

3 – Gestione e organizzazione della memoria

Organizzazione e gestione della memoria
 Gerarchie di memoria
 Allocazione di memoria contigua e non contigua
 Mono utente
 Multiprogrammazione con partizioni fisse e con partizioni variabili
 Swapping di memoria

Memoria Virtuale
 paginazione
 località

Mapping dei blocchi
Paginazione

- Traduzione di indirizzi di Paginazione:
 - Mapping diretto
 - Mapping associativo
 - Mapping combinato
 - Tabelle di pagine multilivello
 - Tabelle inversa delle pagine
 - Condivisione nei sistemi di paginazione

S. Balsamo – Università Ca' Foscari Venezia – SO.3.45

3 – Gestione e organizzazione della memoria

Paginazione: algoritmi di sostituzione di pagina
 Random
 First-In-First-Out (FIFO) e anomalie
 Least-Recently-Used (LRU)
 Least-Frequently-Used (LFU)
 Not-Recently-Used (NRU)
 Modifica a FIFO: Seconda-Chance e pagina clock
 Pagina lontana

Modello Working Set

Sostituzione di pagina con Page-Fault-Frequency (PFF)

Problemi di progettazione di sistemi con Paginazione
 Dimensione della pagina
 Comportamento dei programmi con paginazione
 Sostituzione di pagina globale vs. locale

S. Balsamo – Università Ca' Foscari Venezia – SO.3.46

3 – Gestione e organizzazione della memoria

Segmentazione
 Traduzione di indirizzi
 Condivisione nei sistemi con segmentazione
 Protezione e controllo degli accessi

Sistemi con Segmentazione/Paginazione
 Traduzione dinamica degli indirizzi
 Condivisione
 Protezione e controllo degli accessi

S. Balsamo – Università Ca' Foscari Venezia – SO.3.47

48

Obiettivi

- **Memoria Virtuale** vantaggi e svantaggi della paginazione a previsione e a richiesta
- sistemi di memoria virtuale di
 - con paginazione
 - con segmentazione
 - combinato segmentazione /paginazione
- problemi nella **sostituzione** delle pagine.
- **confronto tra strategie** di sostituzione delle pagine e ottimizzazione
- **impatto della dimensione pagina su prestazioni** della memoria virtuale
- **comportamento del programma** nella paginazione
- **condivisione e protezione** dei sistemi di memoria virtuale
- **hardware** che rende i sistemi di memoria virtuale possibili

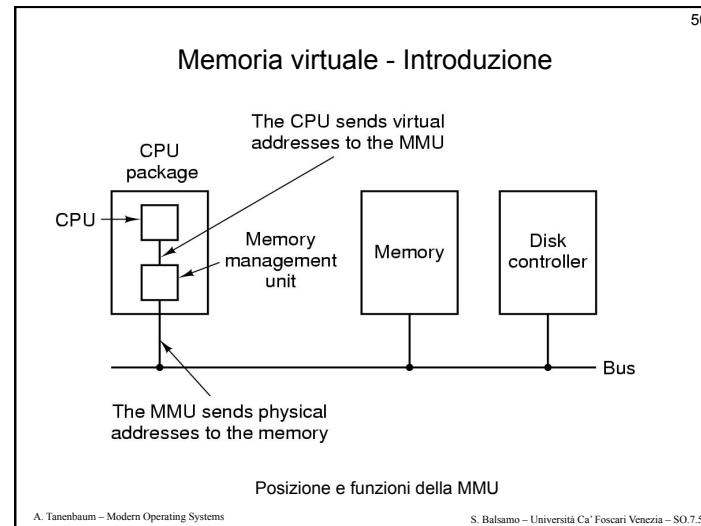
S. Balsamo – Università Ca' Foscari Venezia – SO.3.48

49

Memoria virtuale

- Risolve problemi di spazio di memoria limitato
- Crea l'illusione che esista **più memoria** di quella disponibile nel sistema
- Due tipi di **indirizzi** nei sistemi di memoria virtuale
 - **Indirizzi virtuali**
 - riferiti dai processi
 - **Indirizzi fisici**
 - descrive indirizzi nella memoria principale
- **Memory management unit (MMU)**
 - Componente hw dedicata alla **traduzione** di indirizzi virtuali in indirizzi fisici

S. Balsamo – Università Ca' Foscari Venezia – SO.7.49



51

Memoria virtuale - Introduzione

Evoluzione delle organizzazioni di memoria

	Real	Real	Virtual		
Single-user dedicated systems	Real memory multiprogramming systems		Virtual memory multiprogramming systems		
	Fixed-partition multi-programming	Variable-partition multi-programming	Pure paging	Pure segmentation	Combined paging and segmentation
	Absolute	Re-locatable			

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.51

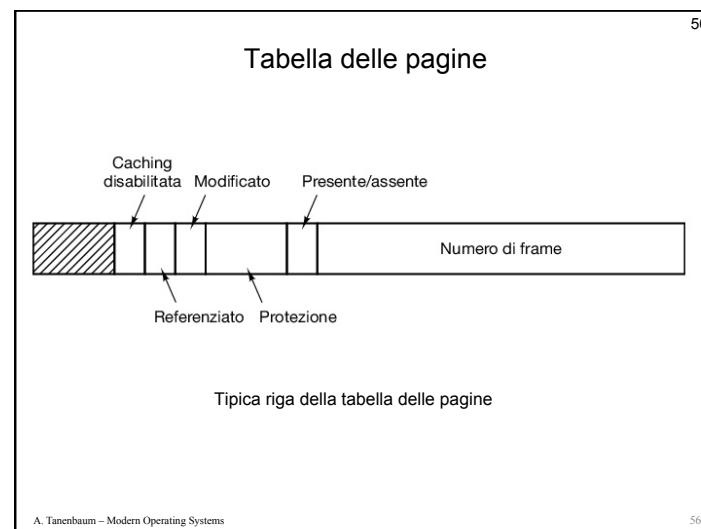
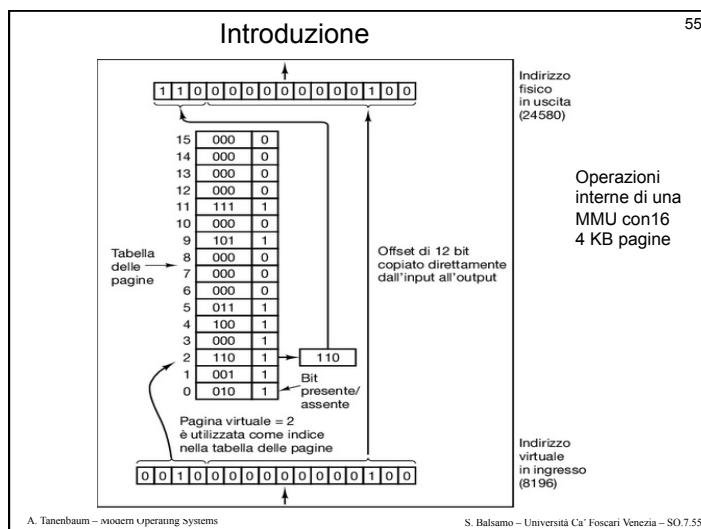
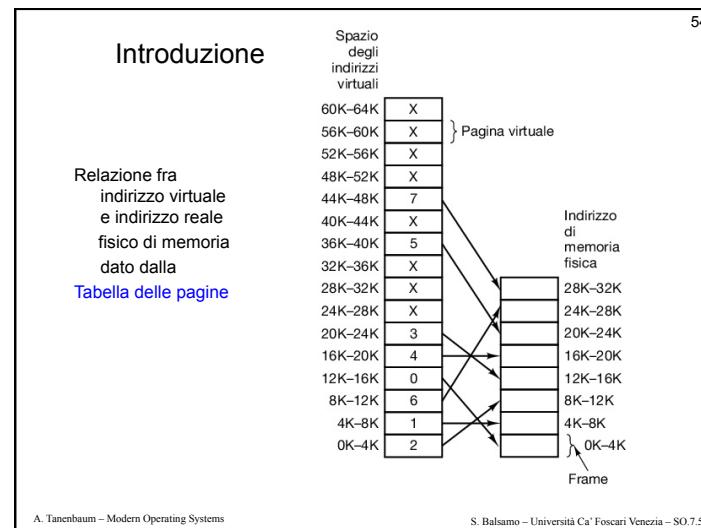
- 52
- ## Memoria virtuale: concetti di base
- **Spazio di indirizzamento virtuale V**
 - Intervallo di indirizzi virtuali ai quali un processo può fare riferimento
 - **Spazio di indirizzamento reale R**
 - Intervallo di indirizzi fisici disponibili su un particolare sistema di computer
 - Meccanismo di **traduzione dinamica** degli indirizzi (DAT – Dynamic Address Translation)
 - Converte indirizzi virtuali in indirizzi fisici durante l'esecuzione del programma
- S. Balsamo – Università Ca' Foscari Venezia – SO.7.52

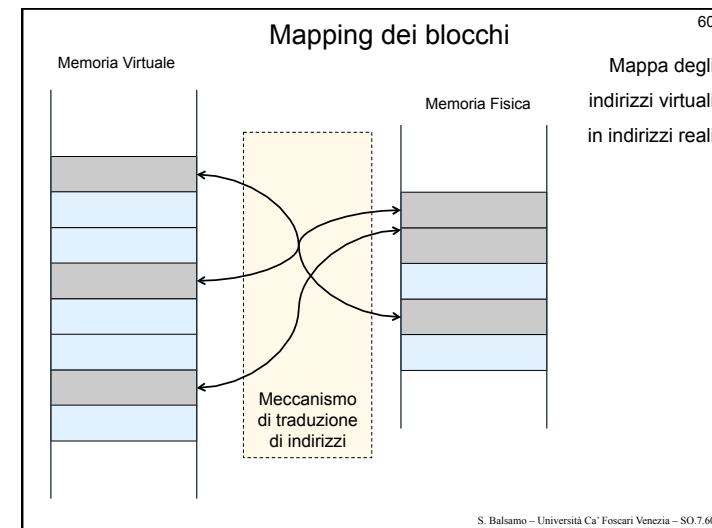
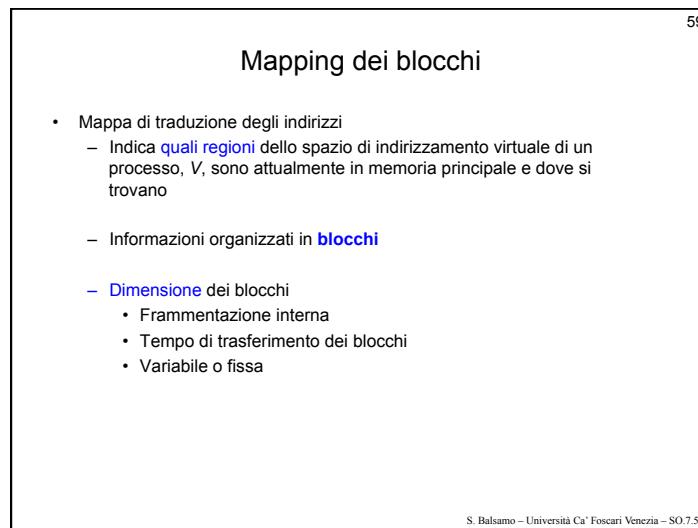
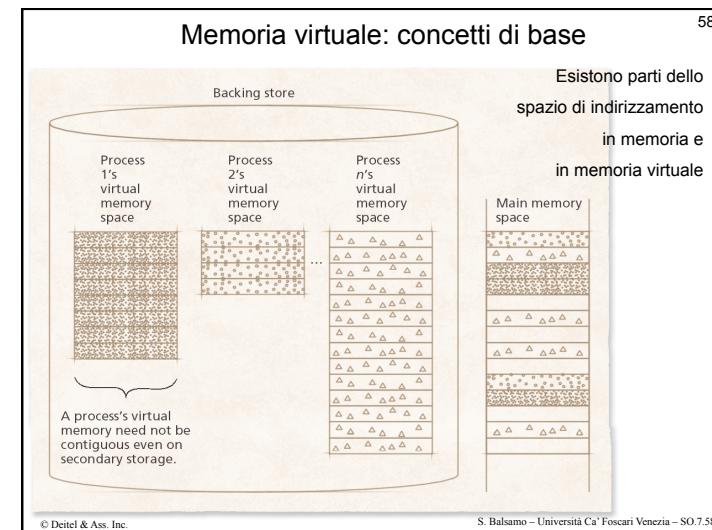
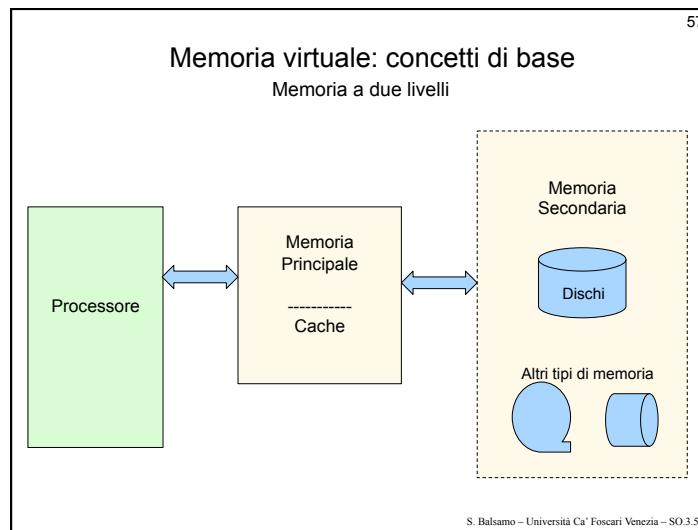
53

