

Memoria Virtuale

Lezione 29
Sistemi Operativi

I Principi

- Abbiamo sinora assunto che durante l'esecuzione di un programma, lo stesso debba risiedere completamente in MC
- Intorno alla metà degli anni '70 viene però osservato che questo non è un requisito fondamentale per consentire l'esecuzione di un programma

Località

- Durante la loro esecuzione i programmi godono di due importanti proprietà:
 - Località temporale: se un elemento x viene referenziato all'istante t , la probabilità che x venga referenziato anche all'istante $t+t'$ cresce al tendere di $t' \rightarrow 0$
 - Località spaziale: se un elemento x di posizione s viene referenziato all'istante t , la probabilità che venga referenziato un elemento x' di posizione s' ,

$$|s - s'| \leq \Delta$$

all'istante $t+t'$ cresce al tendere di $t' \rightarrow 0$

Località

- I principi di località suggeriscono un importante accorgimento nella gestione della memoria:
“non è necessario caricare un programma interamente in memoria per poterlo eseguire, è sufficiente caricarlo località per località”

Località: i vantaggi

- I vantaggi che deriverebbero dall'adozione del suddetto schema sono:
 - Svincolo della dimensione di un programma dalla dimensione della memoria centrale
 - Aumento del livello di programmazione, a parità di memoria centrale disponibile
 - Maggiore velocità nelle operazioni di swapping

Tecniche implementative

- Esistono due approcci per realizzare questa funzionalità:
 - Overlay
 - Demand Paging (Segmentation)

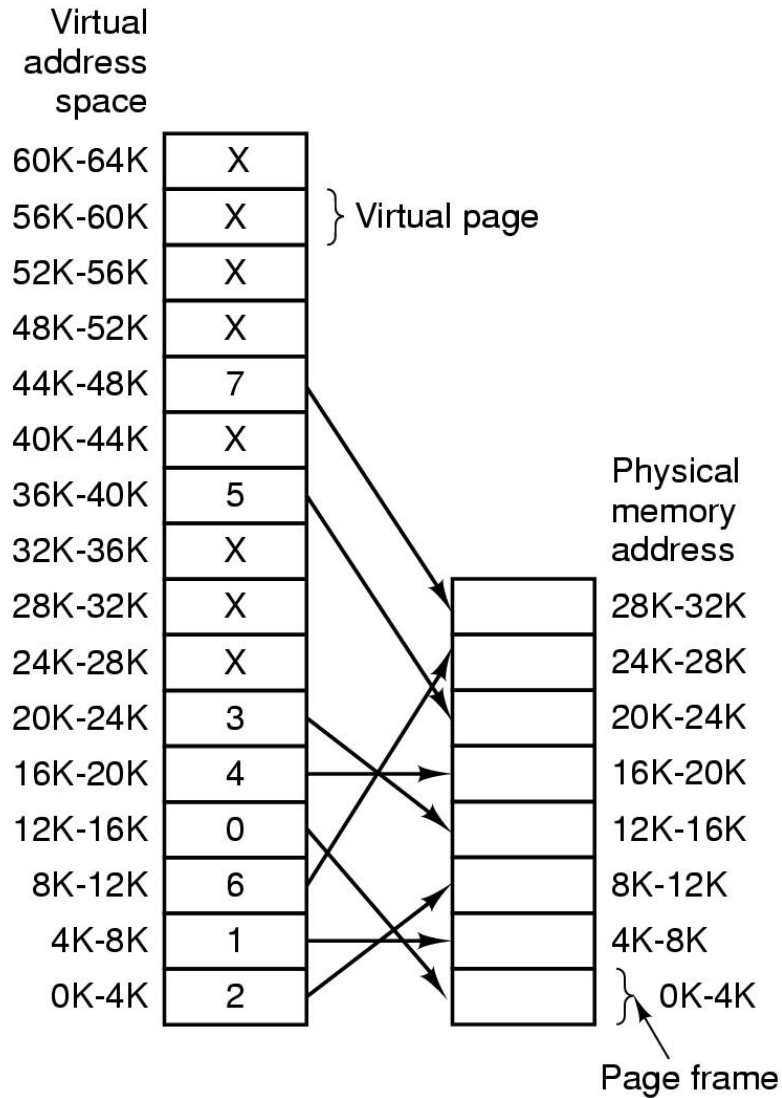
Overlay

- Meccanismo molto primitivo per realizzare la memoria virtuale
- Il programmatore suddivide il programma in porzioni la cui esecuzione non si sovrappone nel tempo
- Ogni porzione è chiamata overlay
- Gli overlay vengono caricati in istanti successivi ed eseguiti

Demand Paging

- Sistema completamente automatico per realizzare memoria virtuale
- Uno schema elementare può essere il seguente (lazy swapper)
 - Carico la prima pagina di programma da eseguire
 - Quando una nuova pagina viene referenziata la carico in memoria

