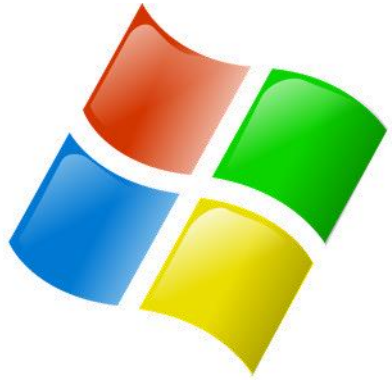
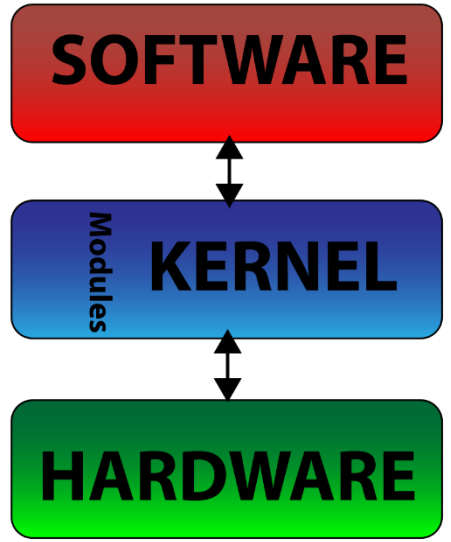
UNIVERSITÀ DEGLI STUDI DELLA BASILICATA

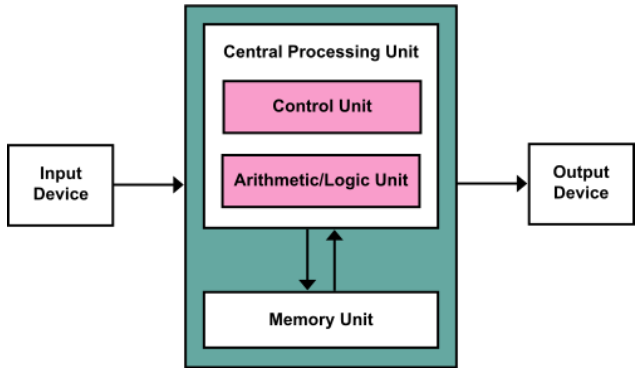


Corso di Sistemi Operativi

Allocazione della memoria



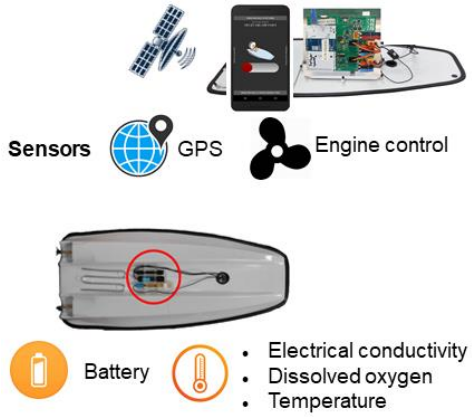
Docente:
Domenico Daniele Bloisi

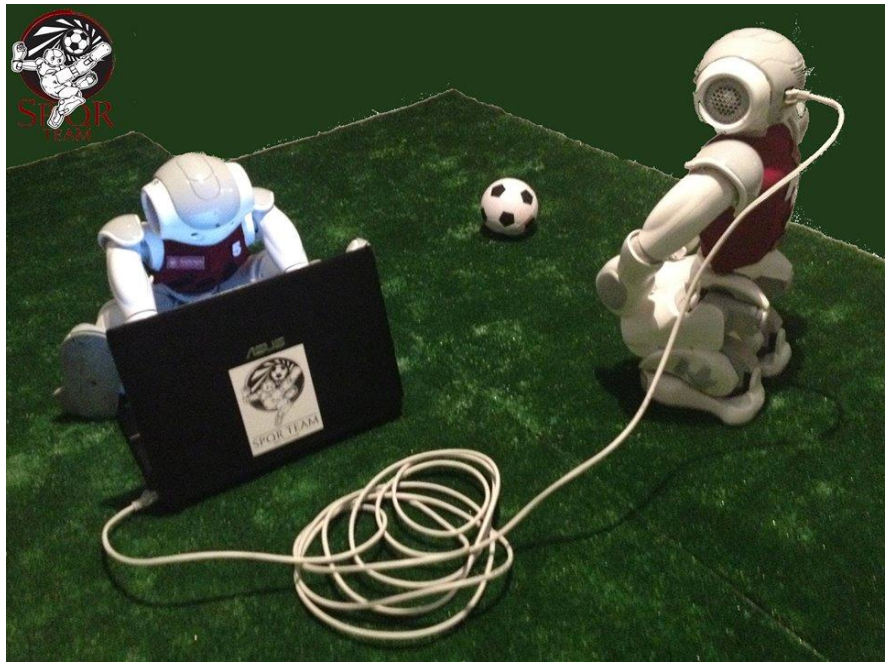


Domenico Daniele Bloisi

- Ricercatore RTD B
  Dipartimento di Matematica, Informatica ed Economia
  Università degli studi della Basilicata
  http://web.unibas.it/bloisi