Memoria virtuale: concetti di base

- Spesso $|V| >> |R|$
 - S.O. deve memorizzare parti di V per ogni processo al di fuori della memoria principale
 - Schema di memoria a due livelli
 - S.O. scambia porzioni di V tra memoria principale (e la cache) e memoria secondaria

S. Balsamo – Università Ca' Foscari Venezia – SO.7.53



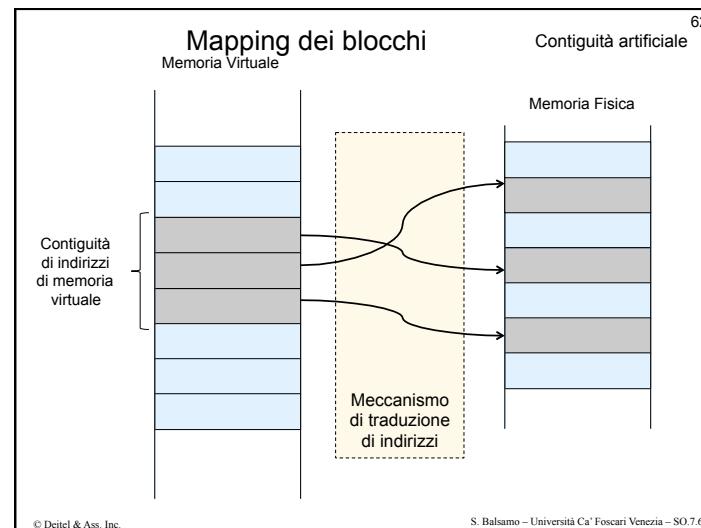


61

Mapping dei blocchi

- Contiguità artificiale
 - indirizzi virtuali contigui potrebbero non corrispondere agli indirizzi di memoria reale contigui
 - Il programmatore vede gli [indirizzi virtuali contigui](#)

S. Balsamo – Università Ca' Foscari Venezia – SO.7.61

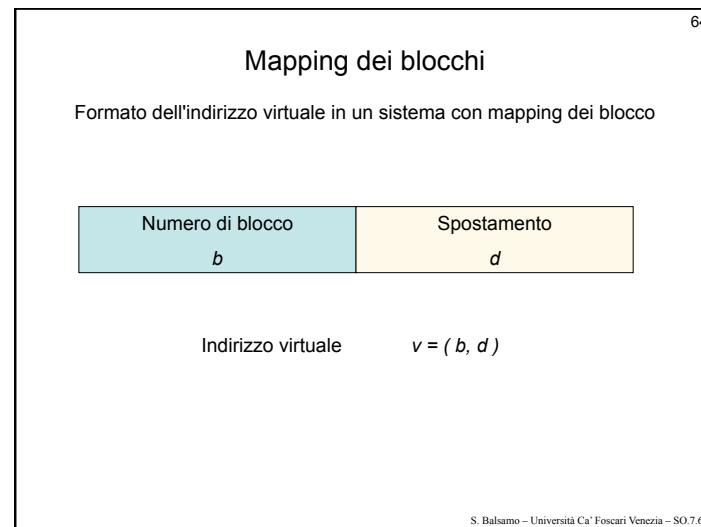


63

Mapping dei blocchi

- **Pagine**
 - I blocchi sono di dimensione **fissa**
 - La tecnica è chiamata [paginazione](#)
- **Segmenti**
 - I blocchi possono avere dimensioni **diverse**
 - La tecnica è chiamata [segmentazione](#)
- **Mappa dei blocchi**
 - Sistema che rappresenta gli indirizzi come coppie ordinate

S. Balsamo – Università Ca' Foscari Venezia – SO.7.63



Mapping dei blocchi

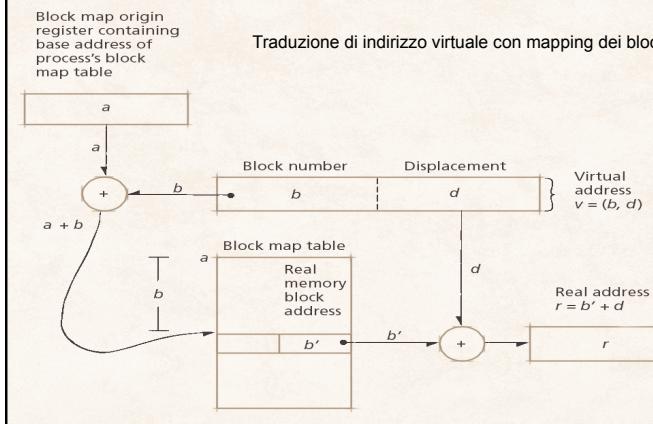
- Dato un indirizzo virtuale $v = (b, d)$ si traduce nell'indirizzo reale r
 - tabella della mappa dei blocchi per ogni processo
 - Una riga per ogni blocco del processo, ordinate
 - Indirizzo reale a della tabella del nuovo processo da caricare
 - Registro origine della tabella della mappa dei blocchi
- Traduzione
 - Si aggiunge ad a il numero di blocco, b , per individuare la riga nella tabella di mappa dei blocchi
 - La riga produce l'indirizzo, b' , inizio del blocco b in memoria principale
 - Si aggiunge lo spostamento d a b' per formare l'indirizzo reale, r

S. Balsamo – Università Ca' Foscari Venezia – SO.7.65

65

Mapping dei blocchi

Traduzione di indirizzo virtuale con mapping dei blocchi



S. Balsamo – Università Ca' Foscari Venezia – SO.7.66

66

Paginazione

- Paginazione usa il mapping di blocchi a dimensione fissa
 - L'indirizzo virtuale nel sistema di paginazione è una coppia ordinata $v = (p, d)$
 - p è il numero della pagina in memoria virtuale su cui risiede il dell'elemento di riferimento
 - d è lo spostamento dall'inizio della pagina p in cui si trova l'elemento di riferimento

S. Balsamo – Università Ca' Foscari Venezia – SO.7.67

67

Paginazione

Formato dell'indirizzo virtuale in un sistema con paginazione puro

Numero di pagina p	Spostamento d
-------------------------	--------------------

Indirizzo virtuale $v = (p, d)$

S. Balsamo – Università Ca' Foscari Venezia – SO.7.68

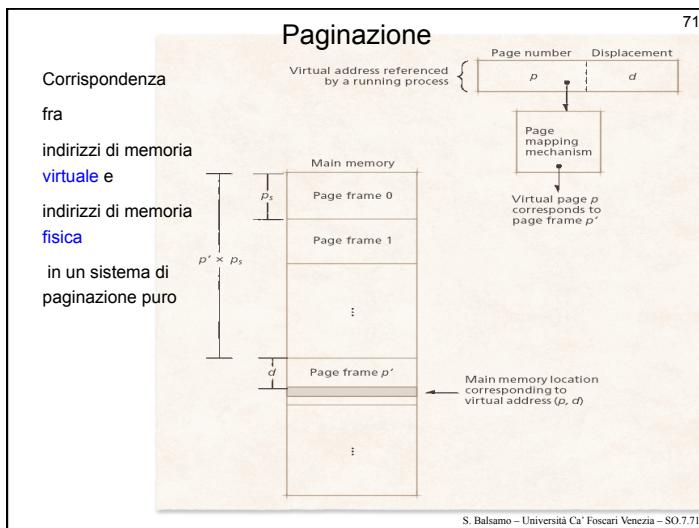
6

69

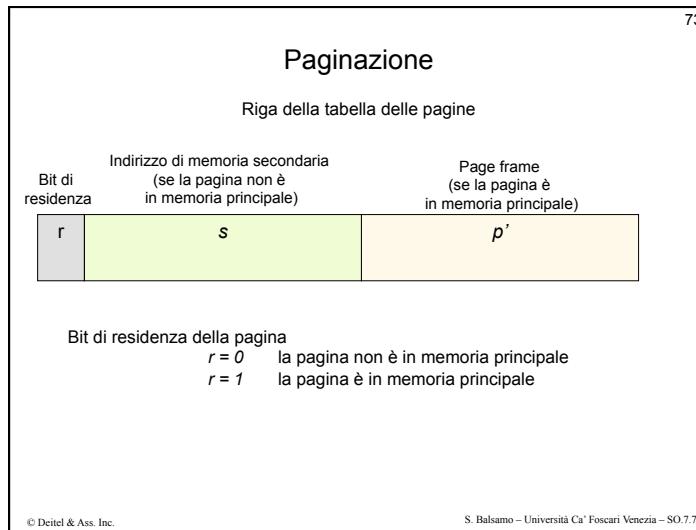
Paginazione

- **Page frame**
 - Blocchi della **memoria principale** a dimensione **fissa** destinati a contenere pagine
 - Inizia a un indirizzo di memoria principale che è un multiplo intero della dimensione fissa della pagina (p_s)
 - Il sistema può posizionare le pagine in un qualsiasi page frame libero

S. Balsamo – Università Ca' Foscari Venezia – SO.7.69



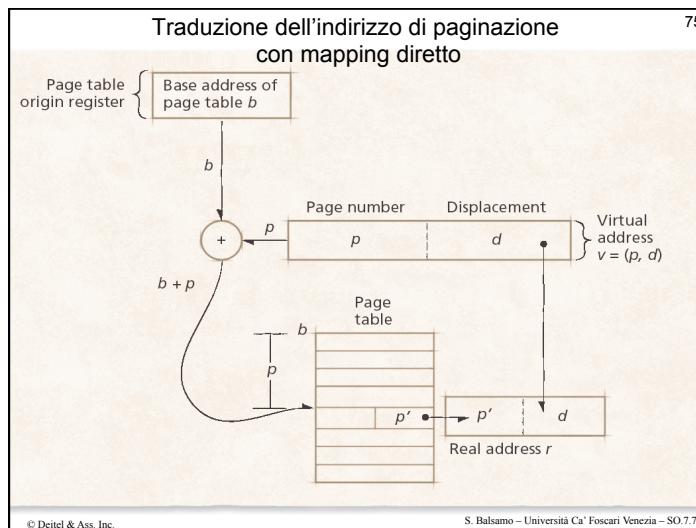
- 72
- ## Paginazione
- **Tabella delle pagine** pagina virtuale \rightarrow page frame
 - ma non tutte le pagine risiedono in memoria principale
 - **Page table entry (PTE)**
(riga della tabella delle pagine)
 - Indica che la **pagina virtuale p** è nel **page frame p'**
 - Contiene un **bit di residenza**, r , per indicare se la pagina è in memoria principale
 - $r = 1$, PTE memorizza il numero di page frame p'
 - $r = 0$, PTE memorizza la posizione della pagina in memoria secondaria
- S. Balsamo – Università Ca' Foscari Venezia – SO.7.72



