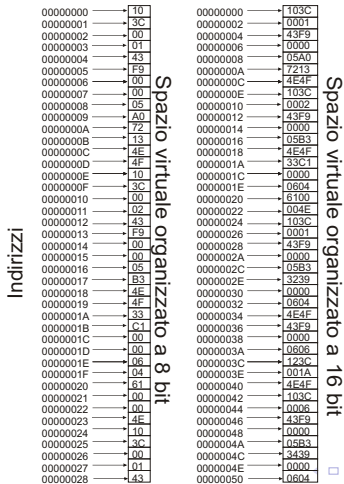


Memoria virtuale

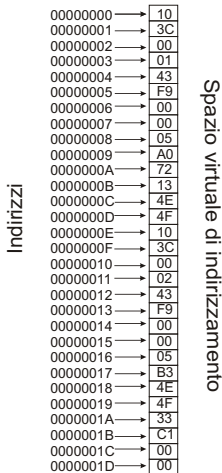
Spazio di indirizzamento virtuale

Riprendiamo lo spazio di indirizzamento virtuale del programma di inversione stringa



Spazio di indirizzamento virtuale

La memoria é organizzata a byte



Spazio di indirizzamento virtuale

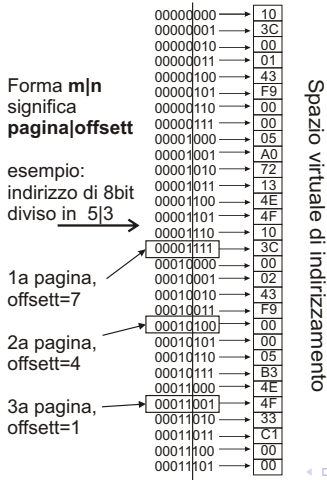
Rappresentiamo gli indirizzi in binario

Indirizzi	00000000	→	10
	00000001	→	3C
	00000010	→	00
	00000011	→	01
	00000100	→	43
	00000101	→	F9
	00000110	→	00
	00000111	→	00
	00001000	→	05
	00001001	→	A0
	00001010	→	72
	00001011	→	13
	00001100	→	4E
	00001101	→	4F
	00001110	→	10
	00001111	→	3C
	00010000	→	00
	00010001	→	02
	00010010	→	43
	00010011	→	F9
	00010100	→	00
	00010101	→	00
	00010110	→	05
	00010111	→	B3
	00011000	→	4E
	00011001	→	4F
	00011010	→	33
	00011011	→	C1
	00011100	→	00
	00011101	→	00

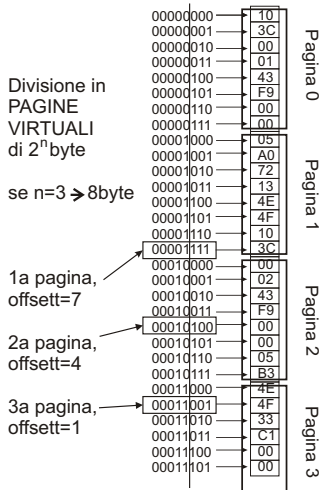
Spazio virtuale di indirizzamento

Divisione dello spazio di indirizzamento virtuale in pagine

Dividiamo l'indirizzo in due parti: numero di pagina virtuale e offset

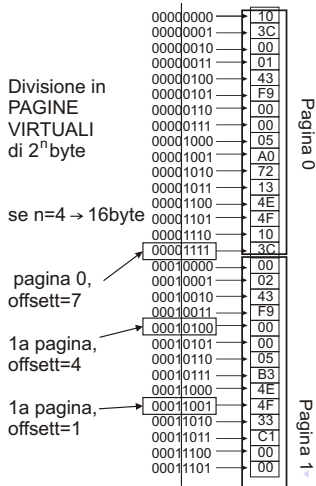


Spazio di indirizzamento virtuale diviso in pagine di 8 byte



Divisione dello spazio di indirizzamento virtuale in pagine

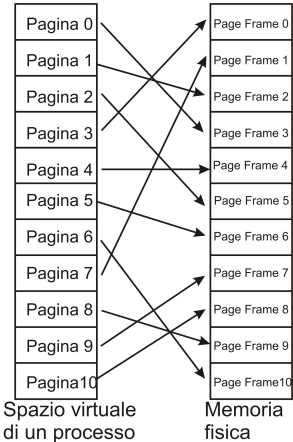
Spazio di indirizzamento virtuale diviso in pagine di 16 byte



