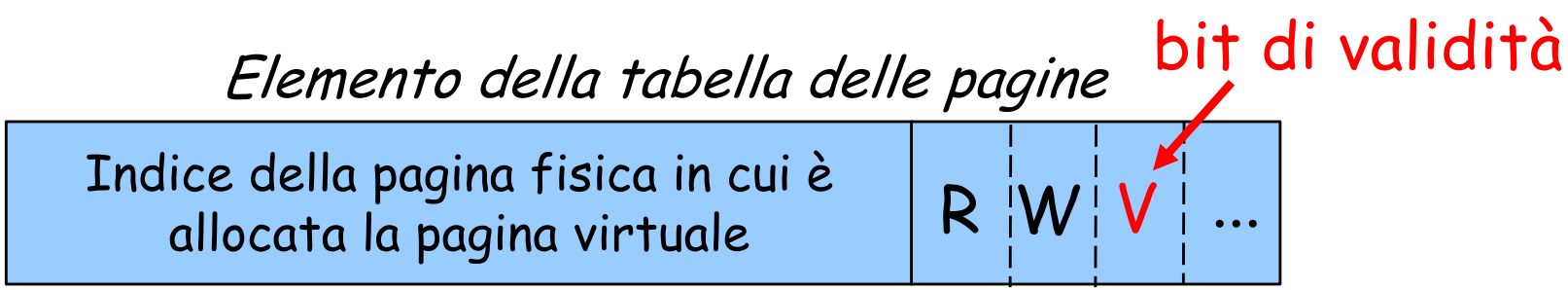
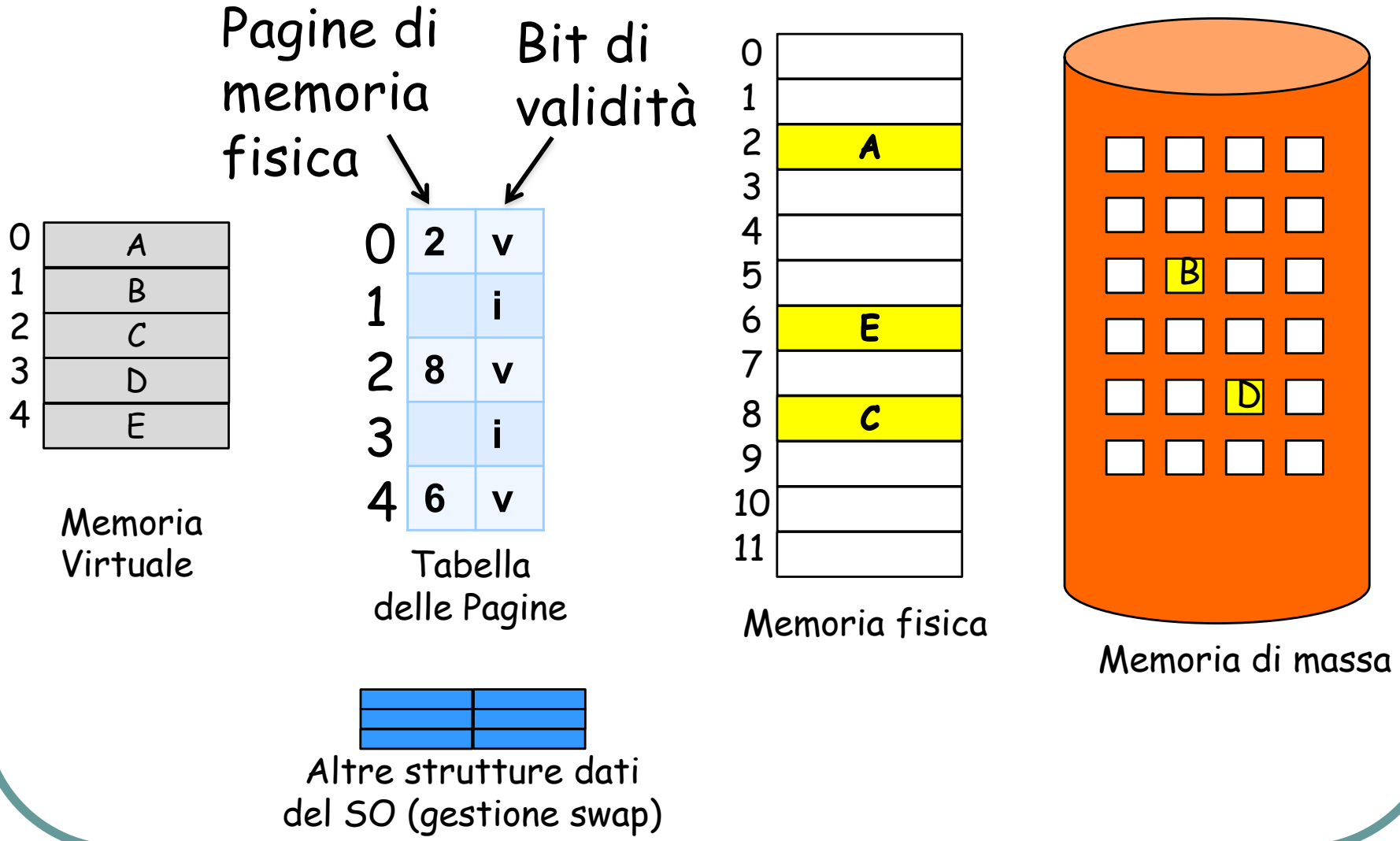
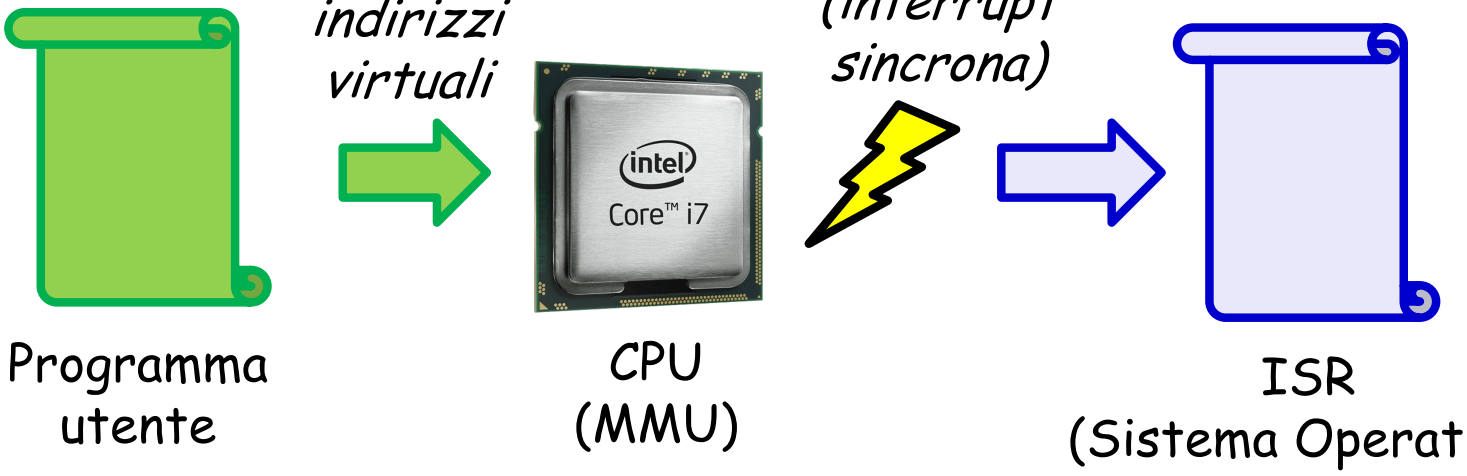