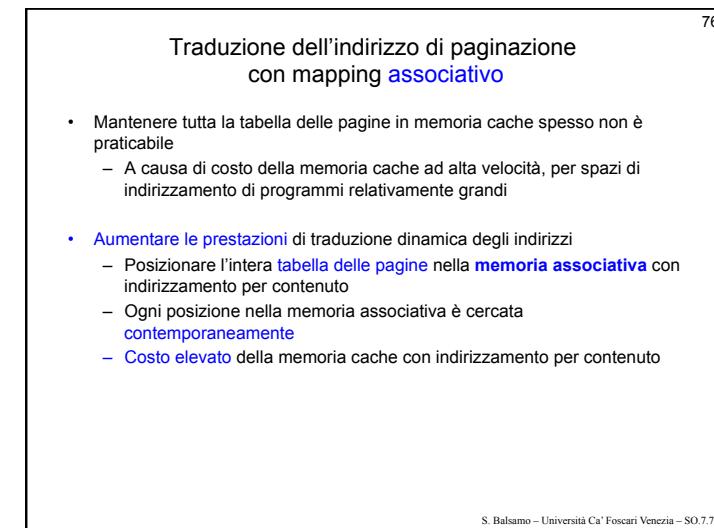
73

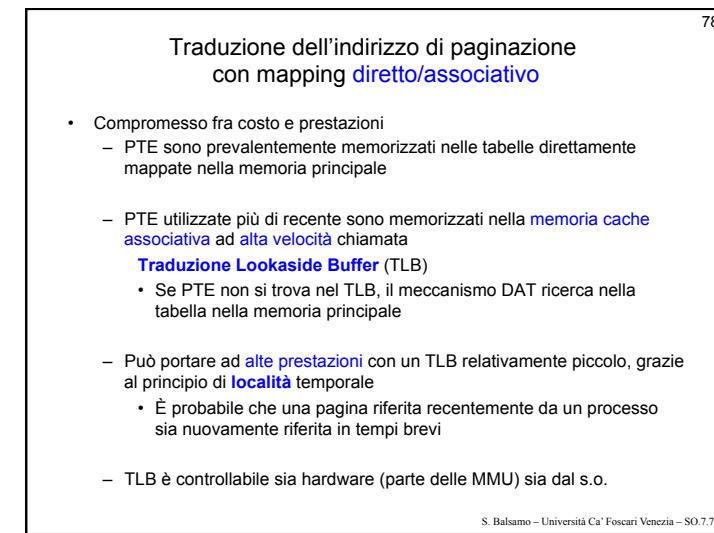
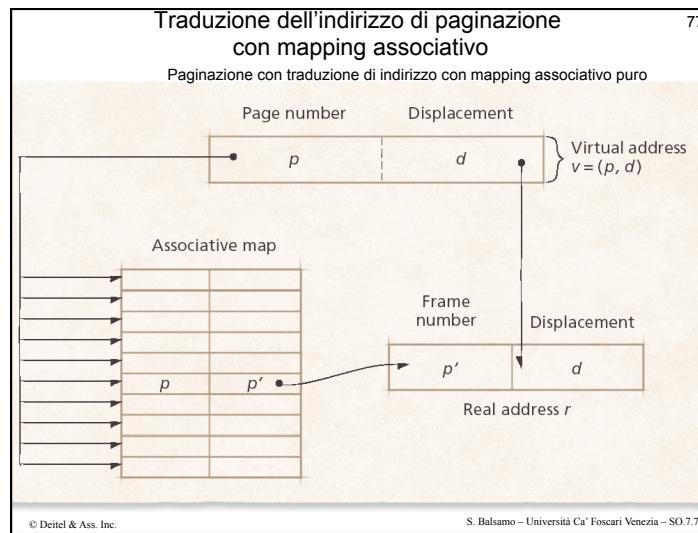


74



75





TLBs – Translation Lookaside Buffers 80

Valido	Pagina virtuale	Modificato	Protezione	Frame
1	140	1	RW	31
1	20	0	R X	38
1	130	1	RW	29
1	129	1	RW	62
1	19	0	R X	50
1	21	0	R X	45
1	860	1	RW	14
1	861	1	RW	75

Una TLB per accelerare la paginazione

A. Tanenbaum – Modern Operating Systems

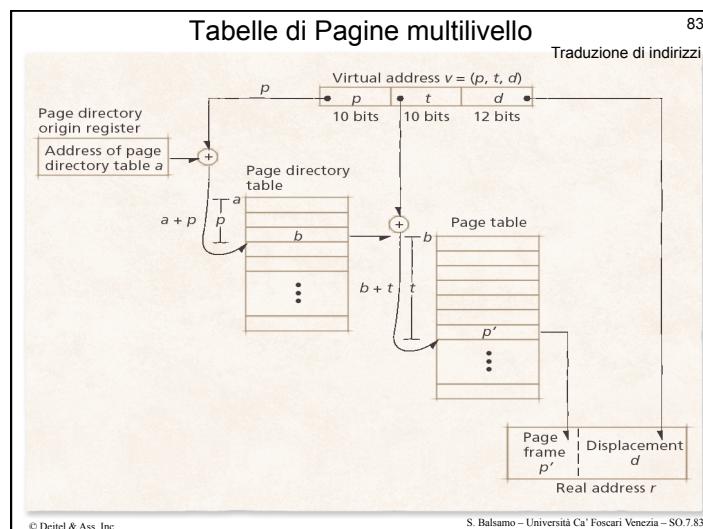
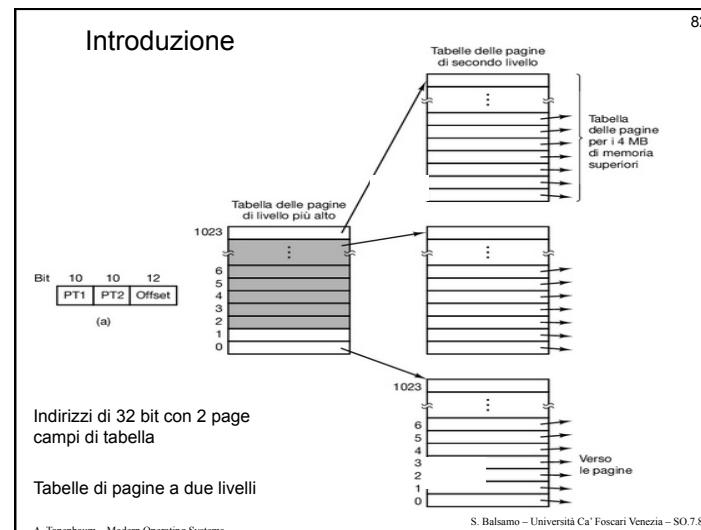
S. Balsamo – Università Ca' Foscari Venezia – SO.7.80

Tabelle di Pagine multilivello

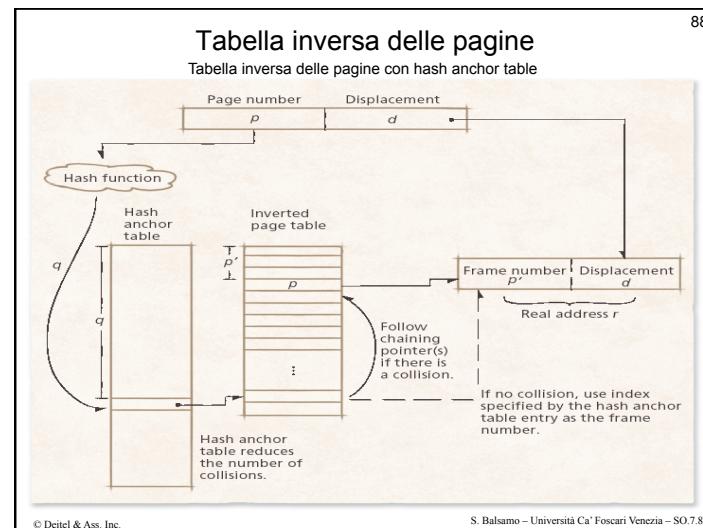
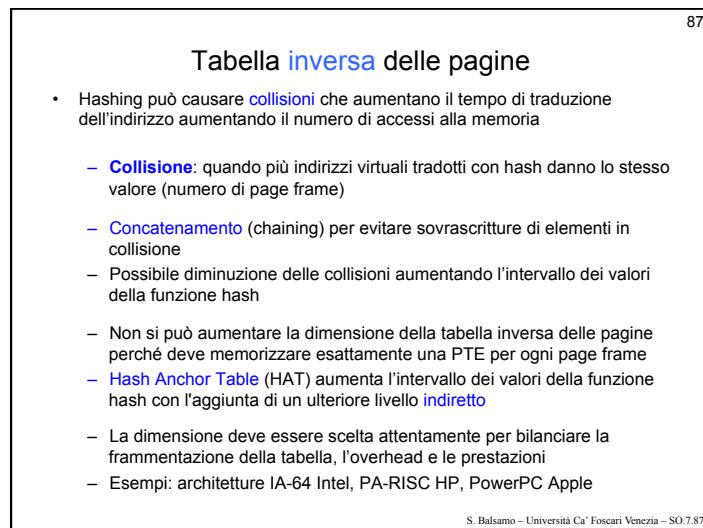
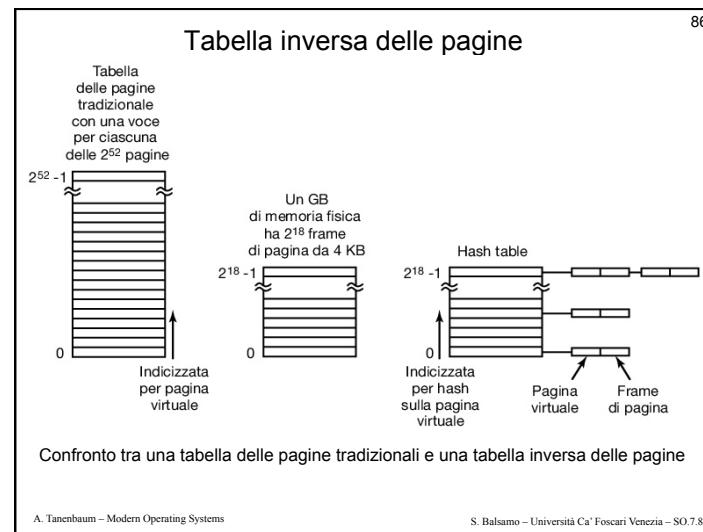
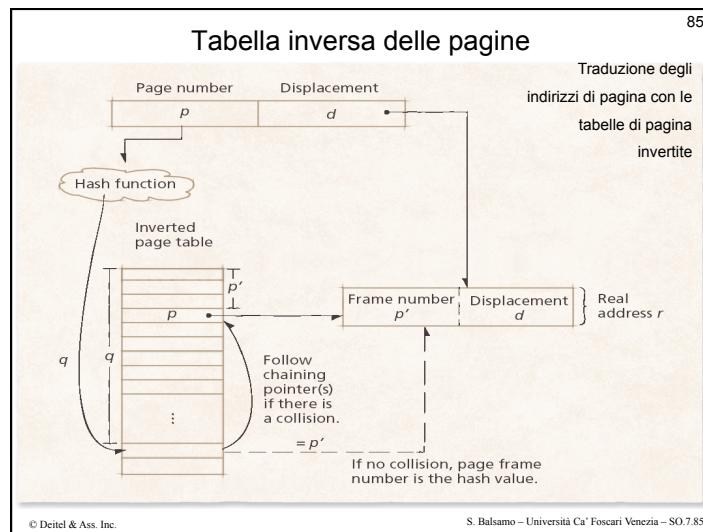
81