Demand Paging



Demand Paging

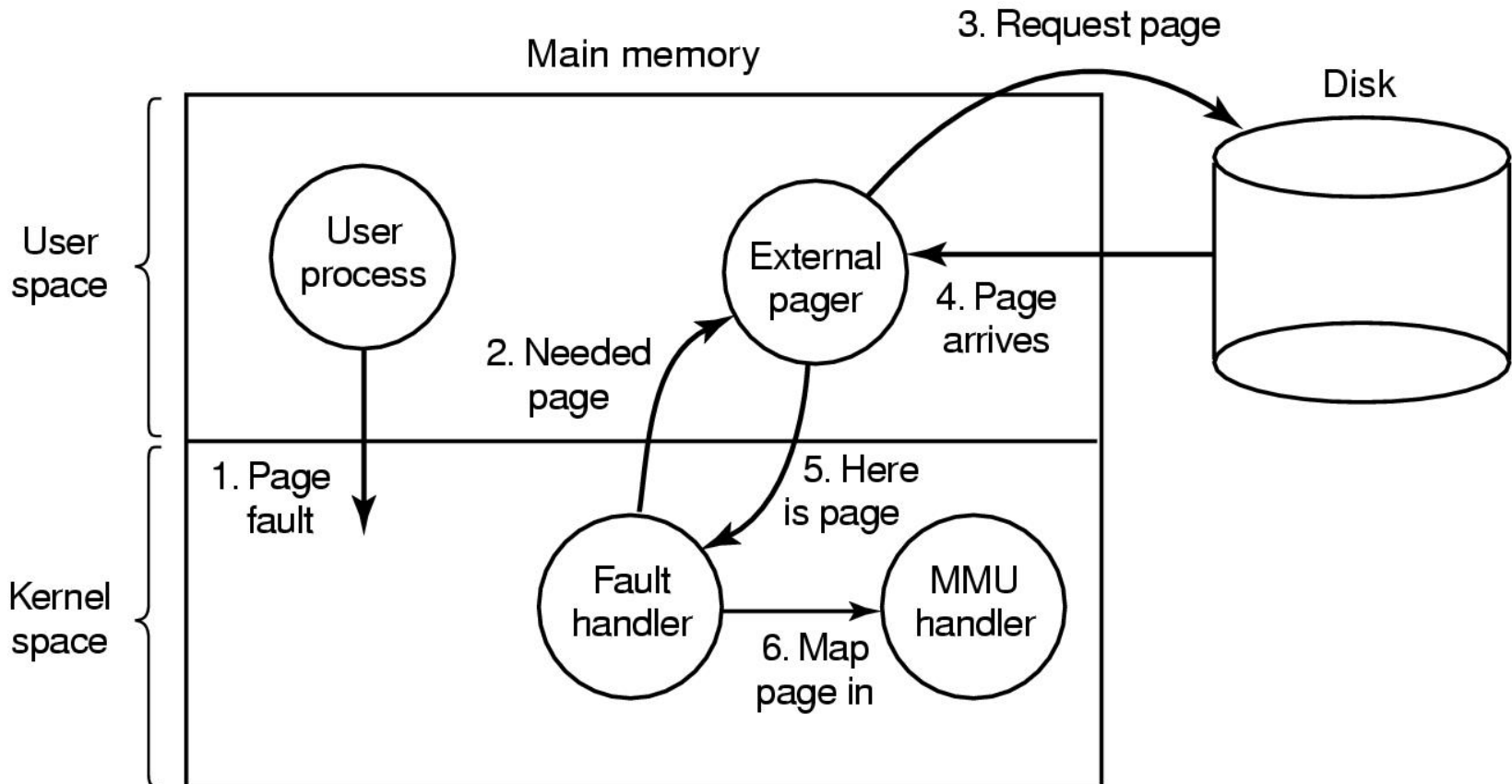
- La precedente strategia richiede però che sia possibile stabilire a priori la presenza o meno di una pagina in memoria
- Uso un bit aggiuntivo nella tabella delle pagine: bit di validità
- L'evento di pagina non trovata in memoria, è denominato **page fault** ed il suo verificarsi produce un'eccezione (#14), gestita da un apposito handler (page fault handler)
- Gestire il page fault significa preoccuparsi di recuperare la pagina da disco e portarla in memoria

Demand Paging

- In un sistema con memoria virtuale il tempo di page fault e la frequenza con cui ciò si verifica sono parametri critici, infatti se p è la probabilità che si verifichi un page fault:

$$\text{tempo medio di accesso} = \text{memory access} + p \cdot \text{tempo di page fault}$$

Gestione del Page Fault



Gestione del Page Fault (1)

1. Viene generata una trap di page fault
2. Salvataggio del contesto interrotto da parte dell'hw
3. Richiamo procedura di gestione dell'eccezione
4. Il SO individua il numero di pagina da caricare
5. Il SO verifica la validità di questo dato e cerca dei frame liberi
6. Se il frame selezionato è dirty, riscrivilo su disco

Page Fault Handling (2)

6. Il SO trasferisce la nuova pagina in memoria
7. La tabella delle pagine viene aggiornata
8. I registri vengono ripristinati, tenendo conto che l'istruzione che ha generato il fault deve essere "ripristinata"
9. Il processo che ha generato il fault viene rimesso in esecuzione

Problemi da affrontare

- La precedente procedura presuppone la soluzione dei seguenti problemi:
 - Come faccio il backup di un'istruzione?
 - Come e dove recupero l'indirizzo delle pagine che risiedono su disco?
 - Quali frame decido di rimpiazzare?