Tabella delle pagine quando alcune pagine non si trovano nella memoria centrale



Page Fault

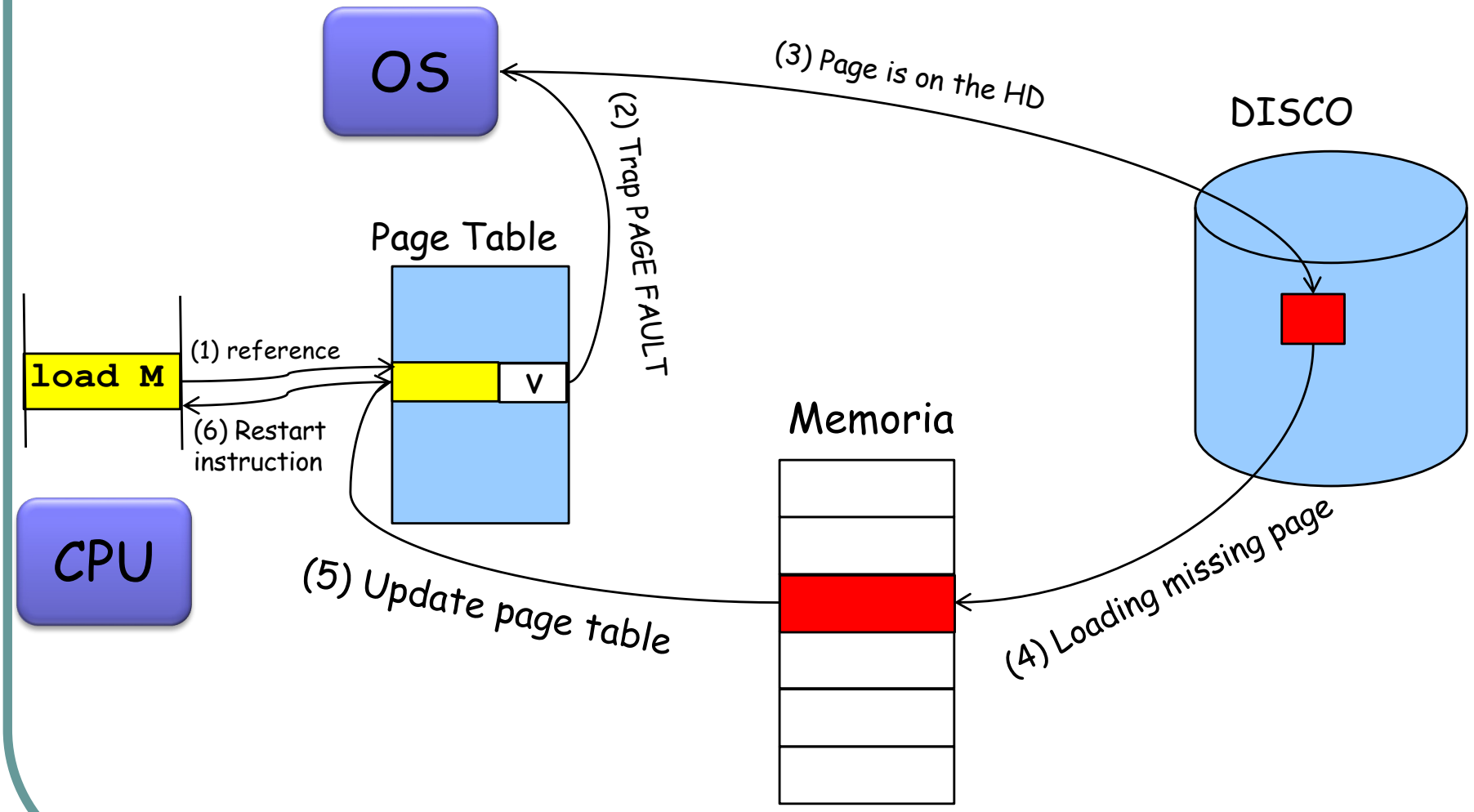
- Se un indirizzo virtuale fa riferimento a una pagina non ancora caricata in memoria:
  - La MMU nota che il bit di validità non è attivo (V=0)
  - La MMU genera una **eccezione di pagina mancante** (**page fault**)



Page Fault

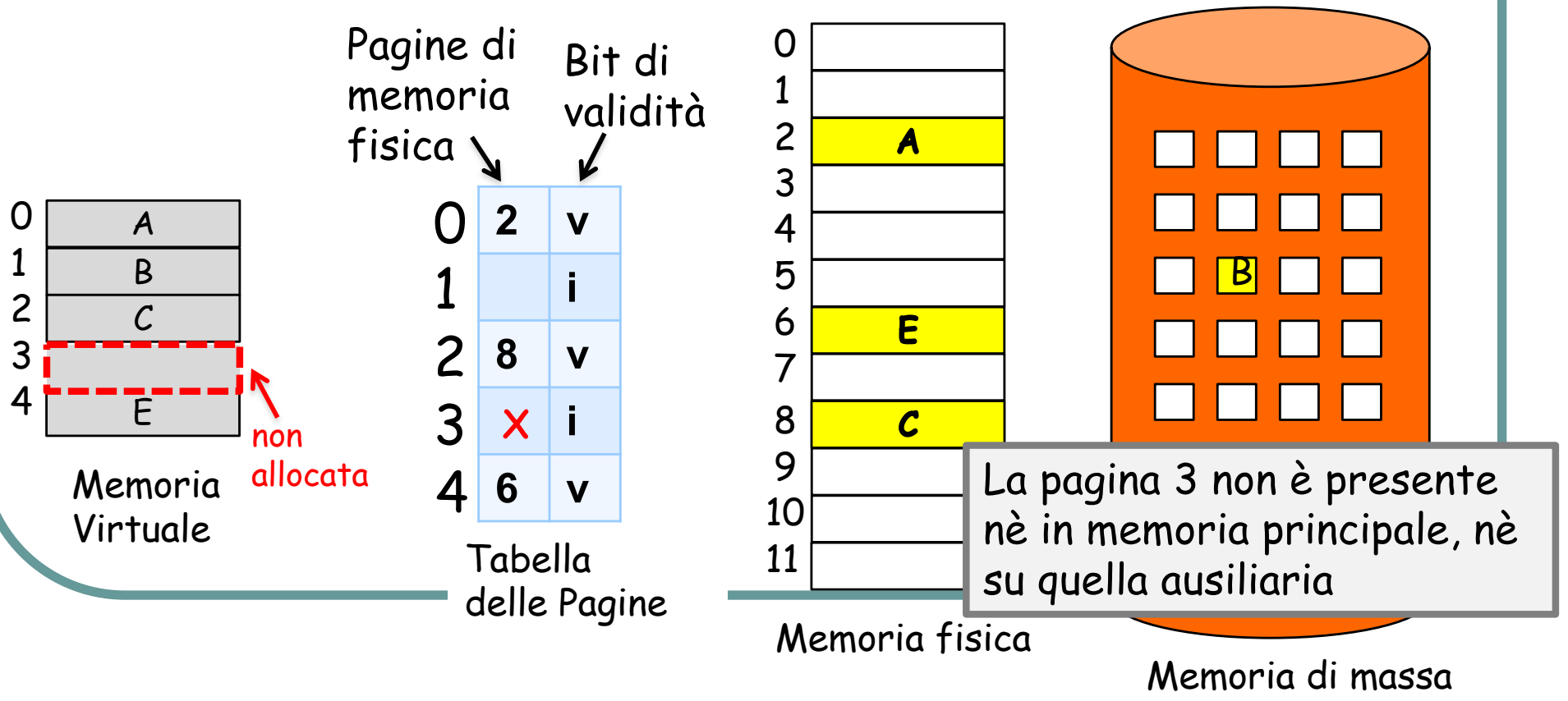
- Una ISR del SO gestisce l'eccezione
  1. Individua una pagina fisica di memoria libera
  2. Trasferisce la pagina desiderata nella memoria libera
  3. Aggiorna le tabelle, bit di validità = 1
  4. All'uscita della ISR, la CPU riavvia l'istruzione che ha causato l'eccezione

Fasi di gestione di un Page Fault



Page fault

La ISR verifica anche che l'indirizzo virtuale sia stato allocato dal processo (usando ulteriori strutture dati del SO). In caso negativo, il processo viene terminato ("Segmentation Fault")



Prestazioni della paginazione su richiesta

- Probabilità di **assenza di pagina**: $0 \leq p \leq 1.0$

- Tempo d'accesso effettivo

  EAT = (1 – p) x accesso alla memoria
  + p x (page fault overhead
  + [swap page out]
  + swap page in
  + restart overhead)