- Tabelle di pagine **multilivello**
 - Sistema in grado di **memorizzare locazioni non contigue** di memoria quelle parti della **tabella delle pagine** di processo che il processo sta utilizzando
 - Riduce** il numero di righe della tabella in memoria
 - Gerarchia** delle tabelle di pagina
 - Ogni livello contiene una tabella che memorizza i puntatori alle tabelle del livello inferiore
 - I livelli più bassi hanno tabelle con le traduzioni di indirizzi
 - Può **ridurre l'overhead** di memoria rispetto al sistema di mappatura diretta
 - Richiede un accesso alla memoria in più per il mapping, limitabile con **località** e uso di TLB

S. Balsamo – Università Ca' Foscari Venezia – SO.7.81



- ## Tabella inversa delle pagine
- 84
- Tabelle **inversa** delle pagine
 - La tabella multilivello **riduce** il numero di righe della tabella in memoria, utile se il processo usa solo una parte dello spazio di indirizzamento
 - Altrimenti non è conveniente
 - La **tabella inversa** memorizza un **PTE** (riga della tabella delle pagine) in memoria per ogni **page frame** del sistema
 - Il numero di righe della tabella in memoria dipende non dallo spazio, V , ma dal numero di **page frame** (memoria fisica), R
 - È l'inverso rispetto ad una tabella di pagine tradizionale
 - Tradizionale: pagina virtuale \rightarrow page frame
 - Inverse: page frame \rightarrow pagina virtuale
 - Non memorizza informazioni relative alla memoria secondaria
 - Utilizza **funzioni hash** per mappare l'indirizzo virtuale nella riga nella tabella inversa delle pagine
- S. Balsamo – Università Ca' Foscari Venezia – SO.7.84



89

Condivisione in un sistema di paginazione

- Condivisione nei sistemi multiprogrammati
 - Riduce la memoria consumata dai programmi che utilizzano i dati e/o istruzioni in comune
 - Richiede che il sistema identifichi ogni pagina come condivisibile o non condivisibile
 - Procedure o dati
 - Procedure non modificabili: *rientranti*
 - Problema della modificabilità

S. Balsamo – Università Ca' Foscari Venezia – SO.7.89



91

Condivisione in un sistema di paginazione

- Esempio
 - `fork` in Unix crea un processo duplicato inizialmente identico
- Tecnica del *copy-on-write*
 - Nel momento della modifica viene creata una copia
 - Miglior uso della memoria
 - Maggiore velocità di creazione dei processi
 - Overhead nel momento della copia

S. Balsamo – Università Ca' Foscari Venezia – SO.7.91

92

Sostituzione in un sistema di paginazione

- Strategie di **sostituzione**
 - Tecnica usata da un sistema per **selezionare le pagine** da sostituire quando la memoria è piena
 - Determina **dove caricare** in memoria principale una pagina o un segmento da inserire
- Strategie di **Fetch**
 - Determina **quando** le pagine o segmenti devono essere caricati nella memoria principale
 - Strategie **a richiesta**
 - nel momento in cui il processo vi fa riferimento
 - Strategie **a previsione**
 - Uso di euristiche per prevedere a quali pagine un processo farà presto riferimento e caricamento di quelle pagine o segmenti
 - Maggiore overhead

S. Balsamo – Università Ca' Foscari Venezia – SO.8.92

93

Località

- Processo tende a fare riferimento alla memoria secondo modelli altamente localizzati
 - Località **spaziale**
 - Località **temporale**
 - In sistemi con paginazione, i processi tendono a favorire determinati **sottoinsiemi** delle loro pagine, e tali pagine tendono ad essere **adiacenti** nello spazio di indirizzamento **virtuale** del processo
 - Osservazioni empiriche

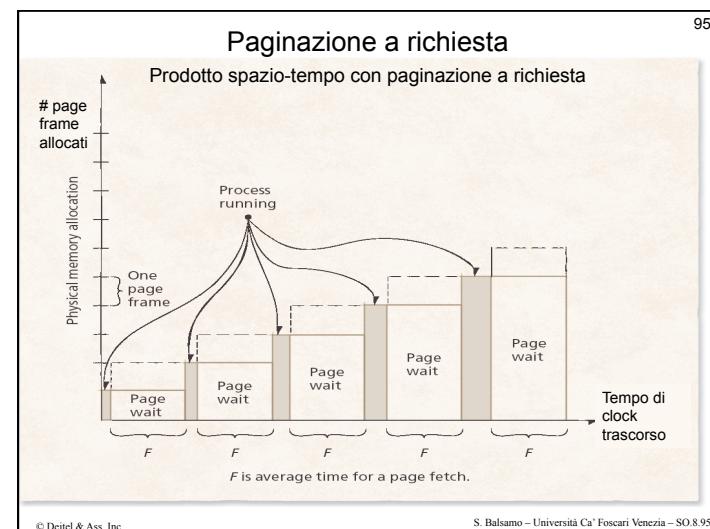
S. Balsamo – Università Ca' Foscari Venezia – SO.8.93

94

Paginazione a richiesta

- Quando un processo **inizia** l'esecuzione, il sistema **carica** in a memoria principale la **pagina** che contiene la sua **prima istruzione**
- Successivamente il sistema carica una pagina dalla memoria secondaria alla memoria principale **solo quando** il processo vi fa **esplicito riferimento**
- Richiede al processo di accumulare pagine una alla volta

S. Balsamo – Università Ca' Foscari Venezia – SO.8.94



96

Paginazione a previsione *prefetching*

- Il Sistema Operativo cerca di prevedere **di quali pagine** un processo **avrà bisogno** e **pre-carica** queste pagine quando vi è spazio di memoria disponibile
- Strategia di paginazione **a previsione**
 - Deve essere progettato attentamente in modo che l'overhead per implementarla non riduca le prestazioni del sistema
 - **Quanta memoria** pre-allocare
 - **Quante pagine** pre-caricare ogni volta
 - **Quale politica**

S. Balsamo – Università Ca' Foscari Venezia – SO.8.96

97

Sostituzione di pagina

- Quando un processo genera un *page fault*, il gestore della memoria deve
 - individuare la pagina di memoria **secondaria** a cui fa riferimento,
 - caricarla nel **page frame** della memoria **principale** e
 - aggiornare la riga corrispondente della **tavella delle pagine**
- Bit di modifica (*dirty bit*)
 - 1 se la pagina è stata modificata, 0 altrimenti
 - aiuta il sistema a determinare rapidamente la modifica di pagina
- Scrittura su disco (*flush*) di una pagina modificata: *overhead*
 - Sistemi con *flush* periodico automatico (preventivo)

S. Balsamo – Università Ca' Foscari Venezia – SO.8.97

98

Strategie di sostituzione delle pagine

- **Strategia di sostituzione di pagina Ottima** (OPT o MIN)
 - Ottiene prestazioni ottime, sostituisce la pagina alla quale non verrà fatto riferimento nuovamente più avanti nel futuro
 - Minimo numero di page fault => minimo tempo di attesa
 - Ideale
 - Conoscenza del **comportamento futuro** esatto del processo
- **Strategie di sostituzione di pagina:** caratteristiche
 - **Euristiche** per selezionare una pagina da sostituire
 - Approssimazioni della previsione del comportamento futuro
 - Overhead

S. Balsamo – Università Ca' Foscari Venezia – SO.8.98

99

Sostituzione casuale delle pagine (RAND)

- Strategia di sostituzione di pagina con **basso overhead** e **semplice**
- **Veloce**
- Non discrimina tra i processi
- **Equa:** ogni pagina nella memoria principale ha una **uguale probabilità** di essere **selezionata** per la sostituzione
- Potrebbe facilmente selezionare come prossima pagina da sostituire anche la pagina che alla quale verrà fatto il prossimo riferimento
- Poco usata

S. Balsamo – Università Ca' Foscari Venezia – SO.8.99

100

Sostituzione delle pagine First-In-First-Out (FIFO)

- Sostituisce la pagina che è **stata nel sistema il più a lungo** (ordine in base al tempo di arrivo)
- Può sostituire anche le pagine molto utilizzate
- Può essere implementata con **overhead** relativamente **basso**
- Non conveniente (non praticabile) per la maggior parte dei sistemi
- Base per altre strategie

S. Balsamo – Università Ca' Foscari Venezia – SO.8.100

Sostituzione delle pagine First-In-First-Out (FIFO)

101

Page reference	Result	FIFO page replacement with three pages available
A	Fault	A - -
B	Fault	B A -
C	Fault	C B A
A	No fault	C B A
D	Fault	D C B
A	Fault	A D C
		⋮ ⋮ ⋮

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.8.101

11.6.3 Anomalia della strategia FIFO

- Anomalia di Belady (o FIFO)
 - Alcuni modelli di riferimento di pagina effettivamente causano un **aumento del numero di errori di pagina all'aumentare del numero di page frame assegnati a un processo**
 - controiduttivo

S. Balsamo – Università Ca' Foscari Venezia – SO.8.102

Anomalia della strategia FIFO

I page faults possono aumentare con il numero di page frame allocati.

103

Page reference	Result	FIFO page replacement with three pages available	FIFO page replacement with four pages available
A	Fault	A - -	Fault
B	Fault	B A -	Fault
C	Fault	C B A	Fault
D	Fault	D C B	Fault
A	Fault	A D C	No fault
B	Fault	B A D	No fault
E	Fault	E B A	Fault
A	No fault	E B A	Fault
B	No fault	E B A	Fault
C	Fault	C E B	Fault
D	Fault	D C E	Fault
E	No fault	D C E	Fault

Three "no faults"

Two "no faults"

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.8.103

Anomalia della strategia FIFO

All pages frames initially empty

104

Youngest page	0 1 2 3 0 1 4 0 1 2 3 4
Oldest page	0 1 2 3 0 1 4 4 4 2 3 3
	P P P P P P P P P P P P
	9 Page faults
(a)	
Youngest page	0 1 2 3 0 1 4 0 1 2 3 4
Oldest page	0 1 2 2 2 3 4 0 1 2 3
	P P P P P P P P P P P P
	10 Page faults
(b)	
• FIFO con 3 page frame	
• FIFO con 4 page frame	
• P indica quale riferimento di pagina mostra page fault	

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.7.104

105

Sostituzione di pagina Least-Recently-Used (LRU)

- Sfrutta la **località temporale** sostituendo la **pagina** che ha trascorso più tempo in memoria senza essere riferita
- Può portare a prestazioni migliori rispetto FIFO
- Maggior overhead di sistema
 - Aggiornamento dei link ad ogni accesso alla memoria
 - Lista delle pagine
 - Oppure uso di un **contatore** per ogni riga di pagina nella tabella
- LRU può portare a scarse prestazioni se la pagina meno utilizzata di recente è la prossima pagina a cui fa riferimento un programma in una iterazione all'interno di un ciclo che fa riferimento a diverse pagine
 - Ad ogni sostituzione la pagina se riferita va ricaricata

S. Balsamo – Università Ca' Foscari Venezia – SO.8.105

Sostituzione di pagina Least-Recently-Used (LRU)

Page reference	Result	LRU page replacement with three pages available		
		A	–	–
A	Fault	B	A	–
B	Fault	C	B	A
C	No fault	B	C	A
B	No fault	B	C	A
B	No fault	A	B	C
A	Fault	D	A	B
D	No fault	A	D	B
A	No fault	B	A	D
B	Fault	F	B	A
F	No fault	B	F	A