Page Replacement Algorithms

- Durante la gestione di un page fault può essere necessario:
 - Individuare le pagine da scaricare su disco
 - Salvare le pagine modificate durante l'esecuzione, quelle non modificate possono essere sovrascritte

Optimal Page Replacement Algorithm

- Sostituire, tra quelle presenti in memoria, la pagina che sarà referenziata il più tardi possibile
 - Ottimale ma non realizzabile
- Possibili stime
 - Tenere traccia dell'uso passato delle pagine e inferire il futuro dal passato
 - Anche quest'euristica è molto difficile da realizzare praticamente

Not Recently Used Page Replacement Algorithm

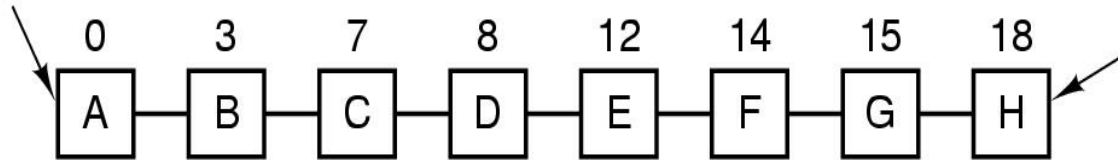
- Per ogni pagina l'hardware gestisce un Reference bit e un Modified bit
 - I bit vengono settati quando la pagina è acceduta e/o modificata
- Le pagine vengono così suddivise in quattro
 1. not referenced, not modified
 2. not referenced, modified
 3. referenced, not modified
 4. referenced, modified
- NRU rimuove le pagine casualmente a partire dalla classe con l'indice più basso non vuota

FIFO Page Replacement Algorithm

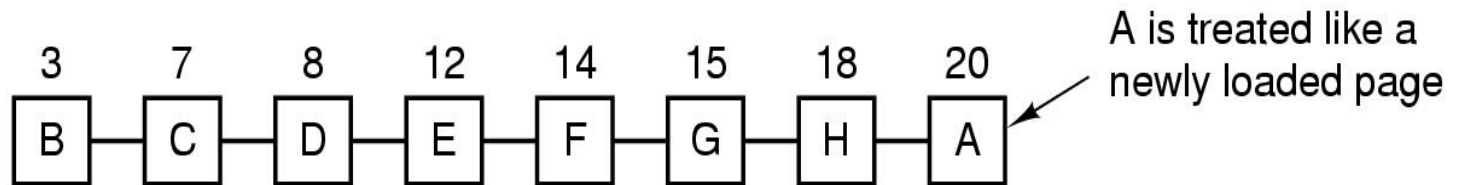
- Mantiene una lista delle pagine che rispetta l'ordine con cui le stesse sono state caricate in memoria (le pagine più “anziane” sono all'inizio della lista)
- Le pagine più anziane sono le prime ad essere sostituite
- Svantaggio
 - Sostituire una pagine che è referenziata spesso

Second Chance Page Replacement Algorithm

Page loaded first



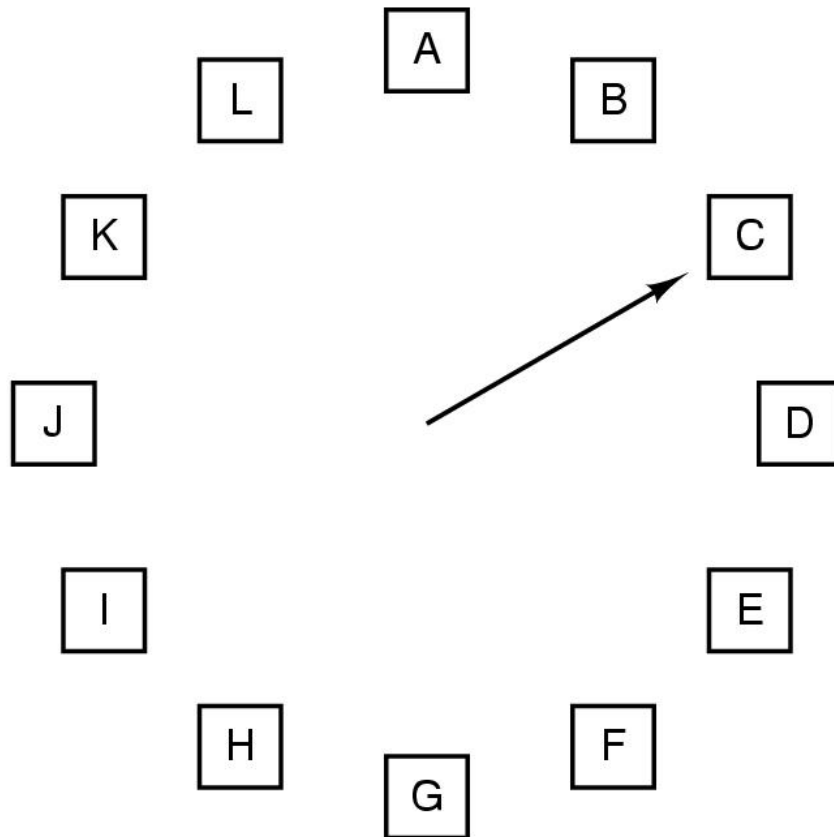
(a)