Esempio:
EAT = (1 – p) x 200ns + p x 8.000.000ns

Page Fault con sostituzione

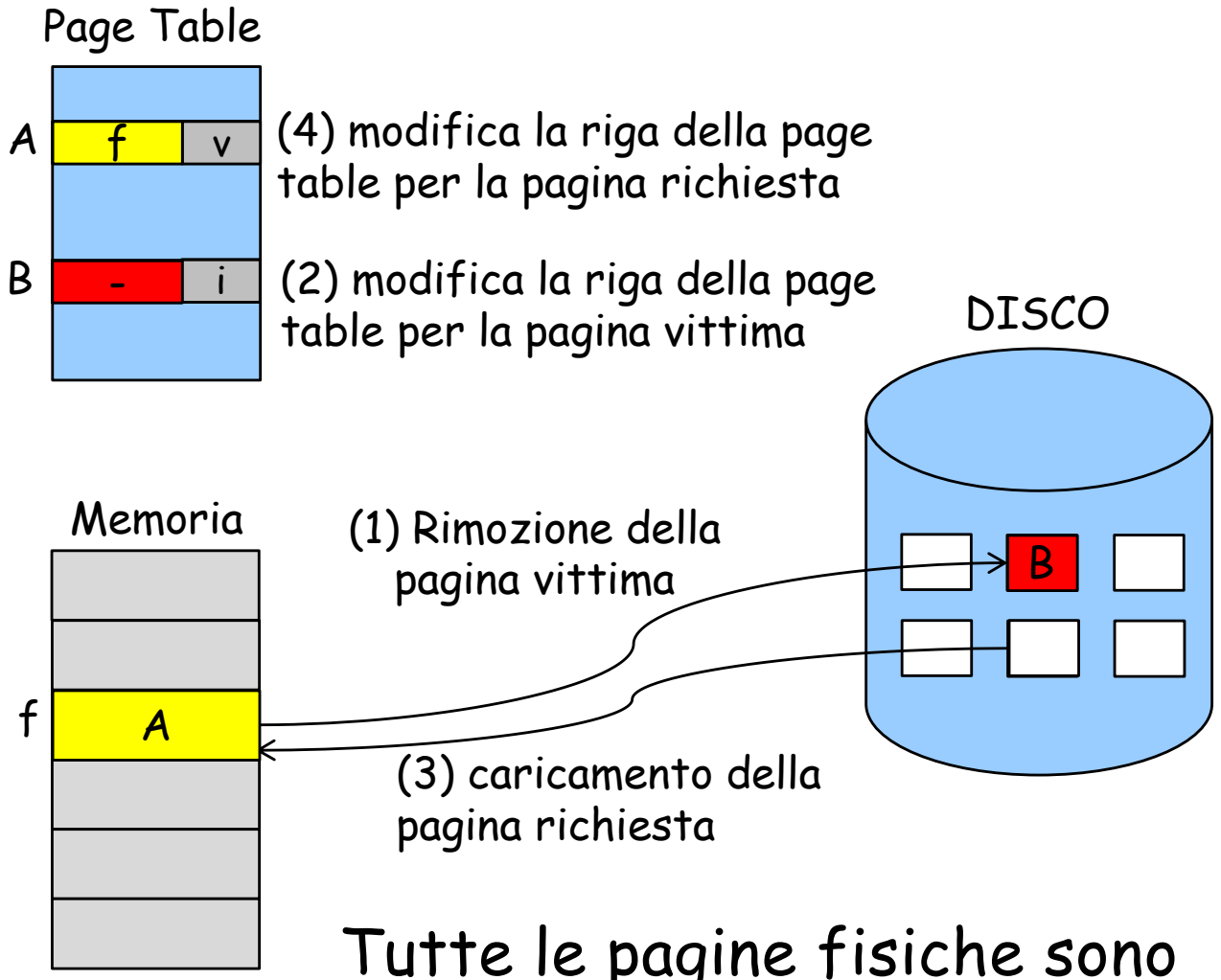
- Serve avere una pagina fisica libera in cui caricare la pagina dal disco
- Per recuperare spazio, si usa un algoritmo+meccanismo di sostituzione
  - Una pagina già in memoria ("vittima") viene spostata su disco

Page Fault con sostituzione

Variante con il meccanismo di sostituzione:

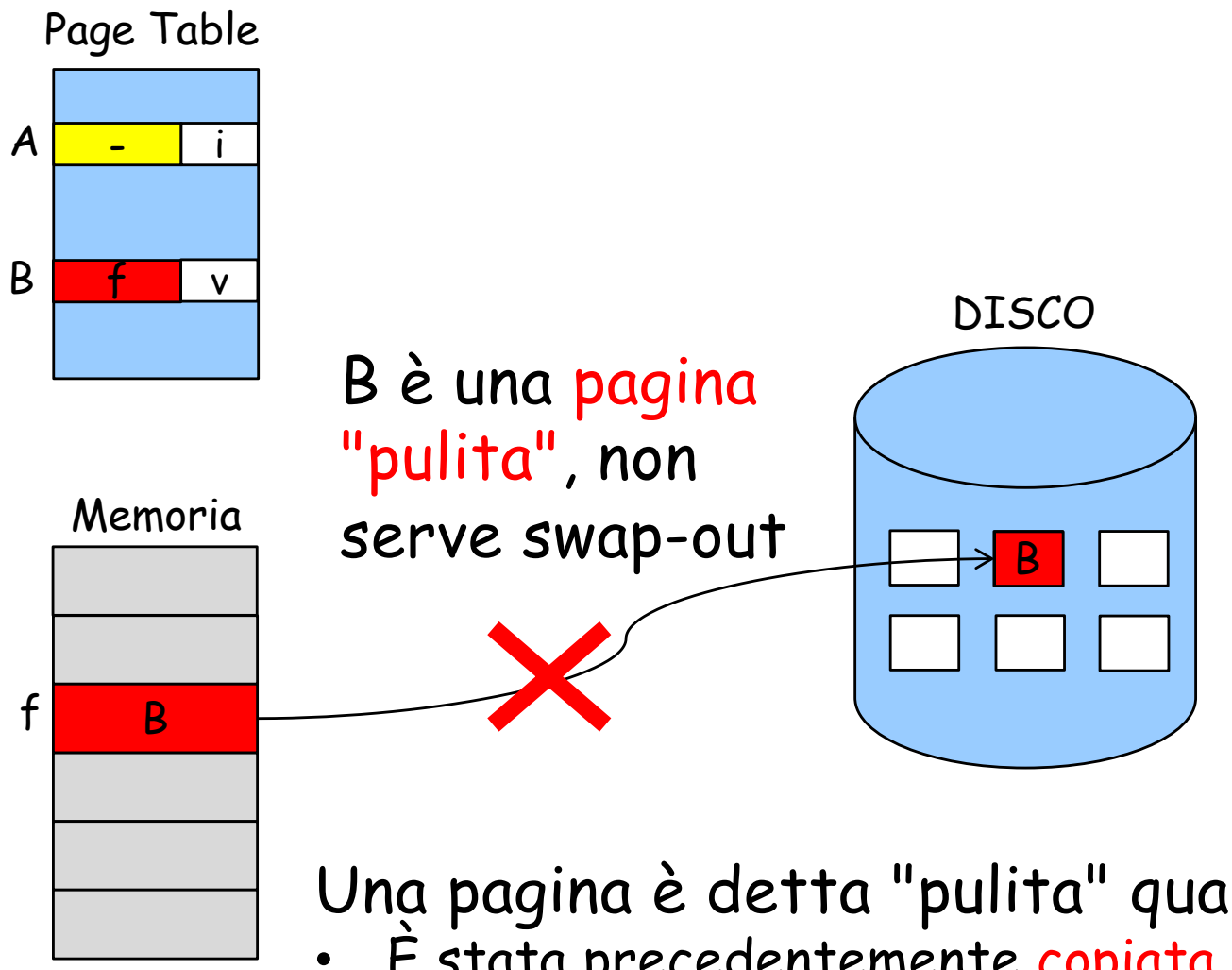
1. Il SO individua la posizione sul disco della pagina richiesta
2. Il SO cerca una pagina fisica di memoria libera:
   - Se esiste, la si usa
   - Altrimenti, occorre togliere dalla memoria principale un'altra pagina, detta "vittima" (scelta con un **algoritmo di sostituzione**)
   - La pagina vittima viene scritta sul disco (**swap-out**)
3. Il SO scrive la pagina richiesta nella pagina di memoria liberata
4. Il SO modifica le tabelle delle pagine (sia del richiedente sia del processo vittima)
5. Si riprende l'esecuzione del processo richiedente

Sostituzione di una pagina



Tutte le pagine fisiche sono occupate, si sceglie una "vittima" (es. B)

Pagine "pulite"



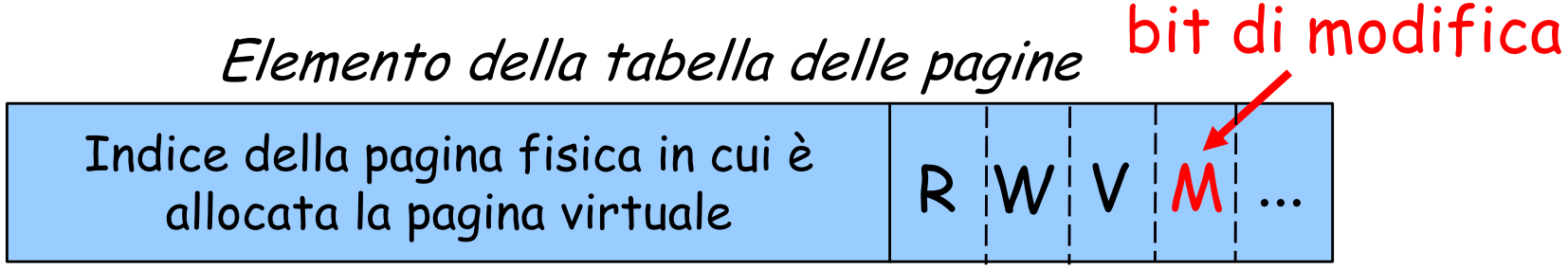
B è una pagina "pulita", non serve swap-out

Una pagina è detta "pulita" quando:

- È stata precedentemente copiata dal disco
- La copia in memoria non ha avuto modifiche

Bit di modifica

- Le pagine pulite sono identificate da un **bit di modifica M** (*dirty bit*)
  - Al caricamento in RAM, viene posto M=0 (la pagina non è stata finora modificata, ma soltanto letta)
  - Alla prima modifica, la MMU pone M=1



Algoritmi di sostituzione delle pagine

- La scelta della pagina vittima è fatta da un algoritmo di sostituzione delle pagine
- Si desidera minimizzare la frequenza dei page fault
- Confronteremo i page fault ottenuti da vari algoritmi
  - Algoritmo "ottimo"
  - FIFO
  - LRU
  - Second-chance