S. Balsamo – Università Ca' Foscari Venezia – SO.8.106

107

Sostituzione di pagina Least-Frequently-Used (LFU)

- Sostuisce la pagina che è **meno intensamente** riferita
- Basato sul **euristica** che una pagina alla quale non si fa riferimento spesso **non è probabile che sia riferita in futuro**
- Potrebbe facilmente selezionare la pagina sbagliata per la sostituzione
 - Una pagina che è stata molto riferita in passato può non essere più riferita di nuovo, ma rimarrà in memoria mentre altre pagine attive più recenti vengono sostituite

S. Balsamo – Università Ca' Foscari Venezia – SO.8.107

108

Sostituzione di pagina Not-Frequently-Used (NFU)

- Sostuisce la pagina che è **non è stata recentemente** riferita
- Usa un contatore associato ad ogni pagina e inizializzato a 0
- Ad ogni interrupt del clock il s.o. fa la scansione dei bit delle pagine di memoria e somma il bit R al contatore, che tiene così traccia del numero di riferimenti
- Si sceglie la pagina con il valore del contatore minimo

S. Balsamo – Università Ca' Foscari Venezia – SO.8.108

Sostituzione di pagina Not-Recently-Used (NRU)

- Approssima la strategia LRU con un **overhead minimo** utilizzando il **bit di riferimento** e il **bit di modifica** per determinare quale pagina non è stata utilizzata di recente e può essere rapidamente sostituita
 - bit di **riferimento** = 0 se la pagina non è stata riferita, 1 altrimenti (bit di accesso)
 - bit di **modifica** = 0 se la pagina non è stata modificata 1 altrimenti
 - due bit hw nella tabella delle pagine
 - cerca una pagina non riferita (approssima LRU)
 - se non la trova cerca un pagina non modificata
- Può essere implementata su macchine che non dispongono di bit di riferimento e / o bit di modifica hardware (simulabile sw)
- Con molti utenti le pagine vengono spesso tutte riferite
 - Periodicamente si possono **azzerare** i bit di riferimento

S. Balsamo – Università Ca' Foscari Venezia – SO.8.109

109

Sostituzione di pagina Not-Used-Recently (NUR)

Tipi di pagine con NUR

Classe	Bit di riferimento	Bit di modifica	Descrizione
Classe 0	0	0	Migliore scelta per la sostituzione
Classe 1	0	1	Sembra non realistico
Classe 2	1	0	
Classe 3	1	1	Peggior scelta per la sostituzione

S. Balsamo – Università Ca' Foscari Venezia – SO.8.110

110

Varianti della strategia FIFO

Second-Chance e Sostituzione a orologio

- Strategia di sostituzione di pagina **Second chance**
 - Esamina il bit di riferimento della **pagina più vecchia**
 - Se è **off (0)** la strategia **seleziona** la pagina per la sostituzione
 - Se è **on (1)** la strategia **azzerà** il bit e **sposta la pagina in coda FIFO** e si considera come un nuovo arrivo
 - Assicura che le **pagine attive** siano quelle con **minor probabilità** di essere **sostituite**
- Strategia di **sostituzione a orologio**
 - Simile a second chance, ma organizza le pagine in **una lista circolare** invece della lista lineare
 - Puntatore che scorre la lista ad ogni page fault; se bit ref =0 si scorre (simula FIFO)

S. Balsamo – Università Ca' Foscari Venezia – SO.8.111

111

Varianti della strategia FIFO: Second-Chance

Operazioni di una strategia **second chance**

- pagine ordinate secondo FIFO
- la **lista delle pagine** se il fault accade al tempo 20, **A** ha bit **R=1** (i numeri sopra alle pagine sono i tempi di caricamento)

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.7.112

112

113

Varianti della strategia FIFO: Sostituzione a orologio

Quando si verifica un page fault, viene analizzata la pagina cui sta puntando la lancetta. L'azione intrapresa dipende dal bit R:
 $R = 0$: Rimuovi la pagina
 $R = 1$: Azzera R e fai avanzare la lancetta

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.7.113

114

Sostituzione delle pagine *Far*

- Idea: **lontananza** delle pagine
- Crea un **grafo di accesso** che rappresenta il modello dei **riferimenti** di un processo
- Grafo:
 - nodi=pagine
 - archi=riferimenti
- Sostituisce la pagina** non referenzialato che è **più lontana nel grafo di accesso** da qualsiasi pagina riferita
- Prestazioni a livelli quasi ottimali
- Non è stata solitamente implementata in sistemi reali
 - La ricerca e gestione del grafo di accesso è complessa e senza supporto hardware

S. Balsamo – Università Ca' Foscari Venezia – SO.8.114

115

Sostituzione delle pagine *Far*

Esempio di grafo di accesso nella strategia di sostituzione *far*

* Denotes a referenced page

Q is chosen for replacement

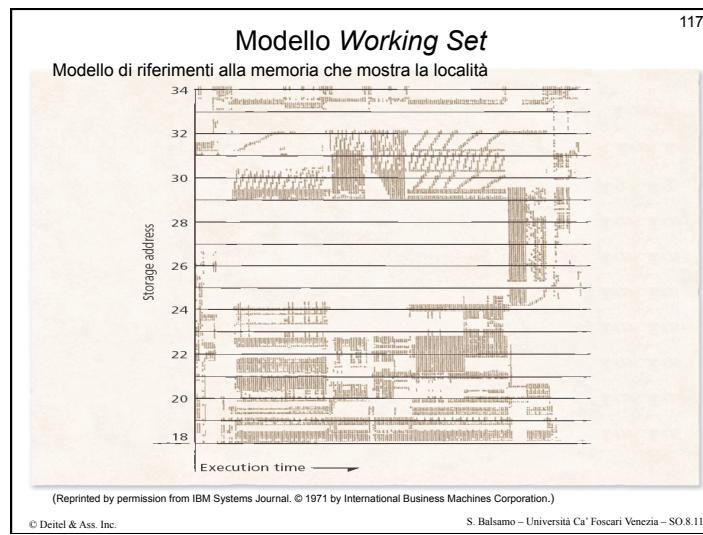
© Deitel & Ass. Inc. S. Balsamo – Università Ca' Foscari Venezia – SO.8.115

116

Modello *Working Set*

- Per eseguire un programma in modo efficiente
 - Il sistema deve mantenere in memoria principale quel **sottoinsieme di pagine favorito** del programma
- Altrimenti
 - Il sistema potrebbe sperimentare un'eccessiva attività di paginazione causando una bassa utilizzazione del processore chiamato **thrashing** quando il programma richiede ripetutamente pagine in memoria secondaria

S. Balsamo – Università Ca' Foscari Venezia – SO.8.116



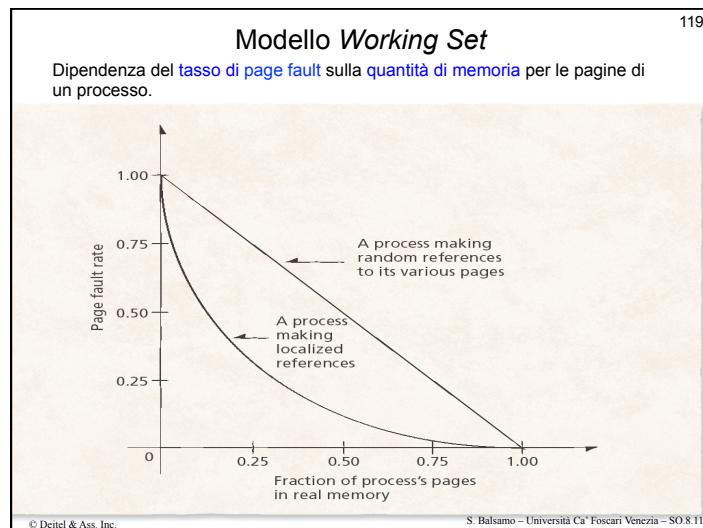
117

Modello Working Set

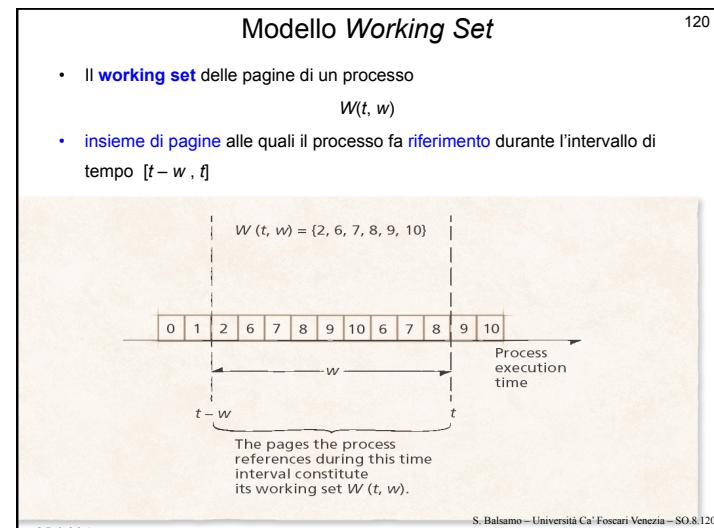
- Poiché i processi mostrano **località**
aumentare il numero di page frame assegnati ad un processo
oltre una certa **soglia**
ha poco o nessun effetto sul tasso di page fault

118

S. Balsamo – Università Ca' Foscari Venezia – SO.8.118



119



121

Modello Working Set

- La dimensione del working set del processo aumenta asintoticamente con la dimensione del programma del processo all'aumentare della finestra del working set

S. Balsamo – Università Ca' Foscari Venezia – SO.8.121

122

Modello Working Set

Dimensione del working set in funzione della dimensione della finestra

The graph plots the dimension of the working set $W(t,w)$ against the dimension of the window w . A horizontal dashed line represents the size of the program. The curve $W(t,w)$ starts at the origin and increases monotonically, approaching the program size as w increases.

- working set: insieme di pagine usate dai w più recenti riferimenti
- $W(t,w)$ dimensione del working set al tempo t

© Deitel & Ass. Inc. S. Balsamo – Università Ca' Foscari Venezia – SO.8.122

123

Modello Working Set

- Allo spostarsi tra working sets di un processo, il sistema mantiene temporaneamente in memoria le pagine di che non sono più nel working set corrente del processo
 - L'obiettivo della gestione della memoria del working set è di ridurre questa cattiva allocazione
 - Tenendo traccia del working set al momento di un page fault si elimina una pagina esterna al w.s.
 - Approssimazioni del w.s. per maggior efficienza
 - Uso del tempo di esecuzione anziché del numero di riferimenti
 - Tempo virtuale attuale (corrente): tempo di CPU usata effettivamente dal suo avvio.
 - Il w.s. è l'insieme di pagine referenziate negli ultimi τ sec. di tempo virtuale

S. Balsamo – Università Ca' Foscari Venezia – SO.8.123

124

Algoritmo di sostituzione di pagina con Working Set

The diagram shows a table of pages with columns for address, time of last use, reference bit R, and status. An arrow points to the current virtual time (2204). The table includes rows for 2084 (R=1), 2003 (R=1), 1980 (R=1), 1213 (R=0), 2014 (R=1), 2020 (R=1), 2032 (R=1), and 1620 (R=0).