(b)

- Le pagine sono ordinate FIFO con un bit d'uso R
- La lista delle pagine quando si verifica un page fault all'istante 20, e A ha il bit R a uno

The Clock Page Replacement Algorithm



When a page fault occurs, the page the hand is pointing to is inspected. The action taken depends on the R bit:

R = 0: Evict the page

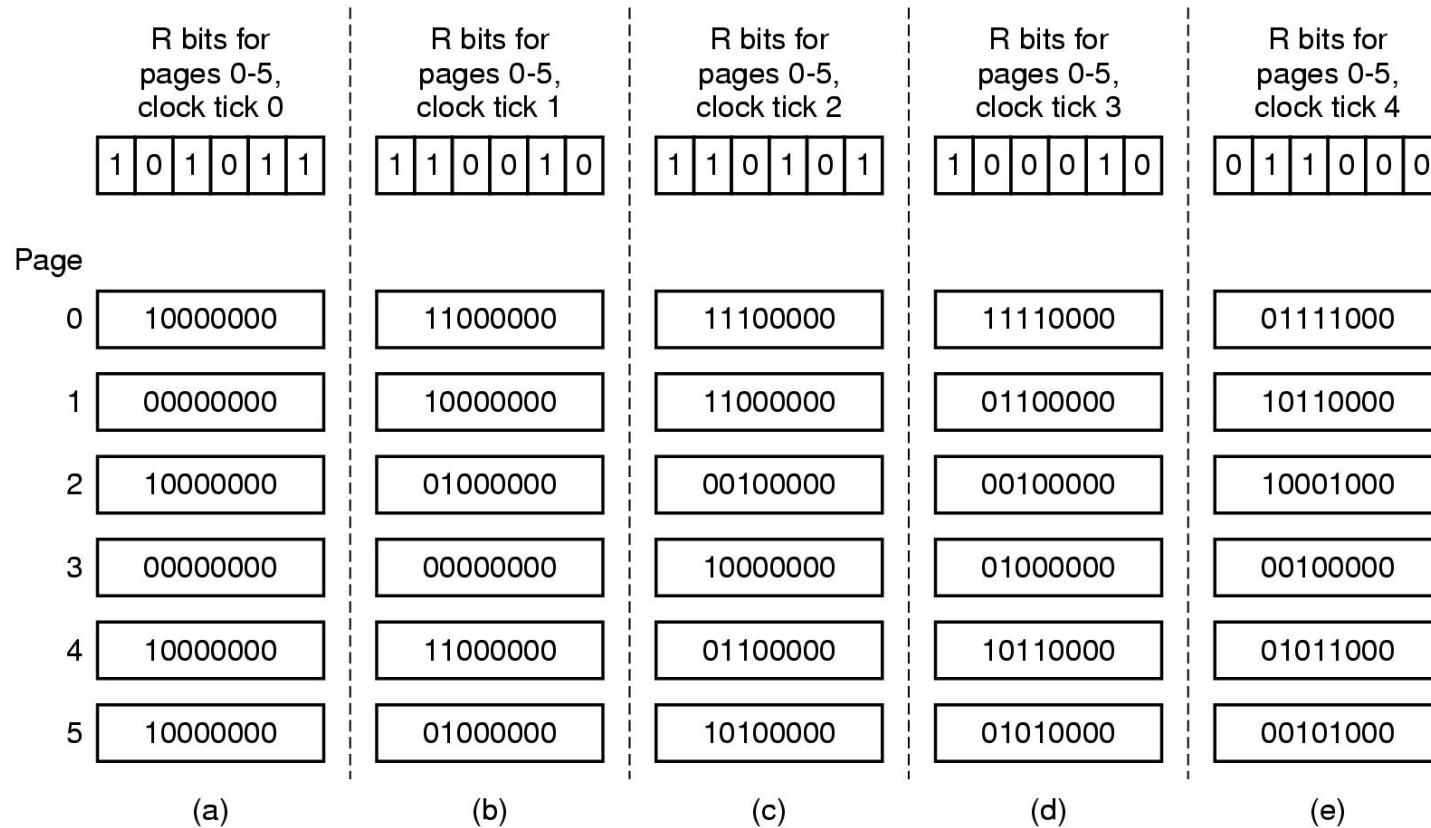
R = 1: Clear R and advance hand

- Come seconda chance ma usa una lista circolare, la lancetta punta alla pagina più vecchia

Least Recently Used (LRU)