Algoritmi di sostituzione delle pagine

- In tutti i nostri esempi, valuteremo gli algoritmi simulando un processo con 5 pagine di memoria virtuale, che vi accede in questa successione:

1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

(in ordine temporale, da sinistra a destra)

Algoritmo FIFO

Successione di riferimenti: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

3 pagine fisiche

MEM. FISICA								
1		1	4	4	4	5	5	5
2		2	2	1	1	1	3	3
3		3	3	3	2	2	2	4
SWAP	1,2,3,4,5	4,5	1,5	2,5	3,5	3,4	1,4	1,2

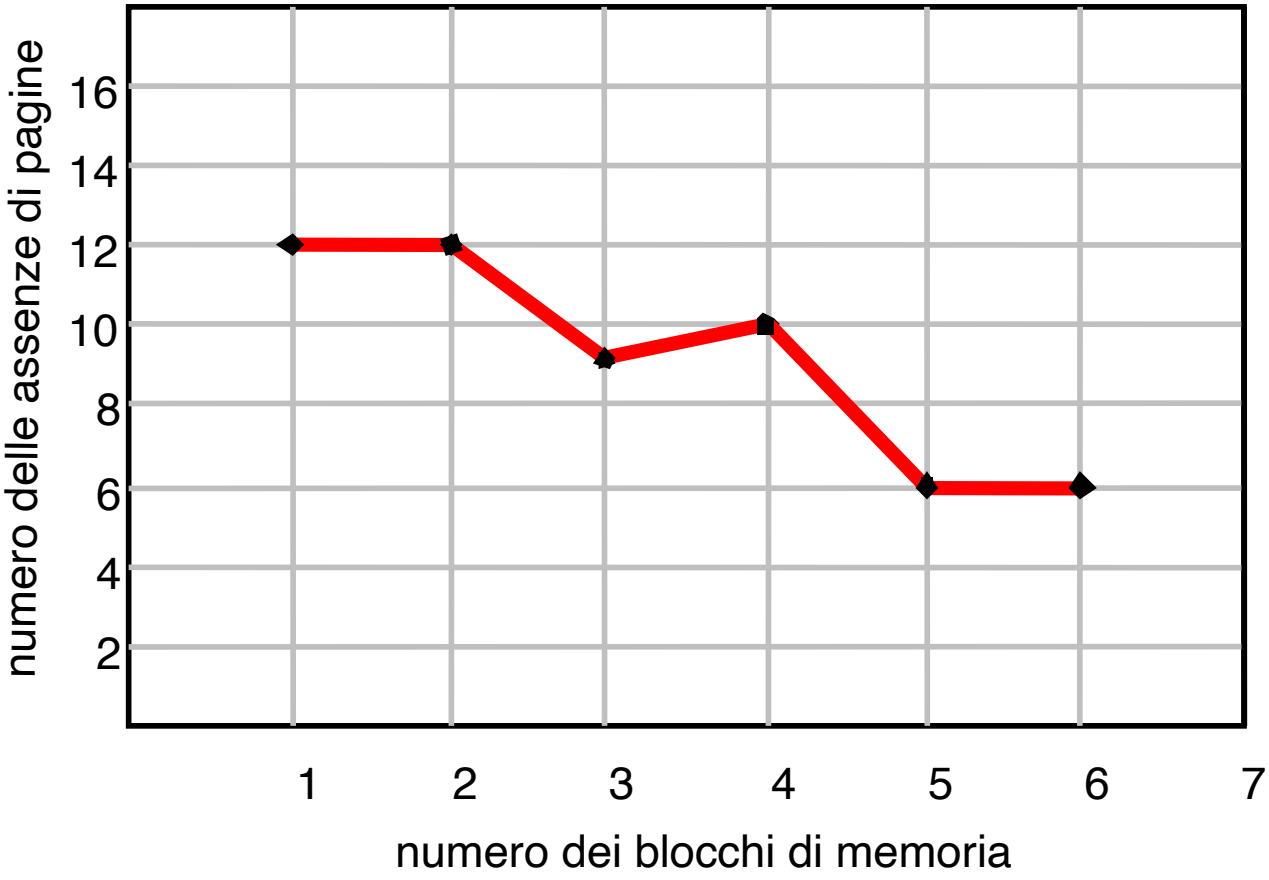
9 assenze di pagine

4 pagine fisiche

MEM. FISICA								
1		1	5	5	5	5	4	4
2		2	2	1	1	1	1	5
3		3	3	3	2	2	2	2
4		4	4	4	4	3	3	3
SWAP	1,2,3,4,5	5	1	2	3	4	5	1

10 assenze di pagine

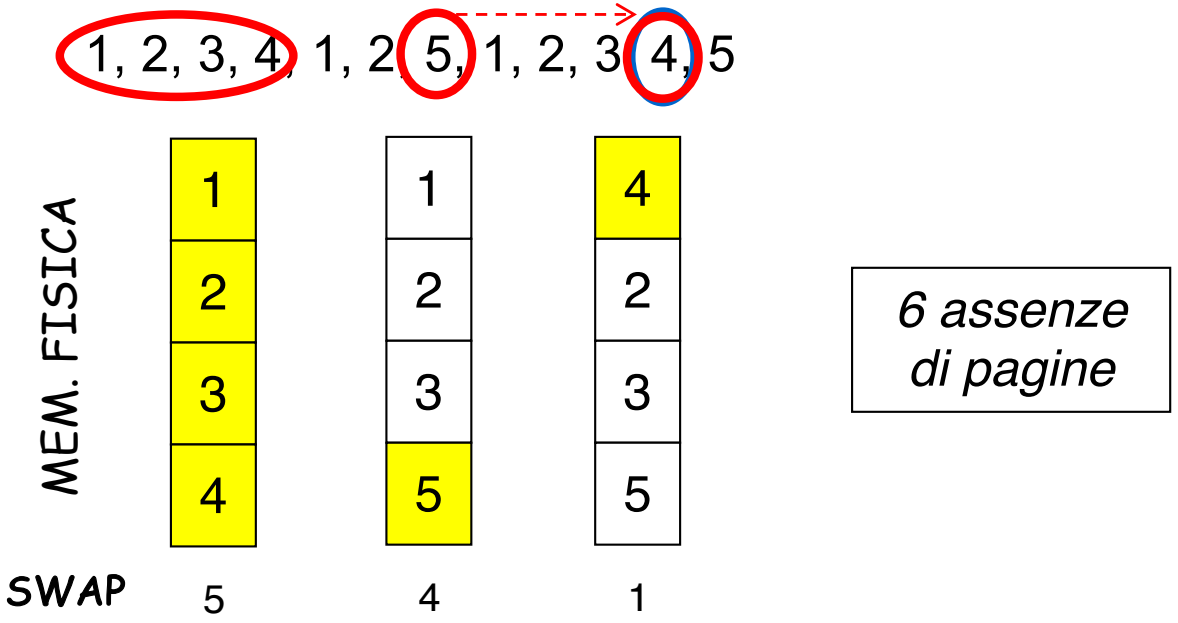
Curva delle assenze di pagine per sostituzione FIFO su una successione di riferimenti



Anomalia di Belady: nell'algoritmo di sostituzione FIFO, avere più pagine fisiche di memoria può causare (paradossalmente) più assenze di pagina!

Sostituzione delle pagine ottimale

- Si sostituisce la pagina che non si userà per il periodo di tempo più lungo
- Esempio con 4 pagine fisiche



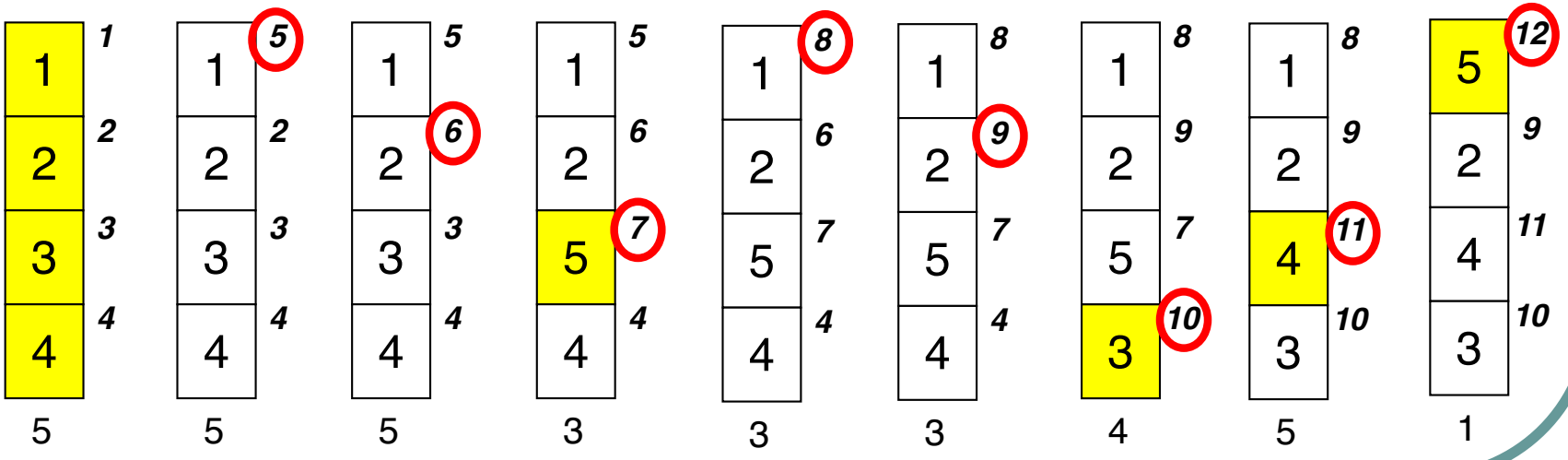
6 assenze di pagine

Purtroppo, è impossibile conoscere in anticipo quali saranno i futuri riferimenti! È un algoritmo usato principalmente per studi comparativi, per valutare le prestazioni di altri algoritmi