Pagine virtuali nelle 'page frame' della memoria fisica

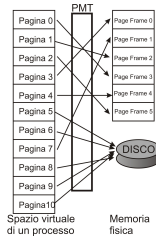
Anche la Memoria fisica é divisa in pagine: PAGE FRAMES

PAGINAZIONE: carica la pagina virtuale in una PageFrame disponibile



Memoria virtuale

Memoria virtuale: la memoria fisica é piú piccola dello spazio di indirizzamento



La funzione (pagina virtuale \Rightarrow page frame fisico) é realizzata con la tabella delle pagine Page Map Table PMT)

PAGE MAP TABLE PMT

Bit di riferimento
 Bit di modifica
 Bit di protezione pagina
 Bit di validità (presente/assente)
 Pagina fisica dov'è allocata la pagina virtuale

	R	M	R	W	V	nr. Page Frame
Pagina virtuale 0						
Pagina virtuale 1						
Pagina virtuale 2						
Pagina virtuale 3						
Pagina virtuale 4						

Riassumendo

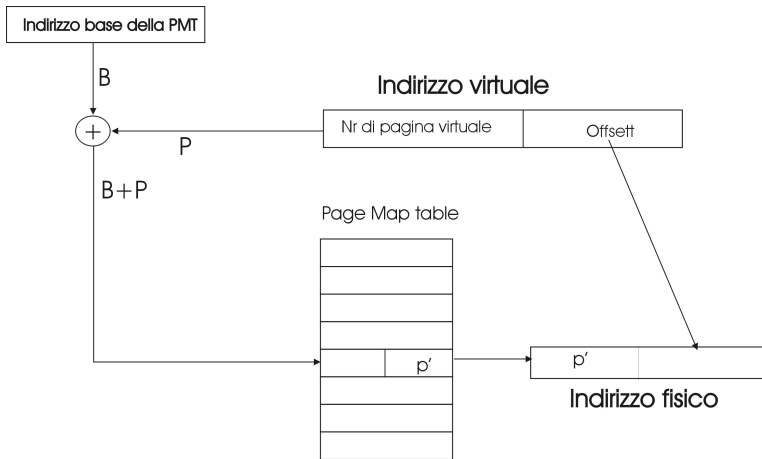
- L'indirizzo virtuale può essere diviso in due parti: x di m bit = nr. di pagina virtuale, w di n bit = offset nella pagina
- Se c è la dimensione di una pagina ($=2^n$), l'indirizzo virtuale è

$$\alpha = x \cdot c + w$$

- I processi possono essere divisi in pagine virtuali; anche la memoria fisica viene divisa in pagine della stessa dimensione
- Ogni pagina virtuale è memorizzata sul disco
- Per eseguire, lo spazio virtuale di un processo deve essere caricato in memoria centrale. Alcune pagine virtuali vengono caricate nei page frame della memoria fisica, le altre sono residenti sul disco.
- Memoria virtuale: la memoria fisica può essere più piccola dello spazio di indirizzamento
- Traduzione indirizzo virtuale \Rightarrow indirizzo fisico: mediante una tabella (PMT) che realizza la seguente relazione

$$\beta = f(x) \cdot c + w$$

Traduzione indirizzi (virtuale - fisico)



Problema: Page Fault

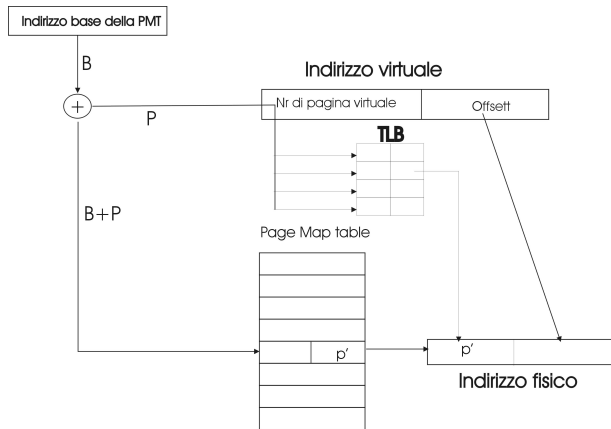
Page fault: evento legato alla non presenza di una pagina in memoria fisica.