Informazioni su una pagina

Tempo di ultimo utilizzo

Pagina referenziata durante questo ciclo

Pagina non referenziata durante questo ciclo

Bit R (Referenziato)

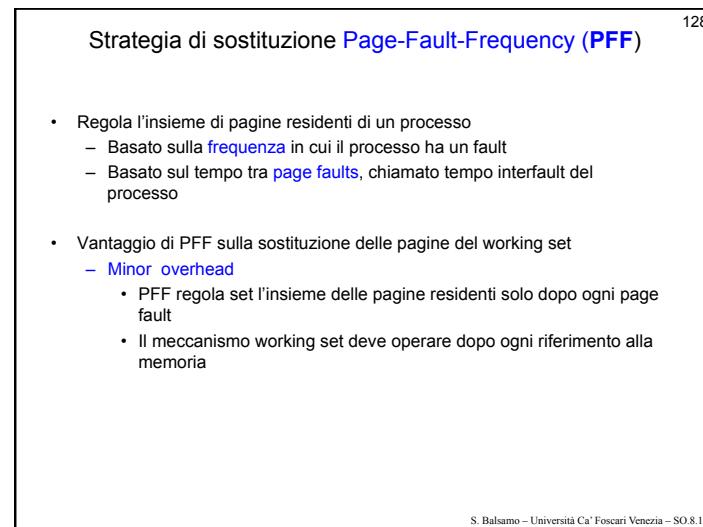
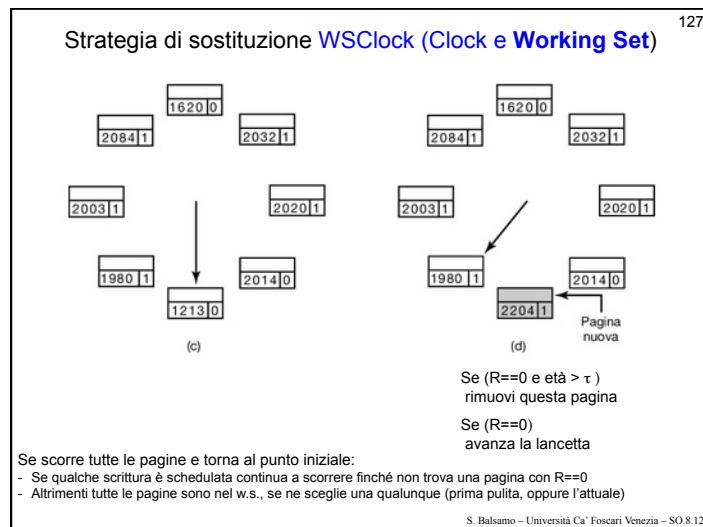
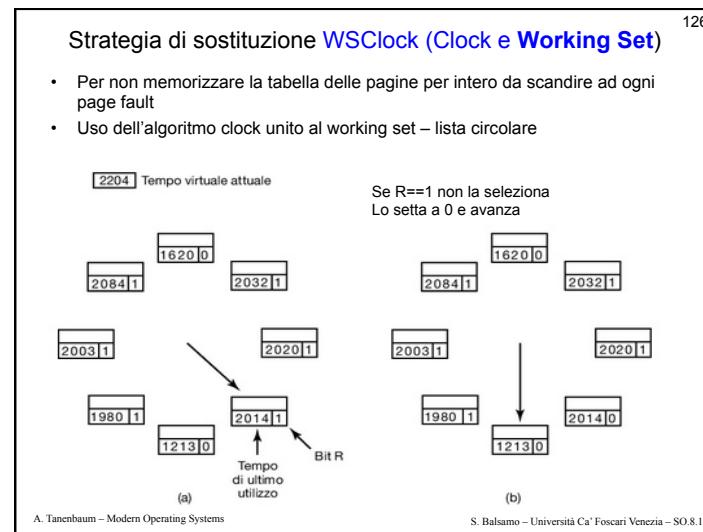
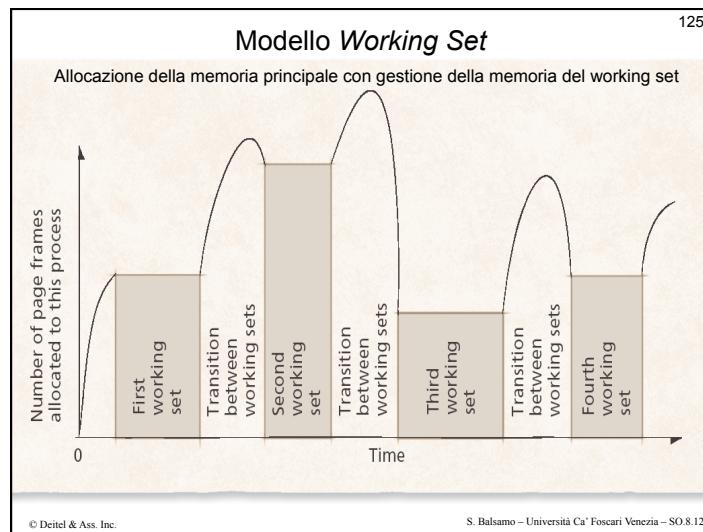
Scandisci tutte la pagine esaminando il bit R: rimuovi questa pagina se ($R == 1$) imposta il tempo di ultimo utilizzo al tempo virtuale attuale

Se ($R == 0$ e età > τ) rimuovi questa pagina

Se ($R == 0$ e età ≤ τ) ricorda il tempo più piccolo

Tabella delle pagine

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.7.124



129

Rilascio delle pagine

- Le pagine inattive possono rimanere in memoria principale a lungo finché la strategia di gestione rileva che il processo non ne ha più bisogno
 - Un modo per risolvere il problema
 - Il **processo chiede una sostituzione volontaria** di pagina per liberare un page frame del quale sa di non aver più bisogno
 - Elimina il ritardo causato dal graduale rilascio delle pagine del working set del processo
 - La miglior opportunità è nel supporto del **compilatore** e del sistema operativo

S. Balsamo – Università Ca' Foscari Venezia – SO.8.129

130

Sintesi e confronto fra le strategie di sostituzione

Algoritmo	Commento
Ottimale	Non implementabile, ma utile come indice di confronto e valutazione
NRU (Not Recently Used)	Approssimazione molto rossa dell'LRU
FIFO (First-In, First Out)	Potrebbe eliminare pagine importanti
Seconda chance	Deciso miglioramento rispetto al FIFO
Clock	Realistico
LRU (Last Recently Used)	Eccellente, ma difficile da implementare con precisione
NFU (Not Frequently Used)	Approssimazione abbastanza rossa dell'LRU
Aging	Algoritmo efficiente che approssima bene l'LRU
Working set	Piuttosto dispendioso da implementare
WSClock	Algoritmo efficiente e buono

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.7.130

131

Dimensione della pagina

- Alcuni sistemi migliorano le prestazioni e l'utilizzo della memoria, fornendo diverse dimensioni di pagina
 - dimensione **piccola** di pagina
 - Minor **frammentazione interna**
 - Può ridurre la quantità di **memoria** necessaria per contenere working set di un processo
 - Maggiore **memoria** disponibile per altri processi – minor spreco di memoria
 - **Tabella** delle pagine più grande

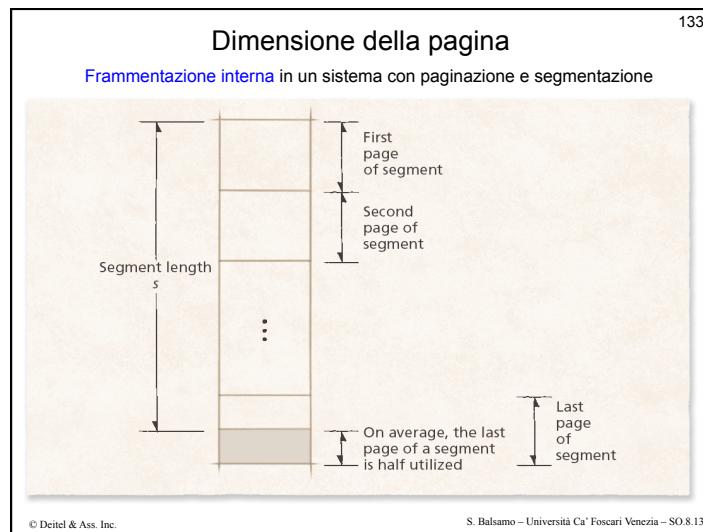
S. Balsamo – Università Ca' Foscari Venezia – SO.8.131

132

Dimensione della pagina

- dimensione **grande** di pagina
 - Riduce la **memoria** sprecata dalla frammentazione della tabella
 - Abilita ogni riga della TLB a mappare una regione della memoria più grande, migliorando le **prestazioni**
 - **Riduce il numero di operazioni di I/O** del sistema per caricare il working set di un processo in memoria
- dimensione **multipla** di pagina
 - Possibile frammentazione esterna

S. Balsamo – Università Ca' Foscari Venezia – SO.8.132



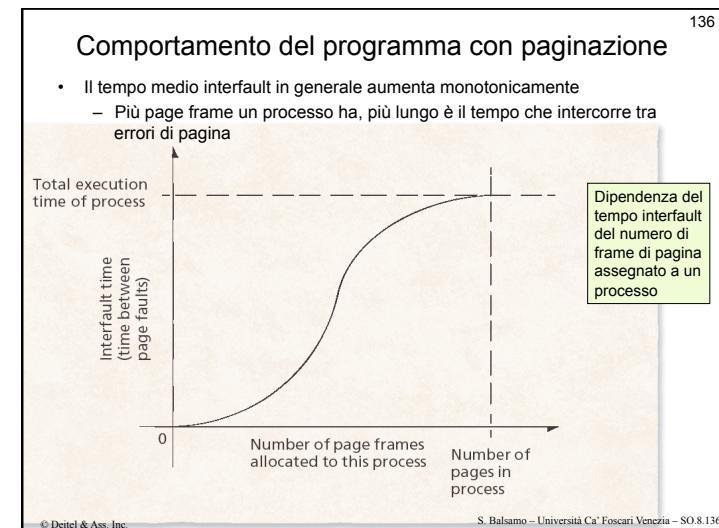
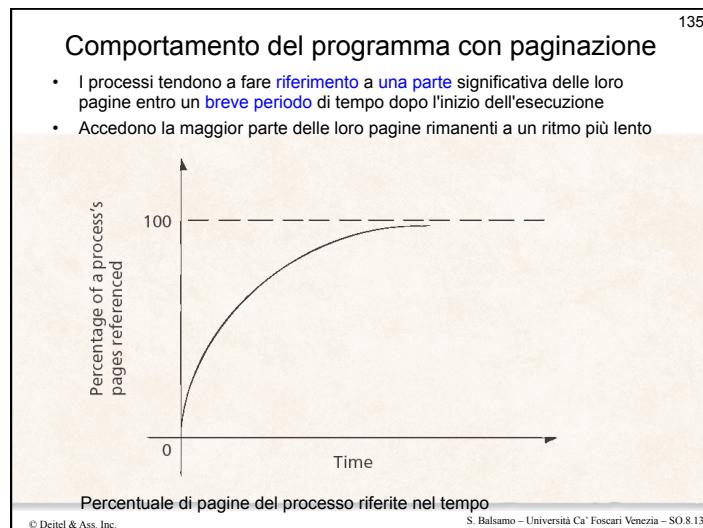
Dimensione della pagina

Dimensioni della pagina in varie architetture di processori

Manufacturer	Model	Page Size	Real address size
Honeywell	Multics	1KB	36 bits
IBM	370/168	4KB	32 bits
DEC	PDP-10 and PDP-20	512 bytes	36 bits
DEC	VAX 8800	512 bytes	32 bits
Intel	80386	4KB	32 bits
Intel / AMD	Pentium 4 / Athlon XP	4KB or 4MB	32- or 36 bits
Sun	UltraSparc II	8KB, 64KB, 512KB, 4MB	44 bits
AMD	Opteron / Athlon 64	4KB, 2MB and 4MB	32, 40, or 52 bits
Intel-HP	Itanium, Itanium 2	4KB, 8KB, 16KB, 64KB, 256KB, 1MB, 4MB, 16MB, 64MB, 256MB	Between 32 and 63 bits
IBM	PowerPC 970	4KB, 128KB, 256KB, 512KB, 1MB, 2MB, 4MB, 8MB, 16MB, 32MB, 64MB, 128MB, 256MB	32 or 64 bits