Algoritmo LRU (Least Recently Used)

- Ad ogni pagina si abbina **un campo numerico, detto "momento d'uso"**
- Ad ogni accesso ad una pagina, la CPU aggiorna il campo, usando il valore di un contatore crescente
- L'algoritmo sostituisce la pagina acceduta meno di recente (quella con il momento d'uso più basso)
- Si complica la ricerca, e la struttura della tabella delle pagine e della MMU

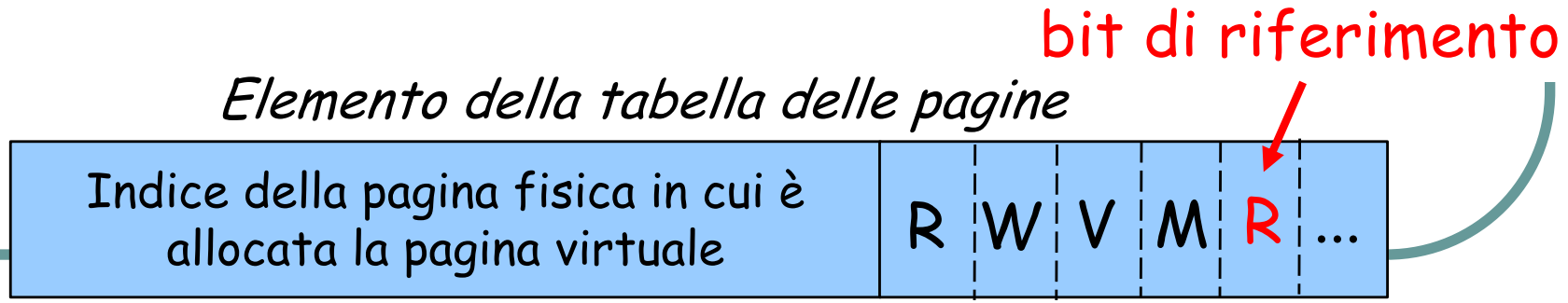
Successione dei riferimenti: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5



8 assenze di pagine

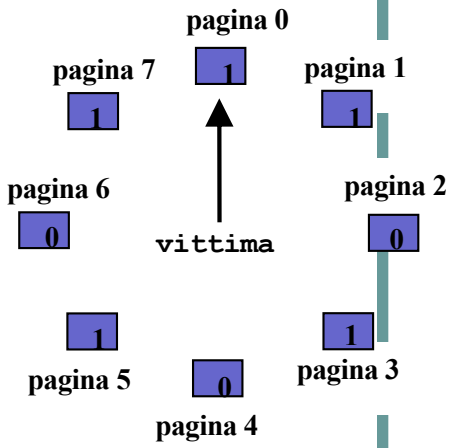
Bit di riferimento

- La maggior parte dei sistemi utilizza una approssimazione dello LRU
- Si utilizza un singolo bit di riferimento, invece che un contatore
- Il SO lo inizializza ponendo R=0
- Ad ogni accesso, la MMU pone R=1

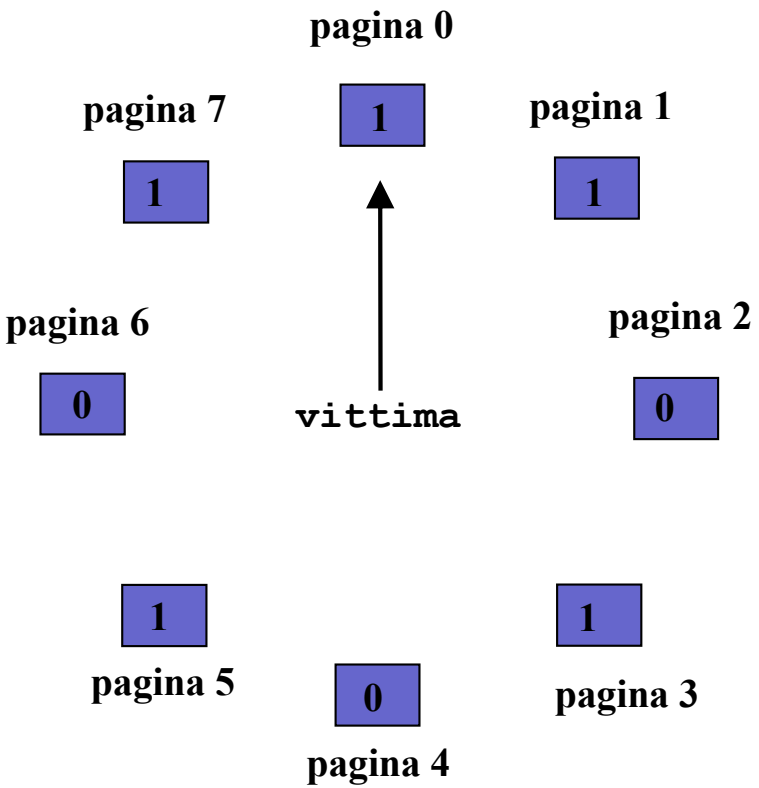


Algoritmo della second chance (approssimazione di LRU)

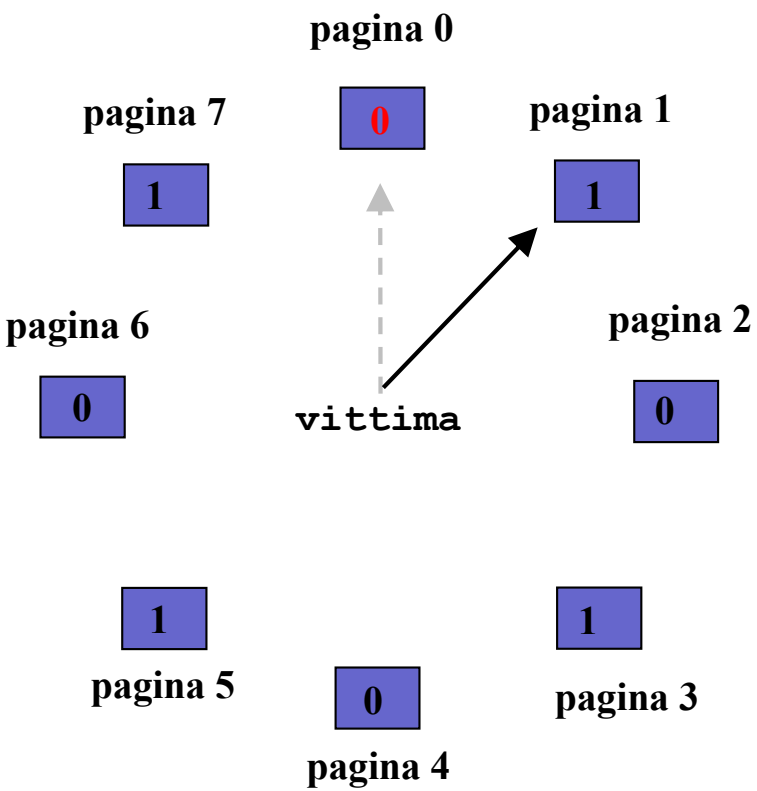
- Algoritmo della **seconda chance**
  - Si itera con un puntatore su una lista delle pagine
  - Se la **pagina candidata** ha il bit di riferimento = 1, allora:
    - la pagina non viene sostituita
    - si pone il suo bit di riferimento a 0
    - si analizza la pagina successiva, ripetendo lo stesso controllo
  - Se la pagina candidata ha il bit di riferimento = 0, allora viene sostituita
  - Ad ogni esecuzione, l'algoritmo riprende da dove si era fermato la precedente esecuzione
  - Alla fine della lista, si riprende dall'inizio