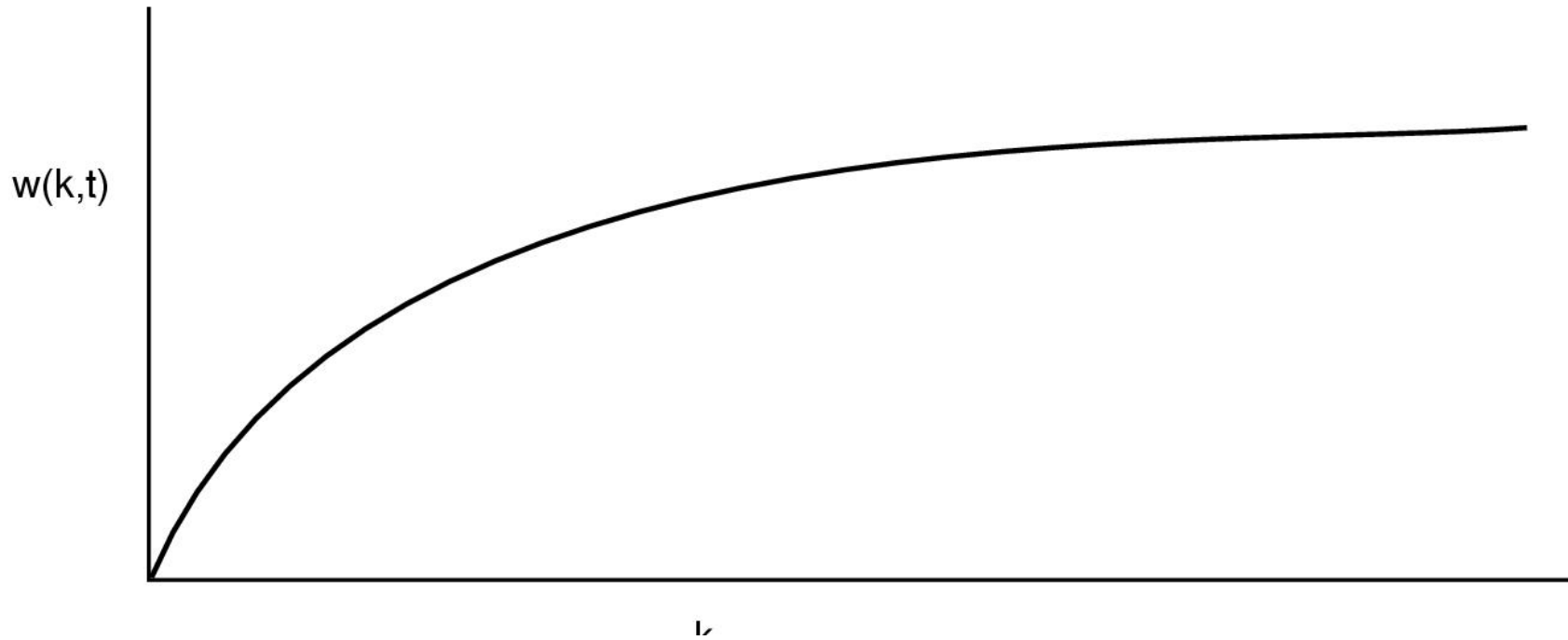
- Assume che le pagine usate recentemente saranno usate anche in un futuro prossimo
 - Elimina le pagine che non sono usate da più tempo
- Mantiene una lista linkata di pagine
 - Quelle usate più recentemente all'inizio,
 - È necessario aggiornare questa lista ad ogni riferimento a pagina !!
- Alternativamente è possibile mantenere un contatore per ciascun elemento della tabella delle pagine
 - La pagina vittima è quella con il contatore più basso
 - Il contatore viene azzerato periodicamente

Simulating LRU in Software



- L'algorithmo di aging simula LRU in software

The Working Set Page Replacement Algorithm (1)



- Il working set è costituito dall'insieme delle pagine usate negli ultimi k riferimenti a memoria
- $w(k,t)$ è la dimensione del working set all'istante t

Review of Page Replacement Algorithms

Algorithm	Comment
Optimal	Not implementable, but useful as a benchmark
NRU (Not Recently Used)	Very crude
FIFO (First-In, First-Out)	Might throw out important pages
Second chance	Big improvement over FIFO
Clock	Realistic
LRU (Least Recently Used)	Excellent, but difficult to implement exactly
NFU (Not Frequently Used)	Fairly crude approximation to LRU
Aging	Efficient algorithm that approximates LRU well
Working set	Somewhat expensive to implement
WSClock	Good efficient algorithm

ALTRE CRITICITÀ

Modeling Page Replacement Algorithms

Belady's Anomaly

All pages frames initially empty

	0	1	2	3	0	1	4	0	1	2	3	4	
Youngest page		0	1	2	3	0	1	4	4	4	2	3	3
			0	1	2	3	0	1	1	1	4	2	2
Oldest page				0	1	2	3	0	0	0	1	4	4
		P	P	P	P	P	P			P	P		

9 Page faults

(a)

	0	1	2	3	0	1	4	0	1	2	3	4	
Youngest page		0	1	2	3	3	3	4	0	1	2	3	4
			0	1	2	2	2	3	4	0	1	2	3
Oldest page				0	1	1	1	2	3	4	0	1	2
					0	0	0	1	2	3	4	0	1
		P	P	P	P			P	P	P	P	P	P