Evento estremamente costoso → scopo del MMU é di limitare il page fault al minimo

Algoritmo di gestione del page fault:

- Se la pagina richiesta é assente dalla memoria → page fault → caricamento della pagina da disco
- Dove metterla in memoria? Se c'è spazio ok, altrimenti rimuovi una pagina
- Aggiorna la PMT
- Se la pagina rimossa é stata modificata → scrivila sul disco

Traduzione indirizzi con tabella TLB

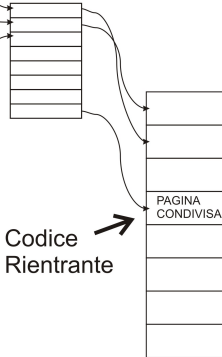


Ogni processo ha la sua PMT

Memoria virtuale
Processo A



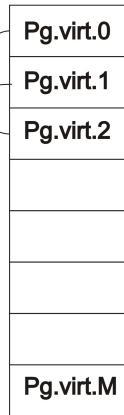
PMT
proc.A



Codice Rientrante



Memoria virtuale
Processo B



PMT
proc.B



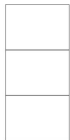
Page Frame memoria fisica

Politiche della memoria virtuale

- Allocazione: quante 'page frames'? (statica/dinamica)
- Caricamento: pre-caricamento/caricamento a richiesta (demand fetch)
- Rimpiazzamento: in seguito ad un page fault, se non ci sono piú 'page frames', quali page frame rimpiazzare? Politiche principali:
 - FIFO (anomalia nelle prestazioni)
 - Least Recently Used (LRU). Rimpiazza le pagine usate meno di recente
 - Most Recently Used (MRU). Rimpiazza le pagine usate piú di recente
 - Least Frequently Used (LFU). Rimpiazza le pagine usate meno frequentemente
 - Ottimo (Belady). Rimpiazza le pagine che saranno richieste piú in lá nel tempo
 - Approssimati (Not Used Recently, Second Chance). Usano i bit R,M
 - Working Set. Allocazione dinamica. Individua un insieme di lavoro.

Esempio: rimpiazzamento LRU,

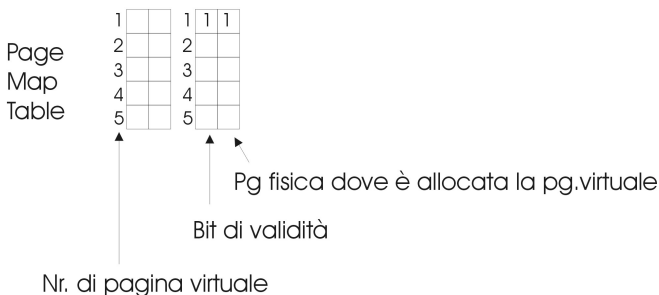
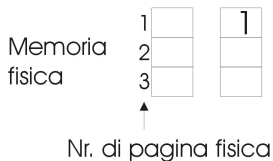
Memoria
fisica