Algoritmo di rimpiazzamento second-chance



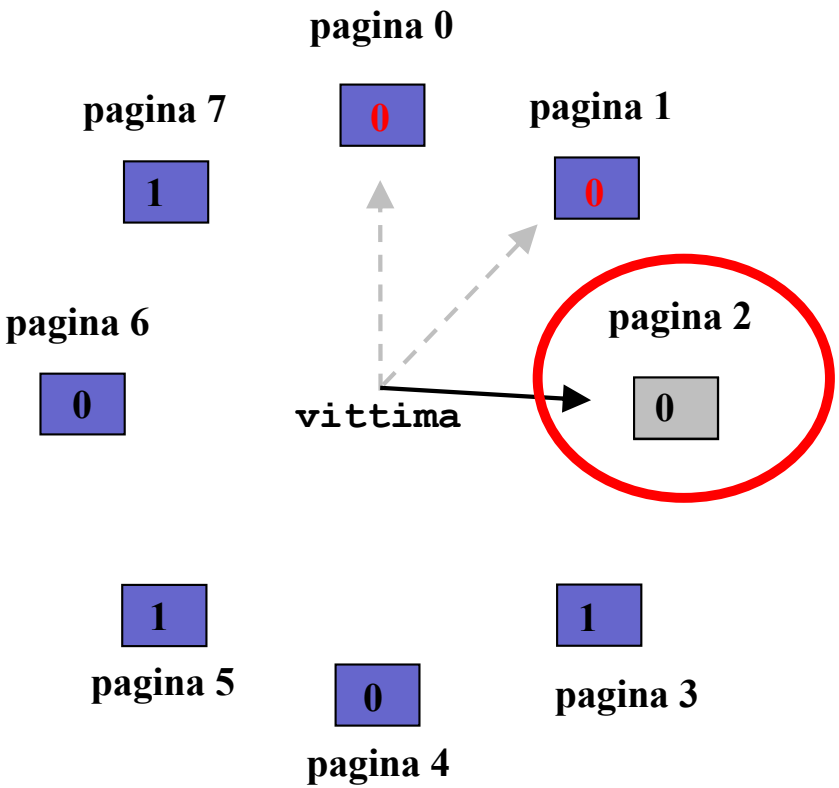
Algoritmo di rimpiazzamento second-chance



La pagina 0 è stata recentemente usata (aveva **R=1**, si pone a 0).

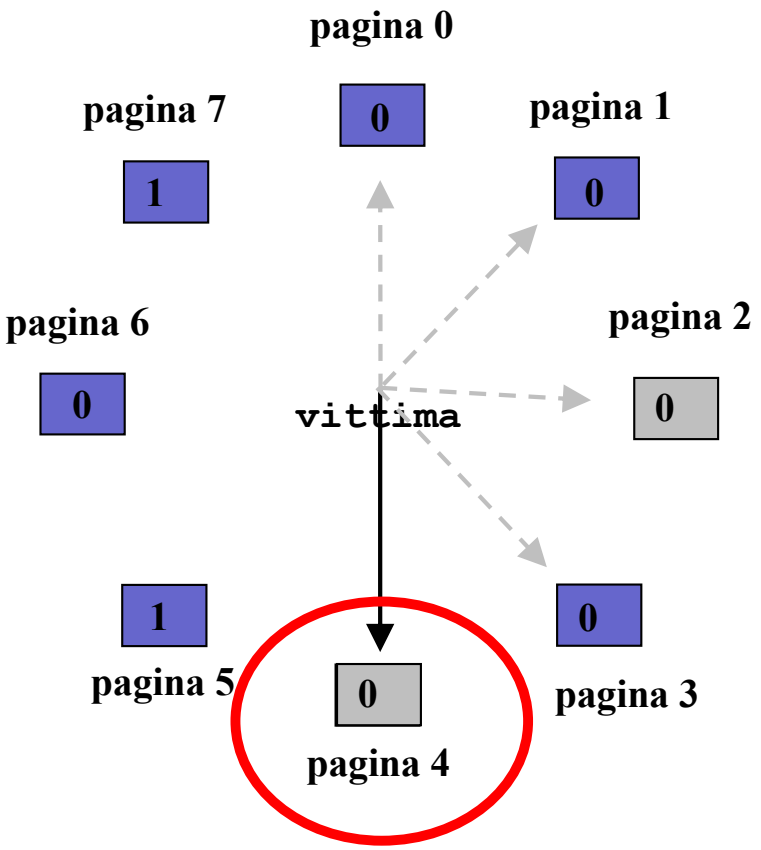
L'algoritmo passa ad ispezionare la successiva.

Algoritmo di rimpiazzamento second-chance



Pagina da rimpiazzare, ha R=0

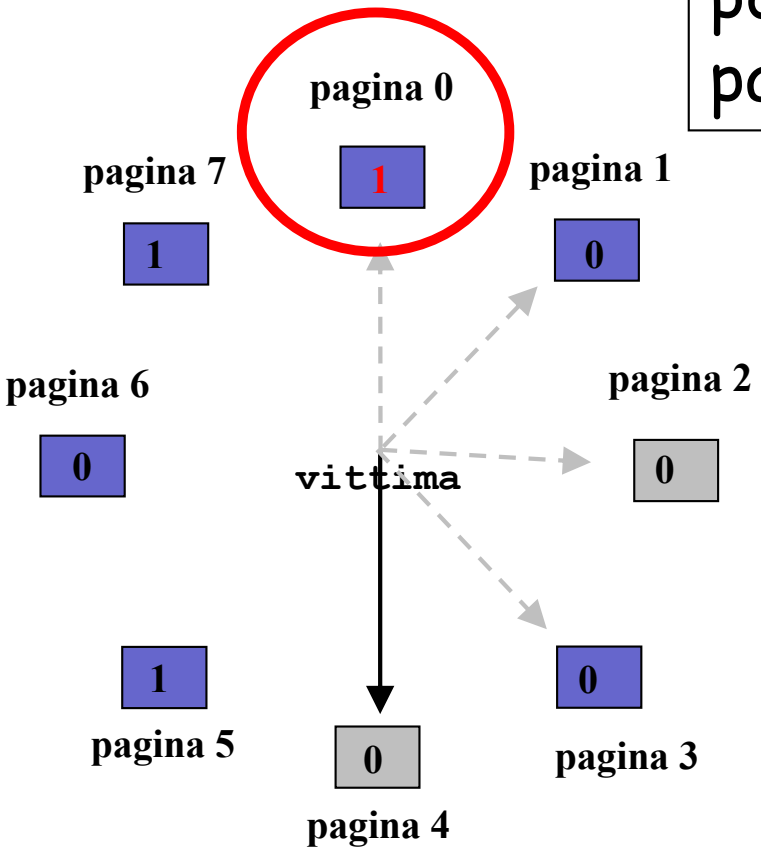
Algoritmo di rimpiazzamento second-chance



Prossima pagina da rimpiazzare (nel caso di un secondo page fault e nessun accesso)

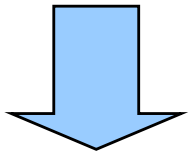
Algoritmo di rimpiazzamento second-chance

Il bit di riferimento è posto di nuovo a 1 se la pagina viene acceduta



Attività di paginazione degenere

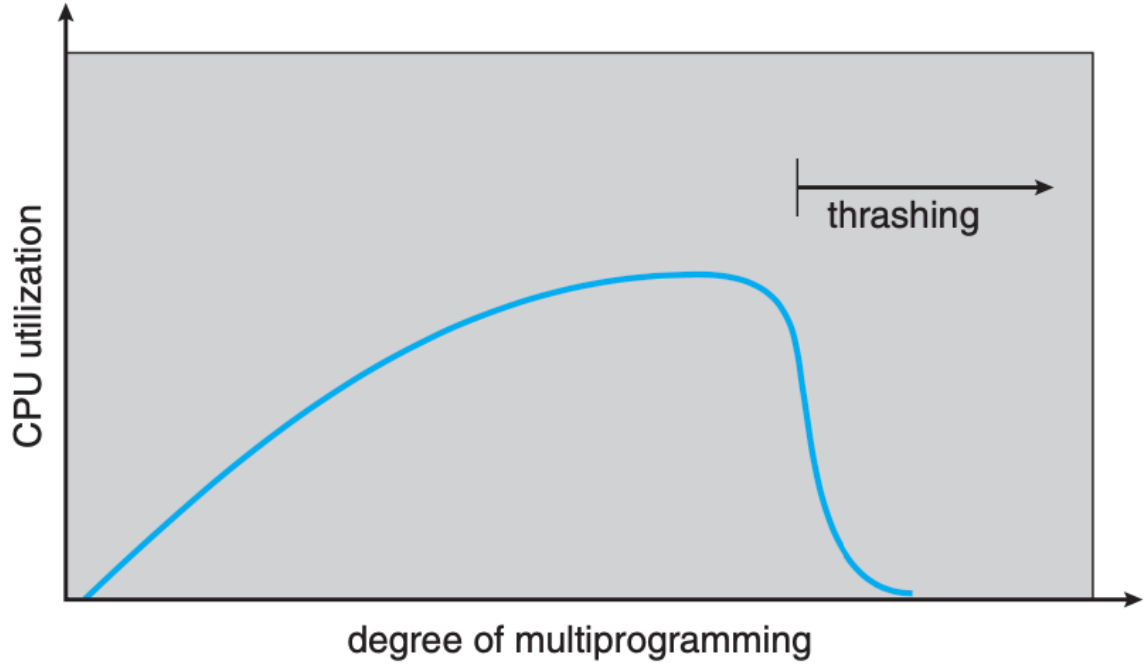
- Se un processo non riceve un numero "sufficiente" di pagine fisiche…
- …, l'algoritmo inevitabilmente seleziona una pagina che sarà acceduta prossimamente
  - Si verificano molte assenze di pagine
  - Basso utilizzo della CPU (il sistema passa più tempo a gestire page fault che ad eseguire i programmi!)
  - Può indurre l'utente o lo scheduler di lungo termine ad ammettere nuovi processi, innescando un circolo vizioso