10 Page faults

(b)

- FIFO con 3 page frames
- FIFO con 4 page frames
- I page fault sono contraddistinti da P

Allocation Policy

	Age
A0	10
A1	7
A2	5
A3	4
A4	6
A5	3
B0	9
B1	4
B2	6
B3	2
B4	5
B5	6
B6	12
C1	3
C2	5
C3	6

(a)

A0
A1
A2
A3
A4
A6
B0
B1
B2
B3
B4
B5
B6
C1
C2
C3

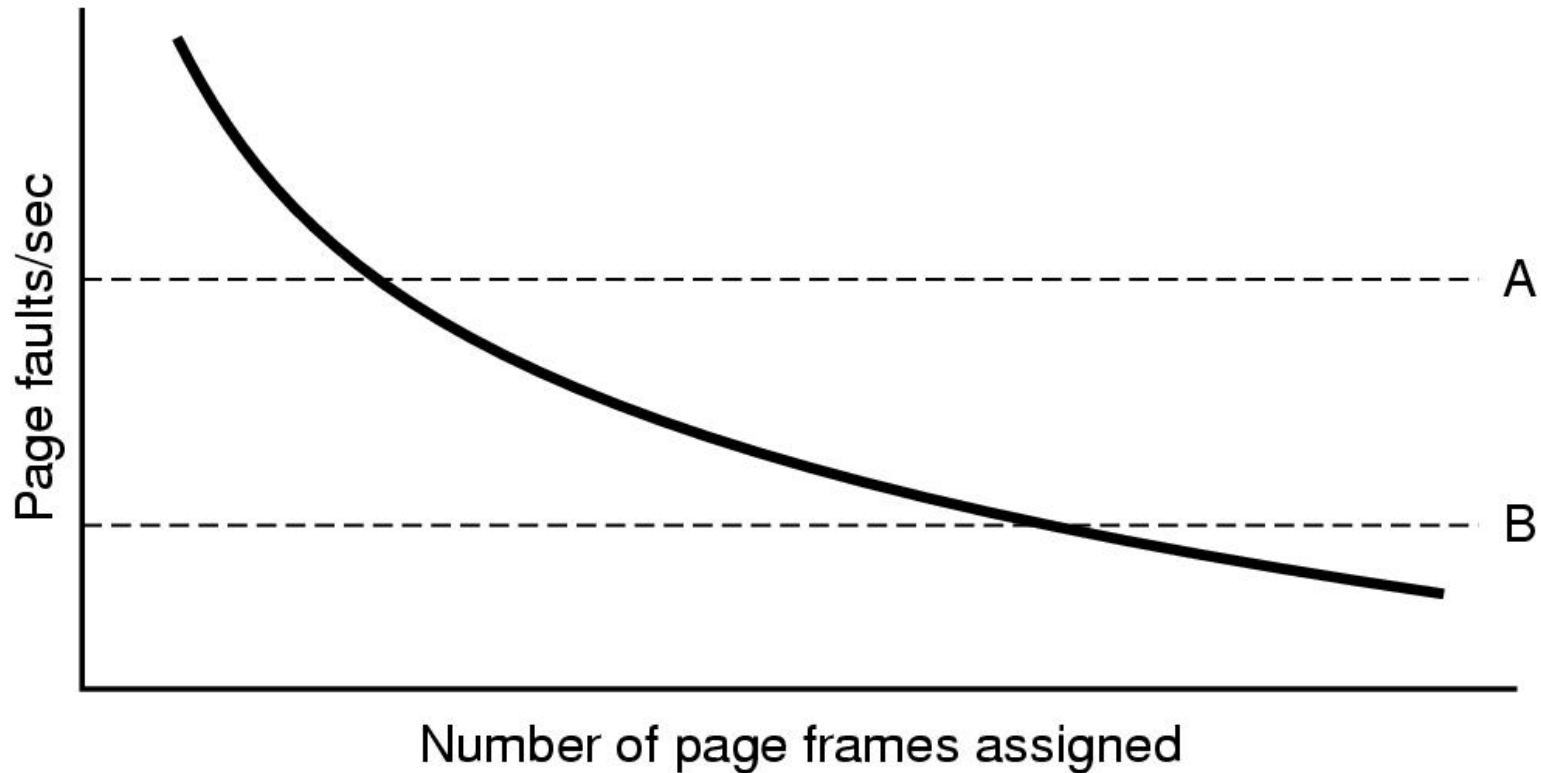
(b)

A0
A1
A2
A3
A4
A5
B0
B1
B2
A6
B4
B5
B6
C1
C2
C3

(c)

- a) Configurazione originale
- b) Rimpiazzamento locale
- c) Rimpiazzamento globale

Allocation Policy (2)



Uno sforzo va fatto per mantenere i parametri coinvolti all'interno della linea tratteggiata

Dimensione Pagine

- Vantaggi pagine piccole
 - Minore frammentazione interna
 - Più adattabili alle dimensioni dei programmi
 - Riduzione spazio di memoria inutilizzato
- Svantaggi
 - Grosse tabelle delle pagine
 - Più accessi a disco