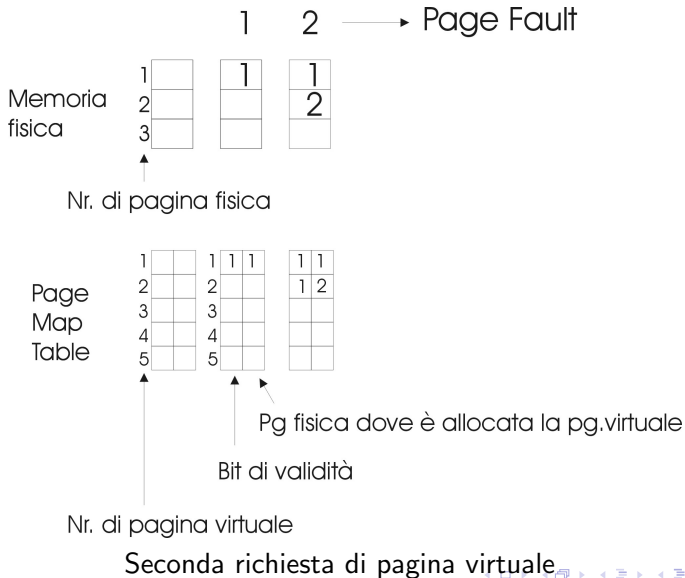
Page
Map
Table

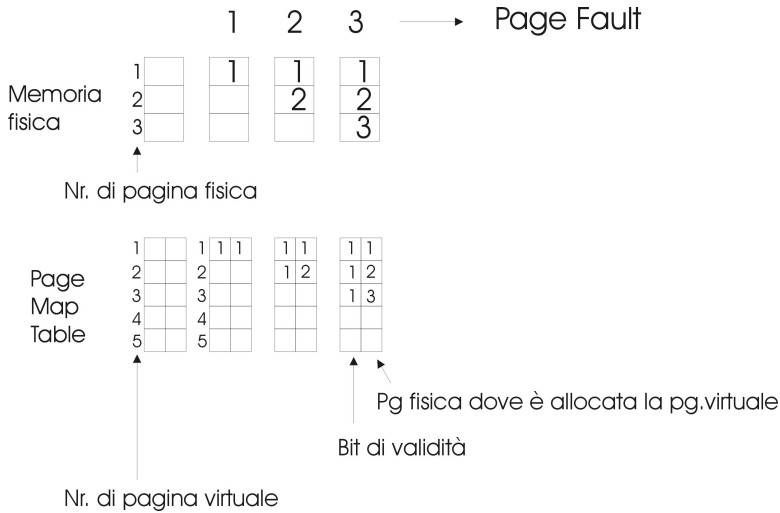


1 → Page Fault



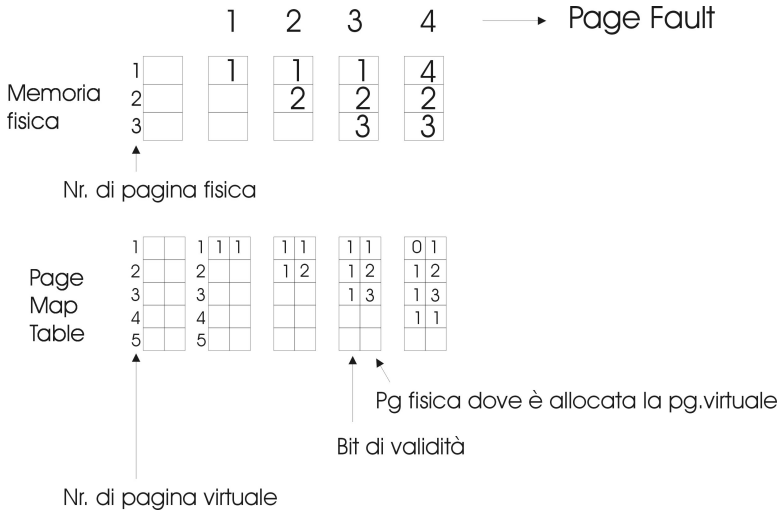
Prima richiesta di pagina virtuale



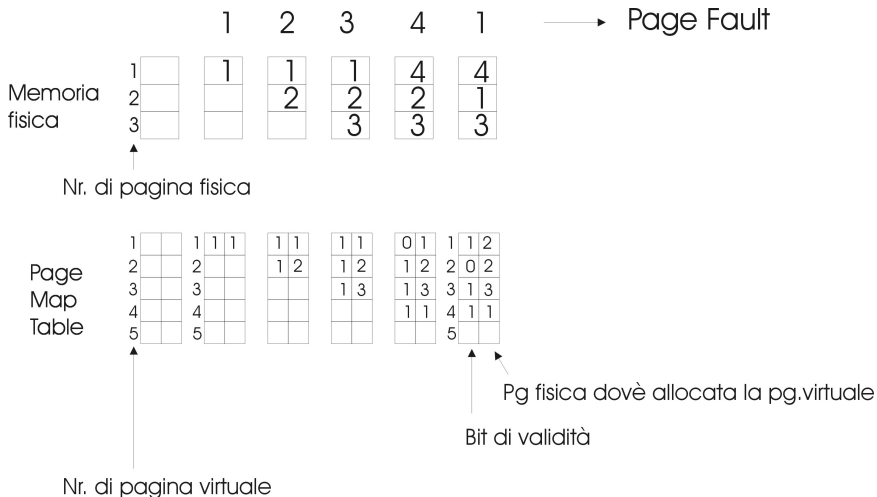


Terza richiesta di pagina virtuale

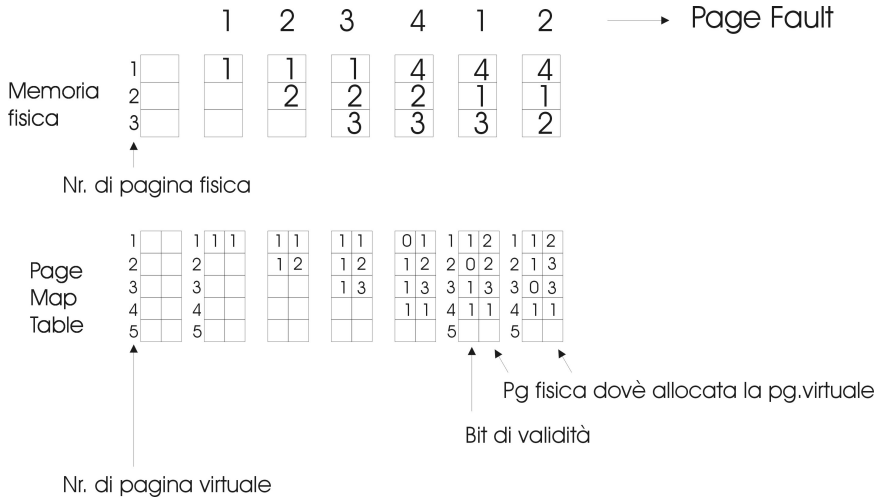




Quarta richiesta di pagina virtuale

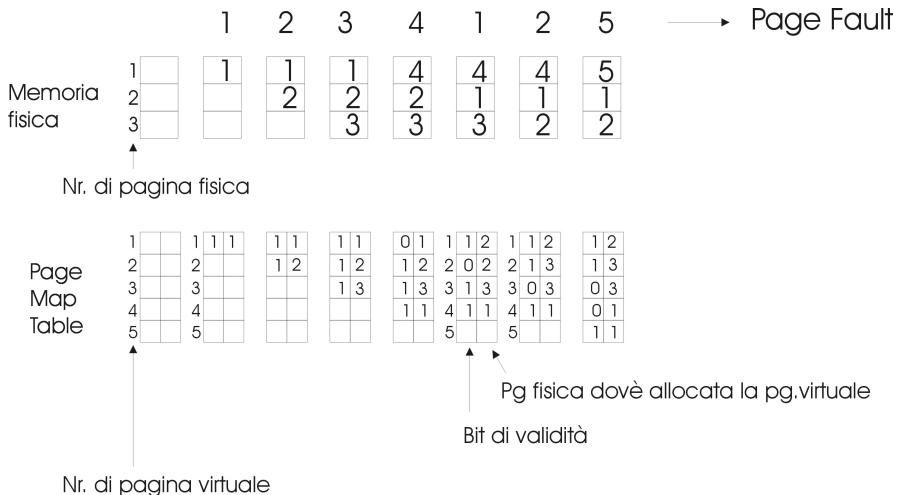


Quinta richiesta di pagina virtuale



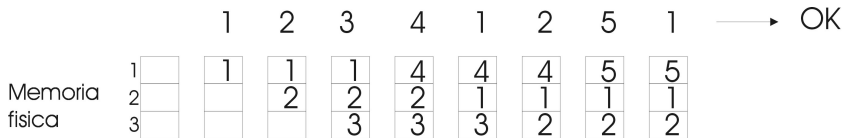
Sesta richiesta di pagina virtuale



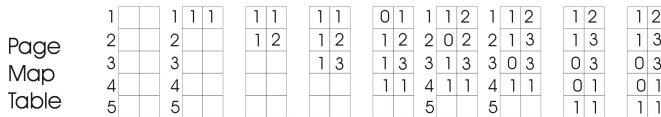


Settima richiesta di pagina virtuale





Nr. di pagina fisica



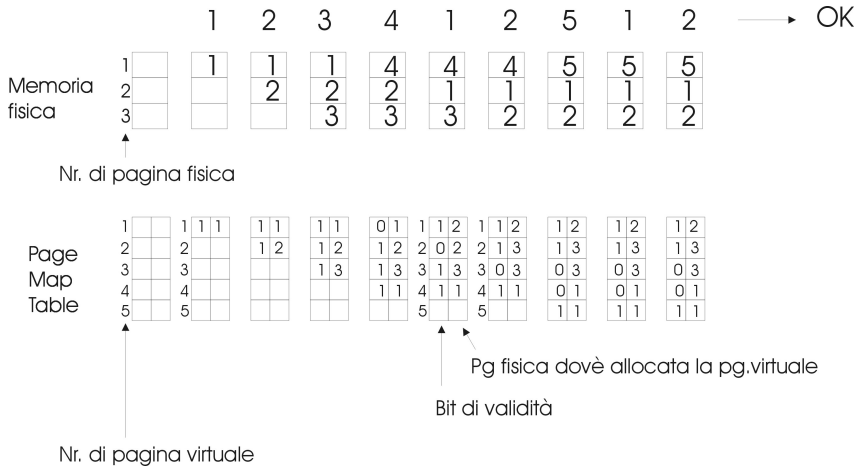
Pg fisica dovè allocata la pg.virtuale