Thrashing ≡ degenerazione dell'attività di paginazione

Attività di paginazione degenere (thrashing)

L'attività di paginazione può degenerare in una situazione patologica, che fa precipitare la produttività del sistema

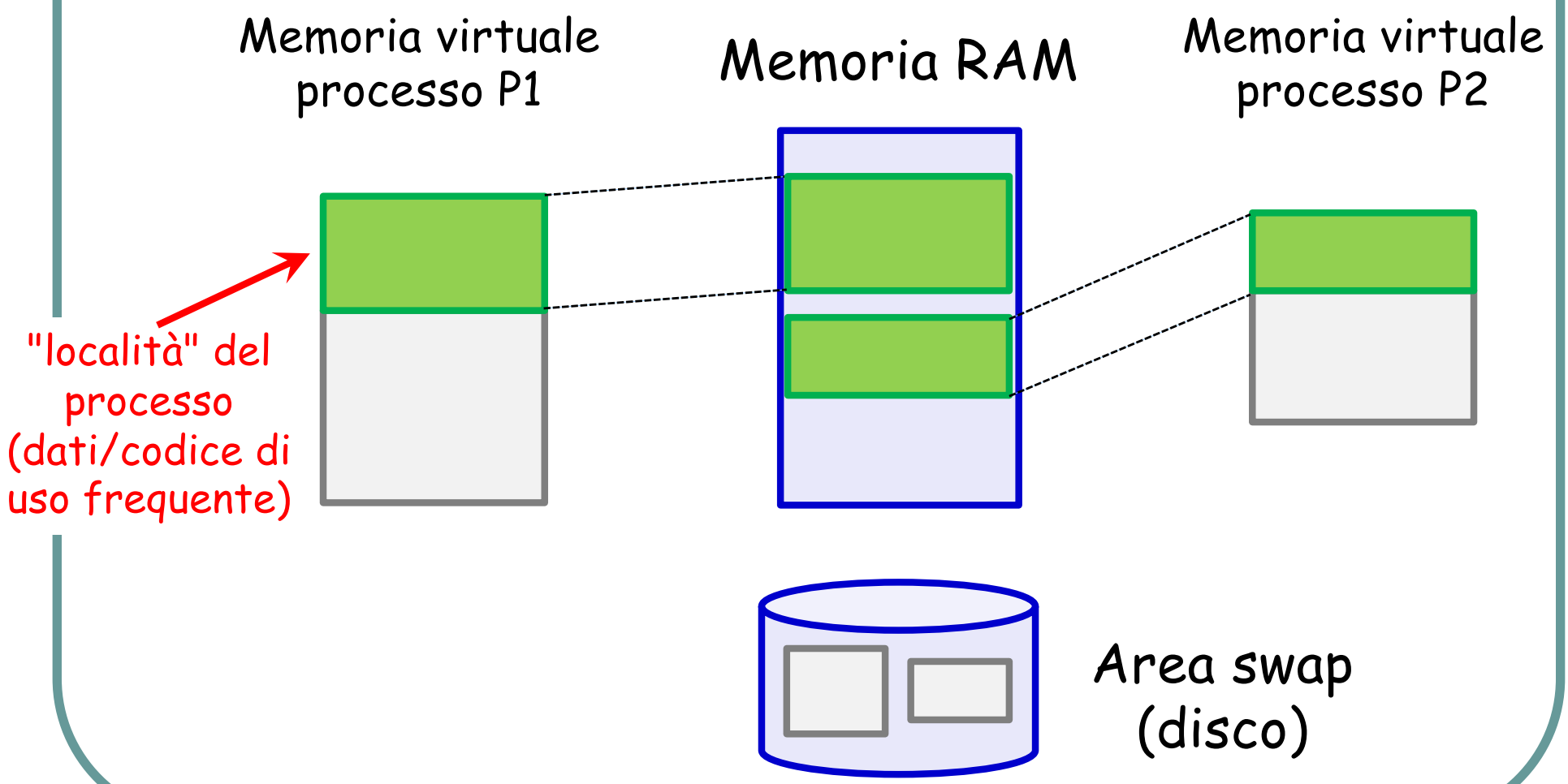


(Numero di processi nel sistema)

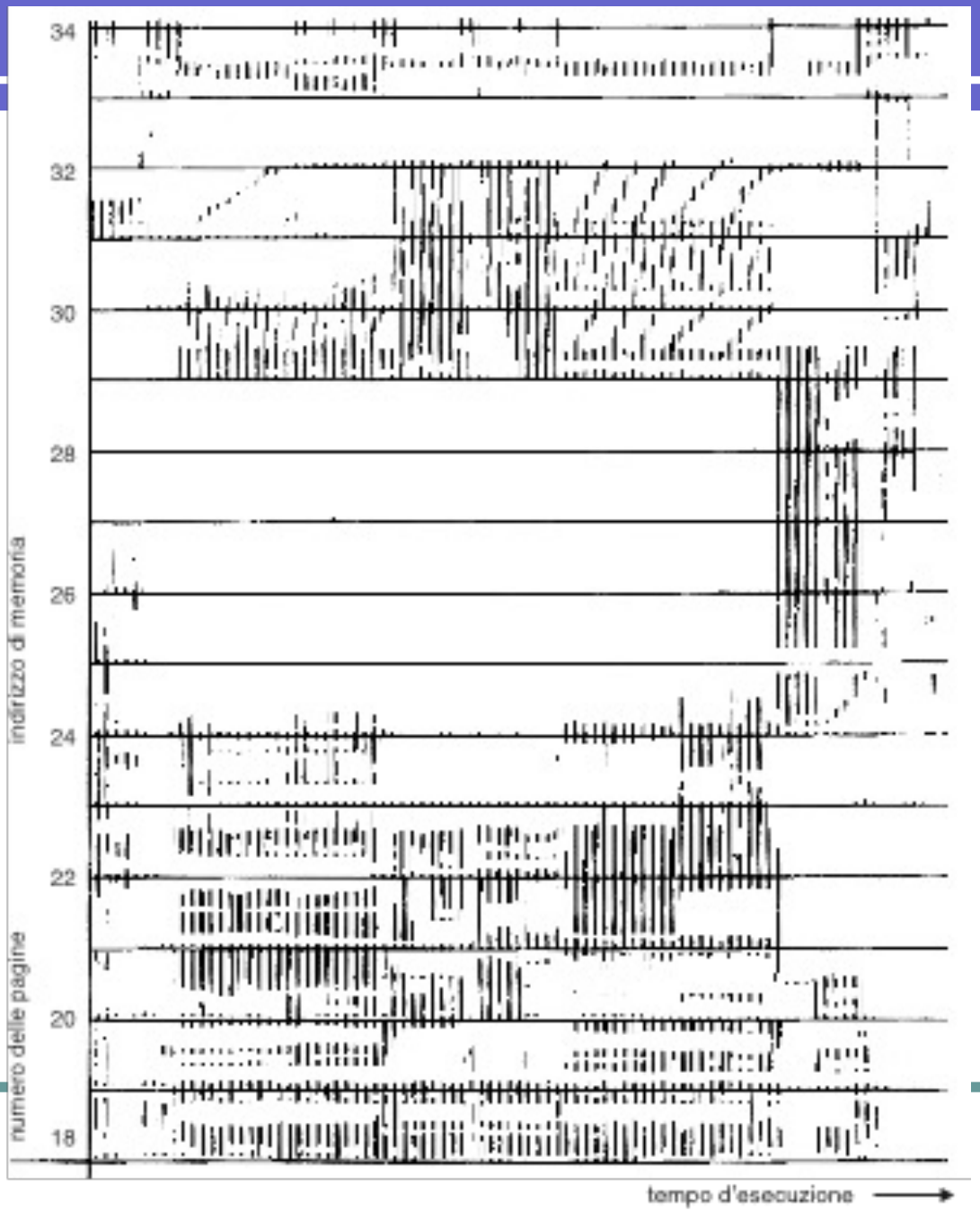
Origine del problema

- La paginazione a domanda funziona bene in virtù del **principio di località**
  - I processi tendono ad accedere a un sottoinsieme di pagine della loro memoria ("**working set**")
  - Per evitare il thrashing, è necessario che almeno il working set dei processi risieda in memoria