Dimensione Pagine

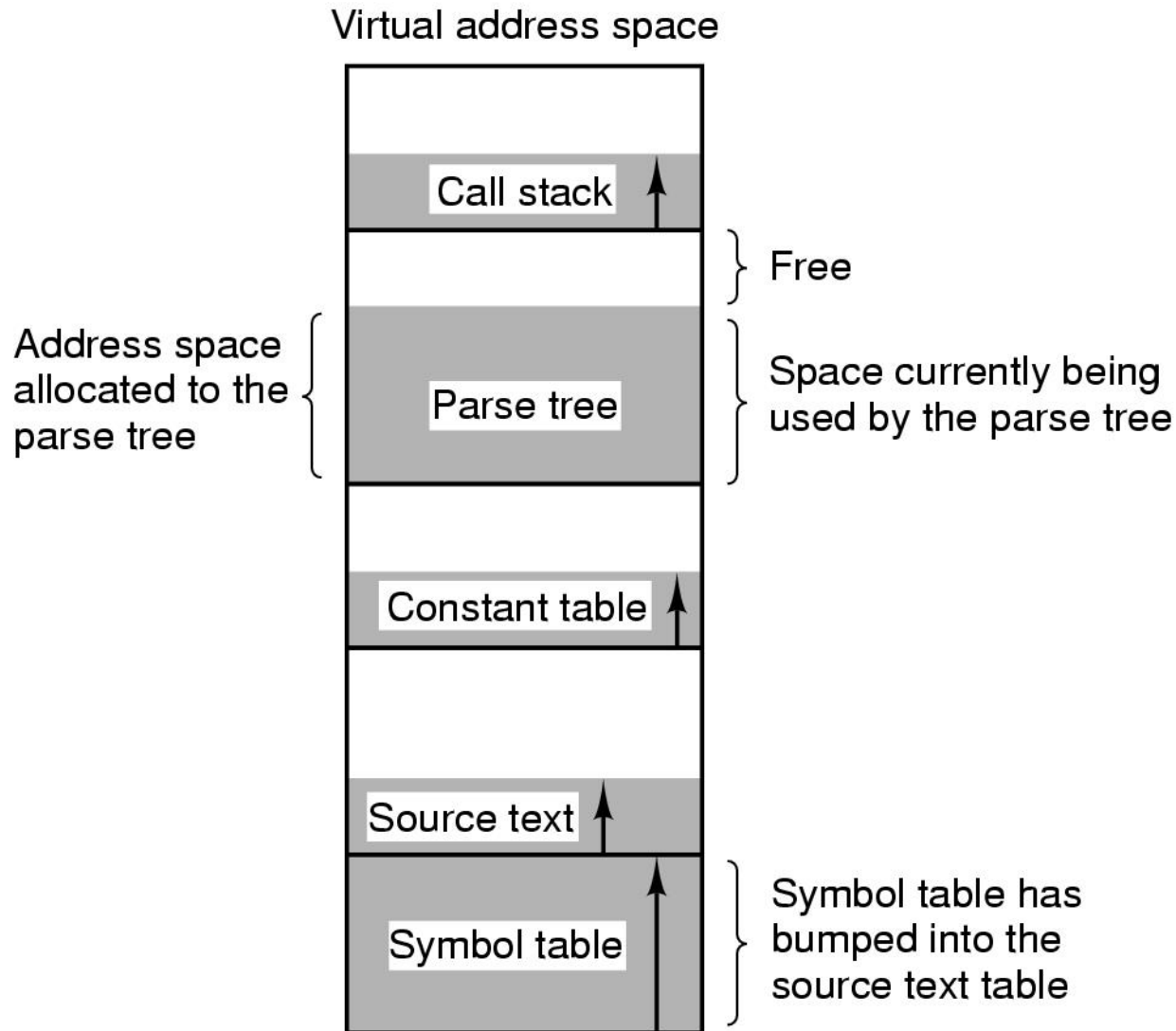
Table 8.2 Example Page Sizes

Computer	Page Size
Atlas	512 48-bit words
Honeywell-Multics	1024 36-bit word
IBM 370/XA and 370/ESA	4 Kbytes
VAX family	512 bytes
IBM AS/400	512 bytes
DEC Alpha	8 Kbytes
MIPS	4 kbytes to 16 Mbytes
UltraSPARC	8 Kbytes to 4 Mbytes
Pentium	4 Kbytes or 4 Mbytes
PowerPc	4 Kbytes

Segmentazione

- La memoria virtuale vista finora è monodimensionale
- Avere più spazi di indirizzi separati può essere vantaggioso in presenza di aree dati distinte che crescono durante l'esecuzione
 - Es. Compilatore che usa aree per
 - Testo sorgente
 - Symbol table
 - Tabella delle costanti intere e floating point
 - Albero di parsing
 - Stack per le chiamate di procedura del compilatore stesso