Bit di validità

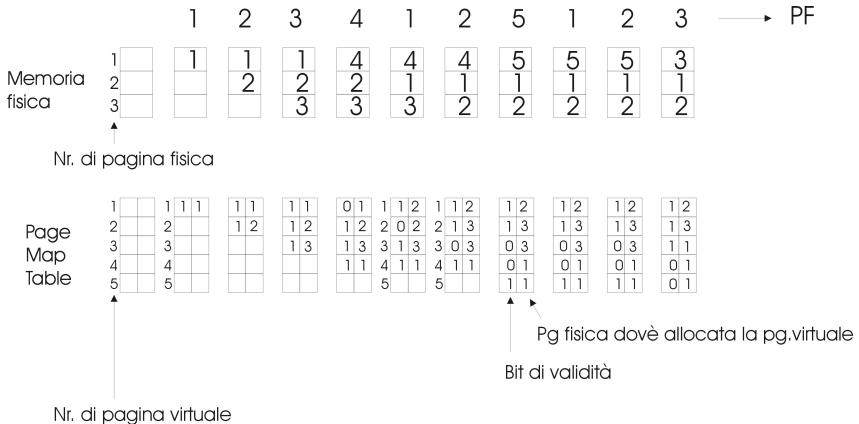
Nr. di pagina virtuale

Ottava richiesta di pagina virtuale

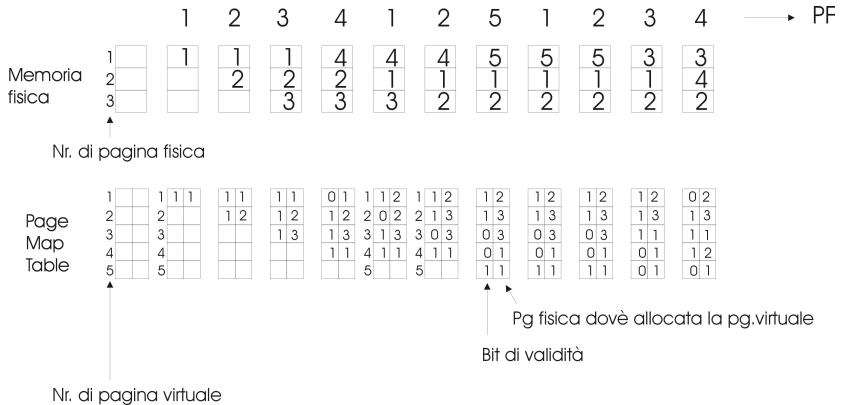




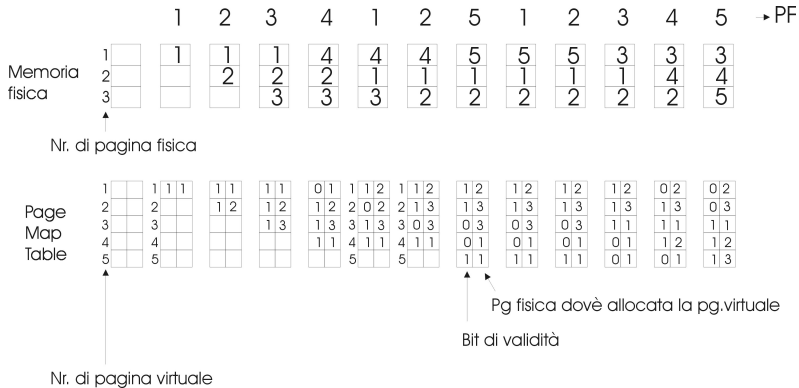
Nona richiesta di pagina virtuale



Decima richiesta di pagina virtuale



Undicesima richiesta di pagina virtuale



Dodicesima richiesta di pagina virtuale
 Page Fault=10/12=83%

Esempio con FIFO

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0
	0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1
		1	1	1	1	0	0	0	3	3	3	3	3	2	2	2
x	x	x	x		x	x	x	x	x	x			x	x		

$$PF=12/17=0.7$$

Esempio con algoritmo ottimo

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1