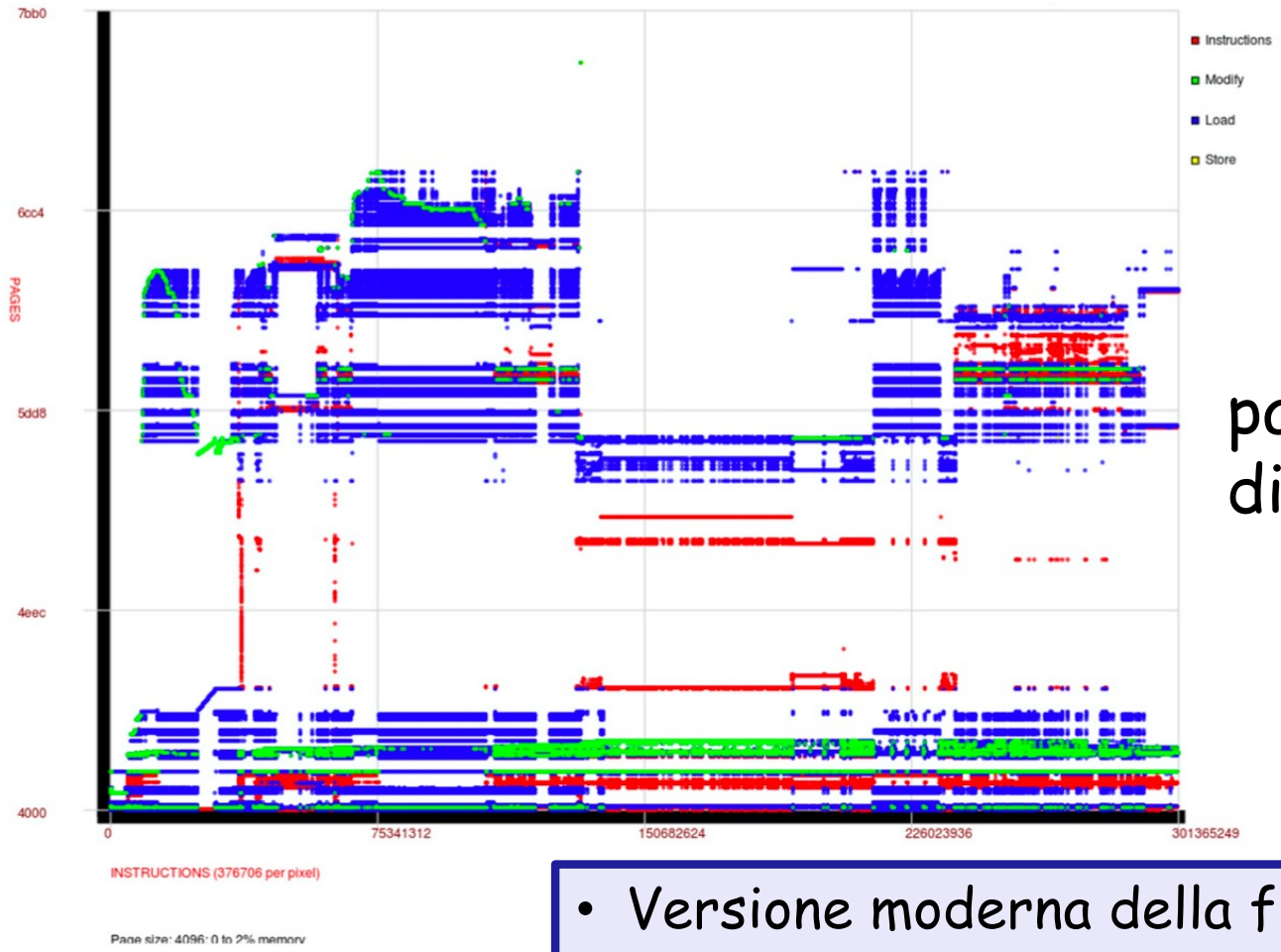
Località di riferimenti alla memoria



Località di riferimenti alla memoria



Località di riferimenti alla memoria



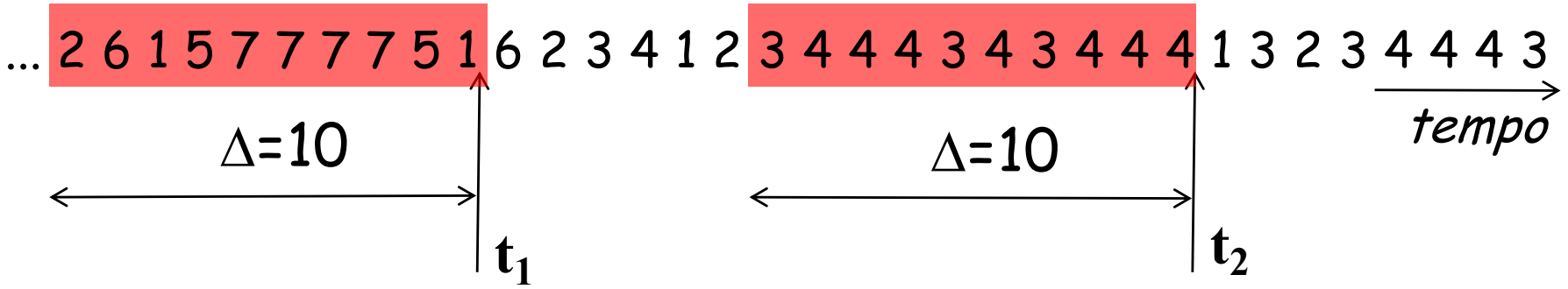
page reference map di Firefox in Linux

- Versione moderna della figura precedente
- Il principio di località è una "legge universale" dell'informatica!

P.J. Denning, "Working Set Analytics." *ACM Computing Surveys (CSUR)* 53.6 (2021): 1-36. DOI: https://doi.org/10.1145/3399709

Working set

- WSS_i (t): **working set size** del processo *i*, all'istante *t*
  - **Numero totale delle pagine** cui il processo fa riferimento in un periodo Δ
  - Varia nel tempo, in base alla attività del processo



$WSS_i(t_1) = |\{1,2,5,6,7\}| = 5$ $WSS_i(t_2) = |\{3,4\}| = 2$

$\Delta \equiv$ finestra empirica di osservazione (es., 10.000 istr.)

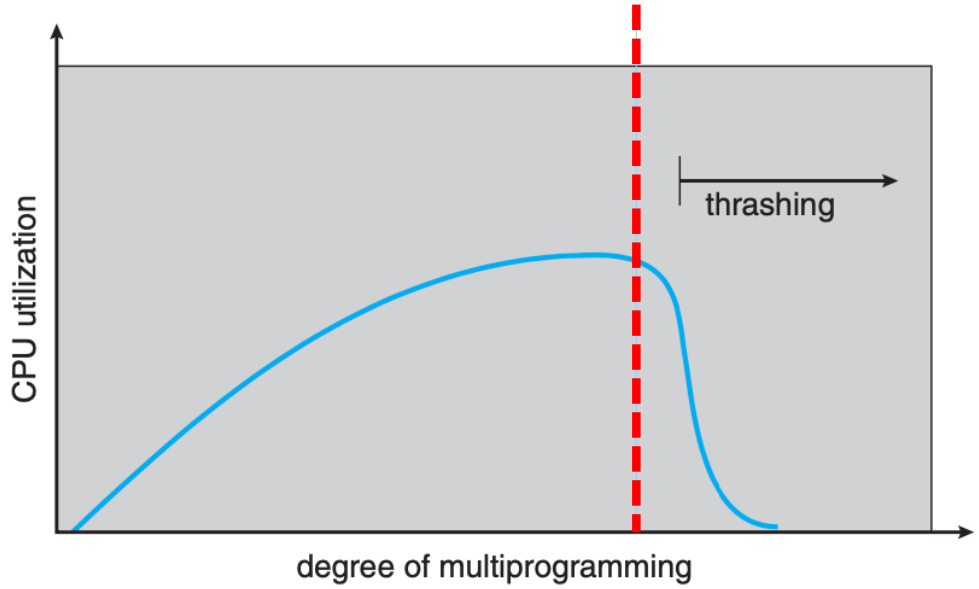
Working set

- $D = \sum WSS_i \equiv$ richiesta totale di memoria, da parte di tutti i processi nel sistema
- $M \equiv$ quantità totale disponibile di memoria fisica
- L'attività di paginazione degenera (*thrashing*) se

$$D > M$$

Soluzione al thrashing

In caso di thrashing, occorre ridurre il grado di multiprogrammazione
(es. sospendere o terminare dei processi)



Quiz

1. Quale dei seguenti bit della tabella delle pagine viene utilizzato nell'algoritmo della "second chance"?

   - ◯ Bit di lettura
   - ◯ Bit di scrittura
   - ◯ Bit di validità
   - ◯ Bit di modifica
   - ◯ Bit di riferimento

2. La paginazione degenere (thrashing) si verifica quando

   - ◯ La somma dello spazio virtuale dei processi supera la quantità di memoria RAM
   - ◯ La somma del working set space dei processi supera la quantità di memoria RAM
   - ◯ La somma dello spazio virtuale dei processi supera la quantità di spazio su swap
   - ◯ La somma del working set space dei processi supera la quantità di spazio su swap

https://forms.office.com/r/Uq2tj1UpuA