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.8.134



Strategie di sostituzione Globali vs. Locali

137

- Implementazione di un sistema di memoria virtuale con paginazione
 - Strategie **Globali** di sostituzione di pagina: **applicate a tutti i processi come unità**
 - Tendono a ignorare le caratteristiche individuali di comportamento del processo
 - Strategia globale LRU (**gLRU**)
 - Sostituisce la pagina meno utilizzata di recente in tutto il sistema
 - **SEQ** strategia di sostituzione di pagina globale (sequenza)
 - Usa la strategia LRU per sostituire le pagine fino a quando non viene rilevato sequenza di errori di pagina a pagine contigue, a quel punto utilizza la strategia most-recently-used (**MRU**)
- Strategie **Locali**: **considerare ogni processo individualmente**
 - Consente di regolare il sistema di allocazione della memoria in base alla importanza relativa di ciascun processo per migliorare le prestazioni

S. Balsamo – Università Ca' Foscari Venezia – SO.8.137

Confronto fra le strategie di sostituzione globali e locali

138

Età
A0
A1
A2
A3
A4
A5
B0
B1
B2
B3
B4
B5
B6
C1
C2
C3

(a) Configurazione originale

Età
A0
A1
A2
A3
A4
A5
B0
B1
B2
B3
B4
B5
B6
C1
C2
C3

(b) Sostituzione locale di pagine

Età
A0
A1
A2
A3
A4
A5
B0
B1
B2
B3
B4
B5
B6
C1
C2
C3

(c) Sostituzione globale di pagine

A. Tanenbaum – Modern Operating Systems

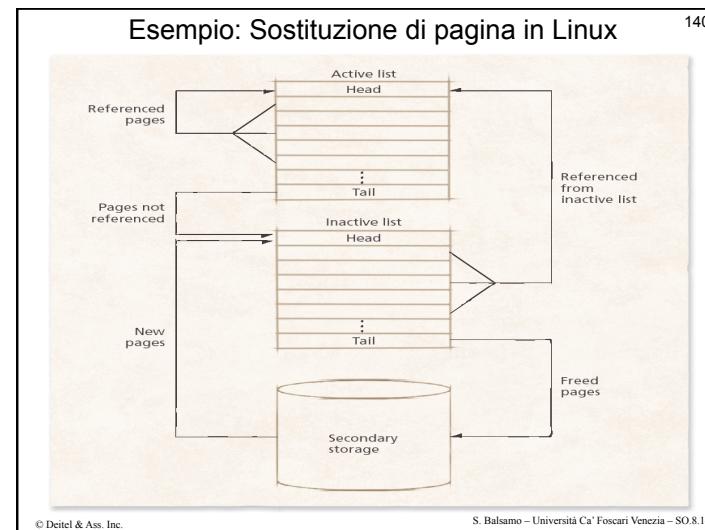
S. Balsamo – Università Ca' Foscari Venezia – SO.7.138

Esempio: Sostituzione di pagina in Linux

139

- Linux utilizza una variante dell'algoritmo orologio per approssimare una strategia di sostituzione di pagina LRU
- Il gestore di memoria utilizza due liste collegate
 - **lista attiva**
 - Contiene le pagine attive
 - Le pagine usate più di recente sono in cima alla lista attiva
 - **lista inattiva**
 - Contiene le pagine inattive
 - Le pagine meno usate di recente sono in fondo alla lista inattiva
 - Vengono sostituite solo le pagine della lista inattiva

S. Balsamo – Università Ca' Foscari Venezia – SO.8.139

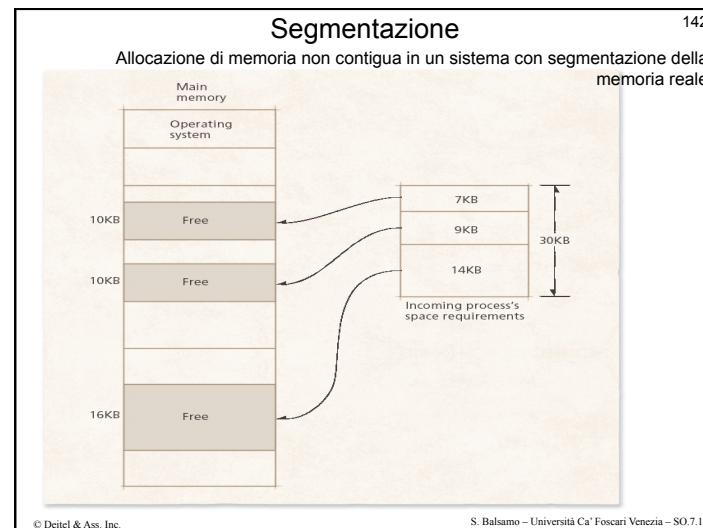


141

Segmentazione

- **Segmento**
 - Contiene porzioni significative del programma (es. procedure, array, stack)
 - Composto da **posizioni contigue**
 - I segmenti **non** devono necessariamente essere **della stessa dimensione** né devono essere **adiacenti** in memoria principale
 - Il segmento è un concetto **logico e non fisico** (come per la pagina)
- Un processo può essere in esecuzione mentre le sue istruzioni correnti e i dati riferiti sono in segmenti di memoria principale
- Se il processo fa riferimento a dati in memoria secondaria il segmento corrispondente va caricato

S. Balsamo – Università Ca' Foscari Venezia – SO.7.141



143

Segmentazione

- Un processo fa riferimento a un **indirizzo di memoria virtuale** $v = (s, d)$
 - s è il numero di **segmento** in memoria virtuale
 - d è lo **spostamento** all'interno del segmento s in cui si trova l'elemento indirizzato

Formato dell'indirizzo virtuale in un sistema di segmentazione pura

S. Balsamo – Università Ca' Foscari Venezia – SO.7.143

144

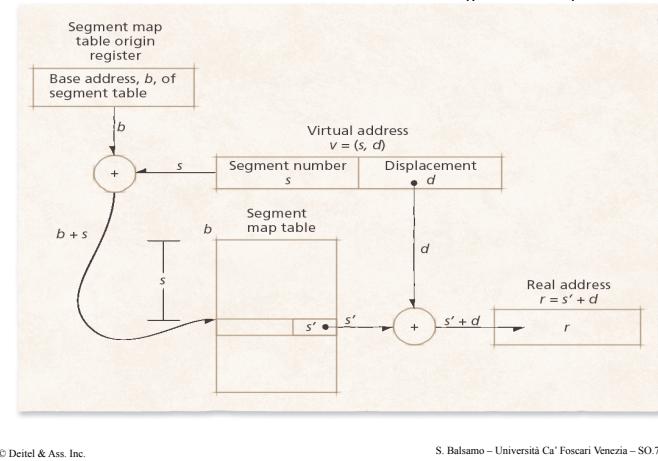
Segmentazione traduzione dell'indirizzo con **mapping diretto**

- Il processo fa riferimento ad un indirizzo di memoria virtuale $v = (s, d)$
 - DAT aggiunge l'**indirizzo di base** della **tavella dei segmenti** del processo, b , al numero di segmento riferito, s
 - $b + s$ costituisce l'**indirizzo** di memoria principale della riga nella tabella dei segmenti per il segmento s
 - La riga contiene l'indirizzo iniziale del segmento in memoria, s'
 - Il sistema aggiunge s' allo spostamento, d , per formare l'indirizzo reale, r

S. Balsamo – Università Ca' Foscari Venezia – SO.7.144

Segmentazione – Traduzione dell'indirizzo con mapping diretto 145

Traduzione dell'indirizzo virtuale in un sistema con segmentazione puro



© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.145

146

Segmentazione traduzione dell'indirizzo con mapping diretto

- Riga della tabella della mappa dei segmenti
 - Indica che quel segmento s inizia all'indirizzo di memoria reale s'
 - Contiene un bit di residenza, r , che indica se il segmento è in memoria
 - Se $r = 1$, s' memorizza l'indirizzo di base del segmento
 - Se $r = 0$, a memorizza la posizione del segmento in memoria secondaria
 - Inoltre contiene un campo lunghezza, l , che indica la dimensione del segmento
 - Può essere usato per evitare che un processo riferisca indirizzi fuori dal segmento (eccezione di overflow del segmento)
 - Bit di protezione per controllare se il tipo di operazione è ammessa (eccezione di protezione del segmento)
 - (es. lettura, scrittura,...)

S. Balsamo – Università Ca' Foscari Venezia – SO.7.146

Segmentazione – traduzione dell'indirizzo con mapping diretto 147

Riga della tabella dei segmenti

Segment resident bit	Secondary storage address (if not in main memory)	Segment length	Protection bits	Base address of segment (if in main memory)
r	a	l		s'

$r = 0$ if segment is not in main memory
 $r = 1$ if segment is in main memory

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.147

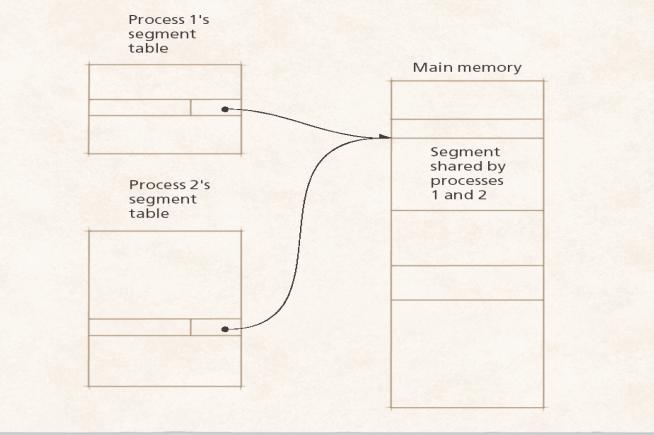
148

Condivisione in un sistema con segmentazione

- La condivisione segmenti può causare meno overhead della condivisione nel sistema con paginazione pura e con mapping diretto
 - Potenzialmente pochi elementi della tabella delle mappa devono essere condivise
 - Controllo delle interferenze e meccanismi di protezione

S. Balsamo – Università Ca' Foscari Venezia – SO.7.148

Condivisione in un sistema con segmentazione



© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.149

149

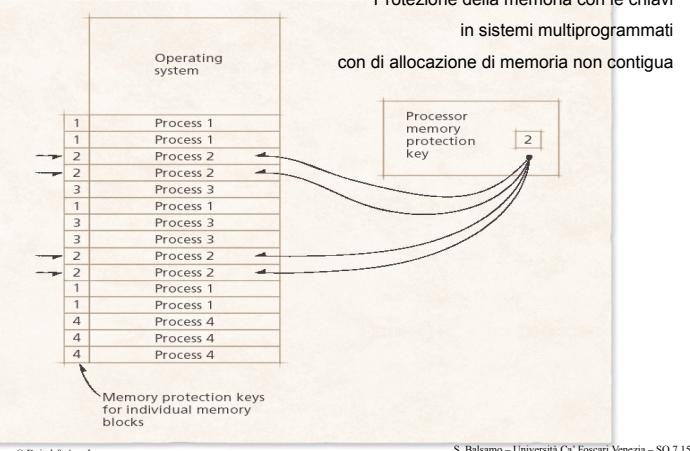
Protezione e Controllo degli Accessi in Sistema con Segmentazione

- Uno schema per implementare la **protezione della memoria** nei sistemi di segmentazione è **chiavi di protezione della memoria**
- Chiave di protezione
– **valore associato ad un processo**
- Se la chiave di protezione per il processore e il blocco richiesto sono gli stessi, il processo può accedere al segmento
- Chiavi manipolabili solo da **istruzioni privilegiate**

S. Balsamo – Università Ca' Foscari Venezia – SO.7.150

150

Protezione e Controllo degli Accessi in Sistema con Segmentazione



© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.151

151

Protezione e Controllo degli Accessi in Sistema con Segmentazione

- Uno schema più comune è quello di utilizzare i **bit di protezione** che specificano se un processo è autorizzato a compiere operazioni e.g. se può leggere, scrivere, eseguire codice o aggiungere ad un segmento
- diritti di accesso** ai segmenti

Type of access Abbreviation Description

Type of access	Abbreviation	Description
Read	R	This segment may be read.
Write	W	This segment may be modified.
Execute	E	This segment may be executed.
Append	A	This segment may have information added to its end.