7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	4	4	3	3	3	3	1	1	1	1
x	x	x	x		x		x		x							

$$PF=7/17=0.41$$

Working Set

Prima figura: finestra di 3
 Seconda figura: finestra di 4

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3		0	0
		1	1	3	3	3	2	2	2	2		2	2	2	2	2
x	x	x	x		x		x	x	x	x		x	x	x	x	x

$PF=14/17=0.82$

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1
7	7	7	7		3	3	3	3	3	3	3	3	3	3		
	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0
		1	1	1				2	2	2	2	2	2	2	2	2
			2	2	2	2	4	4	4	4			1	1	1	1

$PF=8/17=0.47$

- Frammentazione interna: in media ϵ dimensione_pagina/2
- Dimensione delle pagine. Se le pagine sono piccole si ha
 - tabelle PMT possono essere grandi (molte pagine virtuali)
 - tuttavia la localit  viene approssimata al meglio
 - il tempo di trasferimento dal disco ϵ limitato

Se le pagine sono grandi si ha

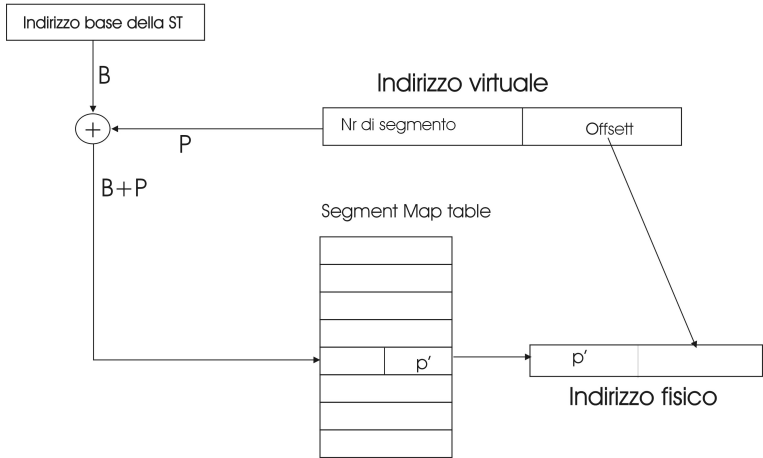
- alta frammentazione interna
- tempo di I/O alto
- troppa informazione in memoria

Generalmente: 512 byte, 2048 byte, 4096 byte

- La dimensione della PMT pu  essere notevole \rightarrow paginazione delle PMT
- Blocco delle pagine in memoria
- La struttura dei programmi ha un grande effetto sul page fault (puntatori, cicli etc)

Spazio virtuale di un processo: diviso in SEGMENTI LOGICI. I segmenti sono gestiti dalla Segment Map Table.

Segmentazione Pura



Segmentazione/paginazione