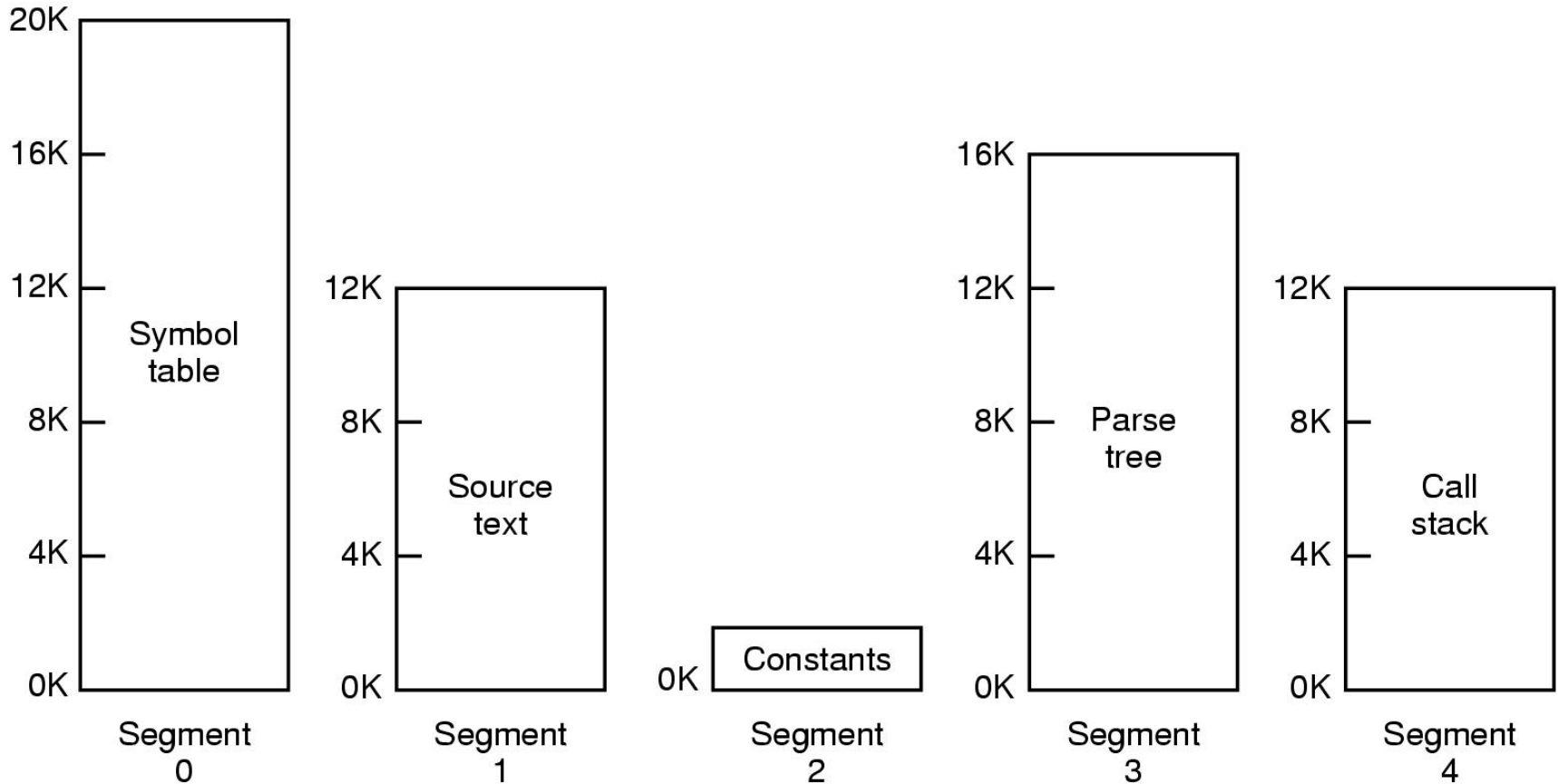
Segmentazione



Segmentazione

- Lo spazio di indirizzamento è costituito da un insieme di spazi di indirizzi logici indipendenti detti **segmenti**
- I segmenti hanno dimensioni diverse e possono cambiare senza interferire gli uni con gli altri
- L'utente definisce i segmenti ma non li gestisce
- Semplifica le operazioni di linking
 - Segmento, offset per indirizzare i moduli
 - I segmenti sono indipendenti
- Facilita la condivisione di librerie
- Soffre di frammentazione esterna

Segmentazione



Ogni segmento cresce o decresce autonomamente

Segmentazione

Consideration	Paging	Segmentation
Need the programmer be aware that this technique is being used?	No	Yes
How many linear address spaces are there?	1	Many
Can the total address space exceed the size of physical memory?	Yes	Yes
Can procedures and data be distinguished and separately protected?	No	Yes
Can tables whose size fluctuates be accommodated easily?	No	Yes
Is sharing of procedures between users facilitated?	No	Yes
Why was this technique invented?	To get a large linear address space without having to buy more physical memory	To allow programs and data to be broken up into logically independent address spaces and to aid sharing and protection

Segmentazione paginata

- Obiettivo: cogliere gli aspetti positivi di entrambe le soluzioni cercando di contenere gli effetti negativi
- Un sistema in cui i processi sono suddivisibili in segmenti che non devono essere necessariamente caricati interamente in memoria perché paginabili

Calcolo indirizzo