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.152

152

Protezione e Controllo degli Accessi in Sistema con Segmentazione ¹⁵³					
Combinazione di diritti di accesso in lettura, scrittura ed esecuzione per definire modalità di controllo di accesso utili					
Mode	Read	Write	Execute	Description	Application
Mode 0	No	No	No	No access permitted	Security.
Mode 1	No	No	Yes	Execute only	A segment made available to processes that cannot modify it or copy it, but that can run it.
Mode 2	No	Yes	No	Write only	These possibilities are not useful, because granting write access without read access is impractical.
Mode 3	No	Yes	Yes	Write/execute but cannot be read	{ Write/execute but cannot be read
Mode 4	Yes	No	No	Read only	Information retrieval.
Mode 5	Yes	No	Yes	Read/execute	A program can be copied or executed but cannot be modified.
Mode 6	Yes	Yes	No	Read/write but no execution	Protects data from an erroneous attempt to execute it.
Mode 7	Yes	Yes	Yes	Unrestricted access	This access is granted to trusted users.

© Deitel & Ass. Inc. S. Balsamo – Università Ca' Foscari Venezia – SO.7.153

Protezione e Controllo degli Accessi in Sistema con Segmentazione ¹⁵⁵					
Riga della tabella dei segmenti con i bit di protezione					
Segment resident bit	Secondary storage address (if not in main memory)	Segment length	Protection bits	Base address of segment (if segment is in memory)	
r	a	l	R W E A	s'	
Protection bits: (1=yes, 0=no)					
R–Read access W–Write access E–Execute access A–Append access					

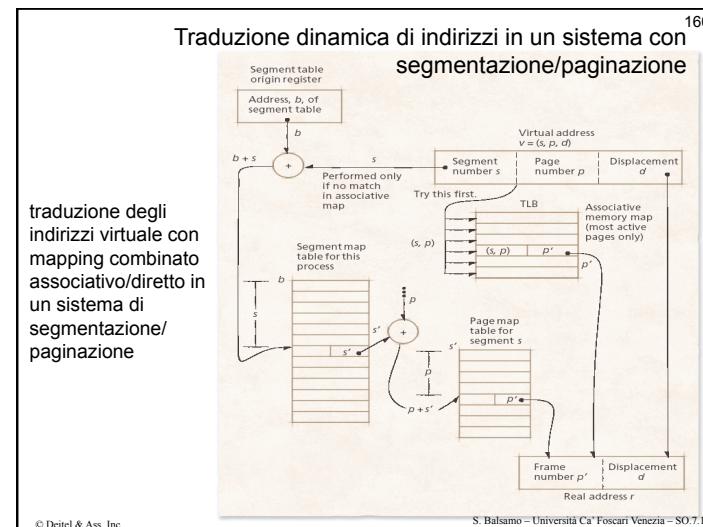
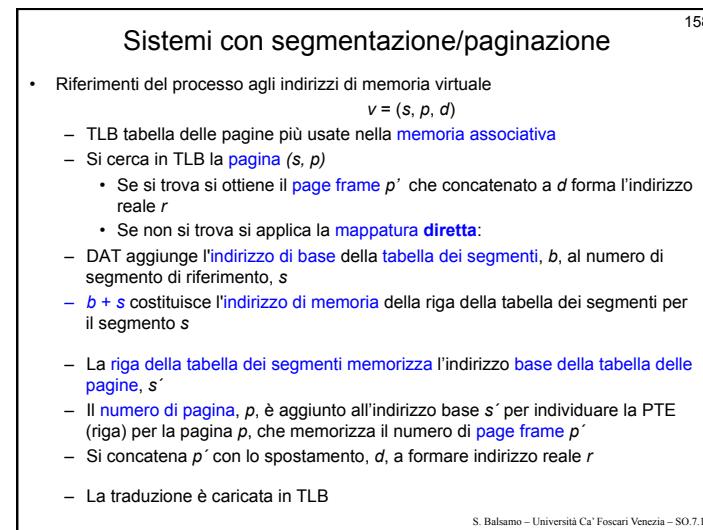
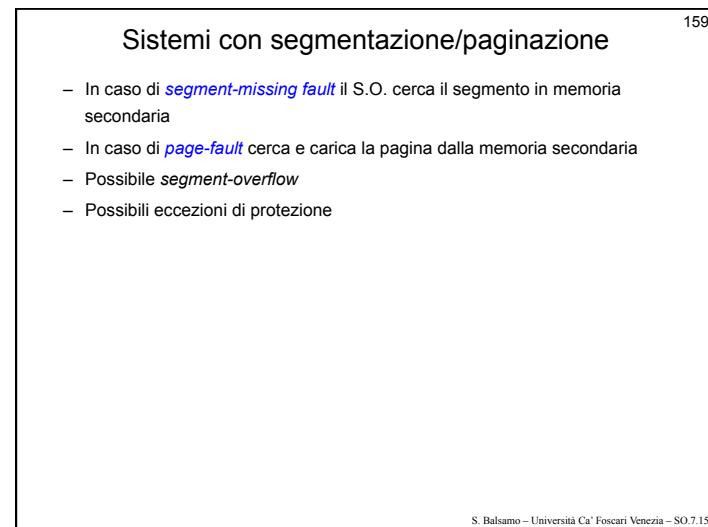
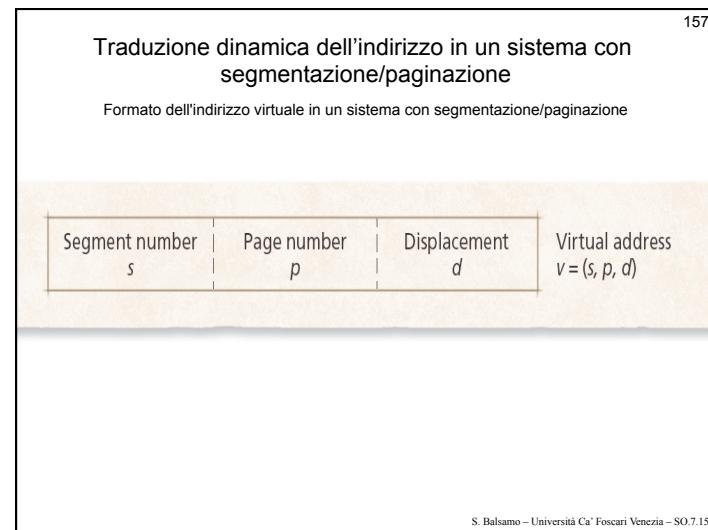
S. Balsamo – Università Ca' Foscari Venezia – SO.7.155

Protezione e Controllo degli Accessi in Sistema con Segmentazione ¹⁵⁴					
<ul style="list-style-type: none"> I bit di protezione vengono aggiunti alle righe della tabella dei segmenti e controllati quando un processo fa riferimento a un indirizzo <ul style="list-style-type: none"> Se il segmento non è in memoria, viene generato un fallimento di accesso al segmento (<i>missing-segment fault</i>) Se $d > l$, viene generata un'eccezione di overflow del segmento (<i>overflow-segment exception</i>) Se non è consentita l'operazione (es. <i>read, write, execute, append</i>), viene generata un'eccezione di protezione di segmento (<i>segment-protection exception</i>) 					

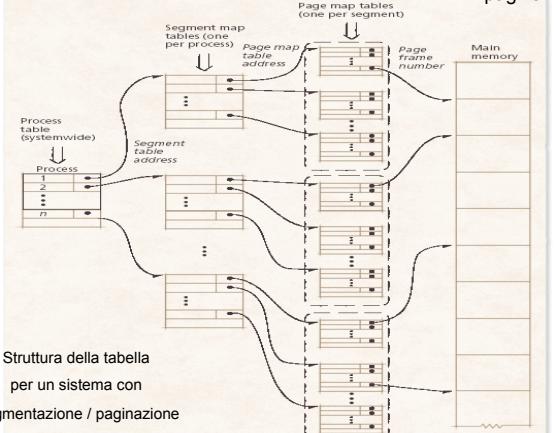
S. Balsamo – Università Ca' Foscari Venezia – SO.7.154

Sistemi con Segmentazione/Paginazione ¹⁵⁶					
<ul style="list-style-type: none"> Per sfruttare i vantaggi delle due tecniche I segmenti occupano una o più pagine Tutte le pagine del segmento non devono necessariamente essere in memoria principale contemporaneamente Le pagine contigue nella memoria virtuale non devono essere contigui nella memoria principale Indirizzo di memoria virtuale è implementato come una tripla ordinata $v = (s, p, d)$ <ul style="list-style-type: none"> s è il numero del segmento p è il numero di pagina all'interno del segmento d è lo spostamento all'interno della pagina in cui trova l'elemento desiderato 					

S. Balsamo – Università Ca' Foscari Venezia – SO.7.156



Traduzione dinamica di indirizzi in un sistema con segmentazione/paginazione¹⁶¹



Struttura della tabella per un sistema con segmentazione / paginazione

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.161

Condivisione e protezione in un sistema con segmentazione/paginazione

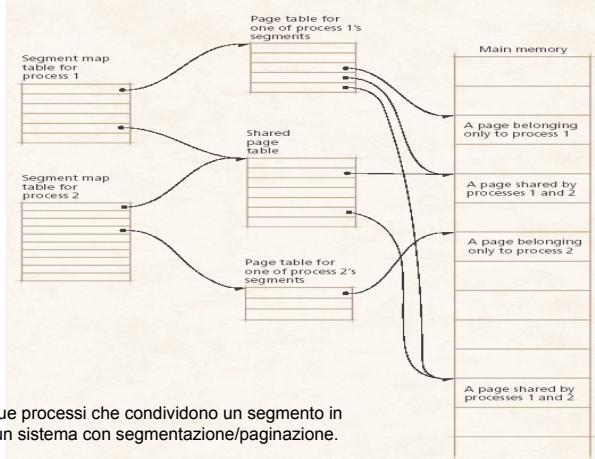
162

- Sistemi di paginazione e segmentazione**
 - Due processi condividono la stessa memoria quando ogni processo ha una riga della tabella di mappa dei segmenti che punta alla stessa tabella delle pagine
 - La **condivisione** è facilitata dalla **paginazione**
 - Controllo degli accessi come da **segmentazione**
- La condivisione richiede una gestione attenta da parte del sistema operativo
 - Diversi processi possono condividere le pagine e in caso di modifica devono essere aggiornate le tabelle
 - Uso di liste di PTE che mappano una pagina condivisa per facilitare l'aggiornamento in caso di modifica

S. Balsamo – Università Ca' Foscari Venezia – SO.7.162

Condivisione e protezione in un sistema con segmentazione/paginazione

163



Due processi che condividono un segmento in un sistema con segmentazione/paginazione.

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.7.163

Esempio: architettura di memoria virtuale IA-32 Intel

164

- Architettura IA-32 Intel supporta sia la **segmentazione pura** e **segmentazione/paginazione** della memoria virtuale
- Spazio di indirizzamento logico
 - Insieme di indirizzi contenuti in ogni segmento
- Segmenti
 - Situato in qualsiasi posizione disponibile nello spazio di indirizzamento lineare del sistema
- traduzione degli indirizzi del segmento
 - Eseguita con **mapping diretto** che utilizza registri del processore ad alta velocità per memorizzare la tabella dei segmenti all'inizio registrata nel descrittore di tabella globale (Tabella Descrittore Globale GDT) o locale (Tabella Descrittore Locale LDT)
- Paginazione
 - Supporta tabelle delle pagine più livelli e formati pagina multiple

S. Balsamo – Università Ca' Foscari Venezia – SO.7.164