- SPQR Robot Soccer Team
  Dipartimento di Informatica, Automatica e Gestionale Università degli studi di Roma “La Sapienza”
  http://spqr.diag.uniroma1.it





Informazioni sul corso

- Home page del corso:
  http://web.unibas.it/bloisi/corsi/sistemi-operativi.html
- Docente: Domenico Daniele Bloisi
- Periodo: I semestre ottobre 2021 – febbraio 2022

  - Lunedì dalle 15:00 alle 17:00 (Aula A18)
  - Martedì dalle 12:30 alle 14:00 (Aula 1)

Ricevimento

- Durante il periodo delle lezioni:
  Martedì dalle 10:00 alle 11:30 → Edificio 3D, II piano, stanza 15
  Si invitano gli studenti a controllare regolarmente la bacheca degli avvisi per eventuali variazioni
- Al di fuori del periodo delle lezioni:
  da concordare con il docente tramite email

Per prenotare un appuntamento inviare una email a
domenico.bloisi@unibas.it

Programma – Sistemi Operativi

- Introduzione ai sistemi operativi
- Gestione dei processi
- Sincronizzazione dei processi
- Gestione della memoria centrale
- Gestione della memoria di massa
- File system
- Sicurezza e protezione

Allocazione dei frame

Problema dell’allocazione:

Quale criterio si deve utilizzare per assegnare la memoria libera ai diversi processi che devono essere eseguiti?

Vincoli per l’allocazione dei frame

1. Non si possono assegnare più frame di quanti siano disponibili

2. È necessario assegnare almeno un numero minimo di frame

↓

prestazioni

Vincoli per l’allocazione dei frame

- il numero minimo di frame per ciascun processo è stabilito dell’architettura del sistema
- il numero massimo di frame per ciascun processo è definito dalla quantità di memoria fisica disponibile

Algoritmi di allocazione dei frame

- allocazione uniforme
- allocazione proporzionale
- allocazione locale
- allocazione globale

Allocazione uniforme

allocazione uniforme

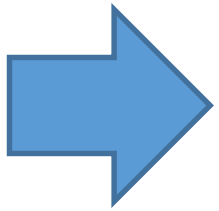
Assumendo di avere *m* frame liberi e *n* processi da servire, si assegnano *m*/*n* frame a ogni processo

Allocazione proporzionale

allocazione proporzionale

La memoria disponibile si assegna a ciascun processo secondo la propria dimensione

- Si assuma di avere *m* frame liberi
- Sia s_i la dimensione di memoria richiesta dal processo p_i
- Si definisce $S = \sum s_i$ e si assegna a p_i una quantità di memoria pari a $s_i / S \times m$



Allocazione proporzionale

allocazione proporzionale

È possibile utilizzare uno schema di allocazione proporzionale in cui il rapporto dei frame non dipende dalle dimensioni relative dei processi, bensì dalle priorità degli stessi (oppure da una combinazione di dimensioni e priorità)

Allocazione locale

allocazione locale

L’allocazione locale prevede che a un processo venga allocata una certa quantità di memoria e che si possa scegliere un frame da allocare solo all’interno di quell’insieme dei frame.

L’allocazione locale utilizza la sostituzione locale, nella quale il numero di blocchi di memoria associati ad un processo non cambia.

La sostituzione locale può penalizzare un processo, non rendendo disponibili pagine di memoria meno usate

Allocazione globale

allocazione globale

È la politica di allocazione più usata.

Viene implementata usando la sostituzione globale: un processo può scegliere un frame per la sostituzione dall'insieme di tutti frame.

Pro: un processo ad alta priorità può aumentare il proprio livello di allocazione dei frame a discapito dei processi a bassa priorità

Contro: Il tasso di page fault di ogni processo non è controllabile poiché dipende anche dal comportamento degli altri processi.

Recupero

La strategia per implementare una politica globale di sostituzione delle pagine è quella di *garantire che ci sia sempre sufficiente memoria libera per soddisfare nuove richieste*

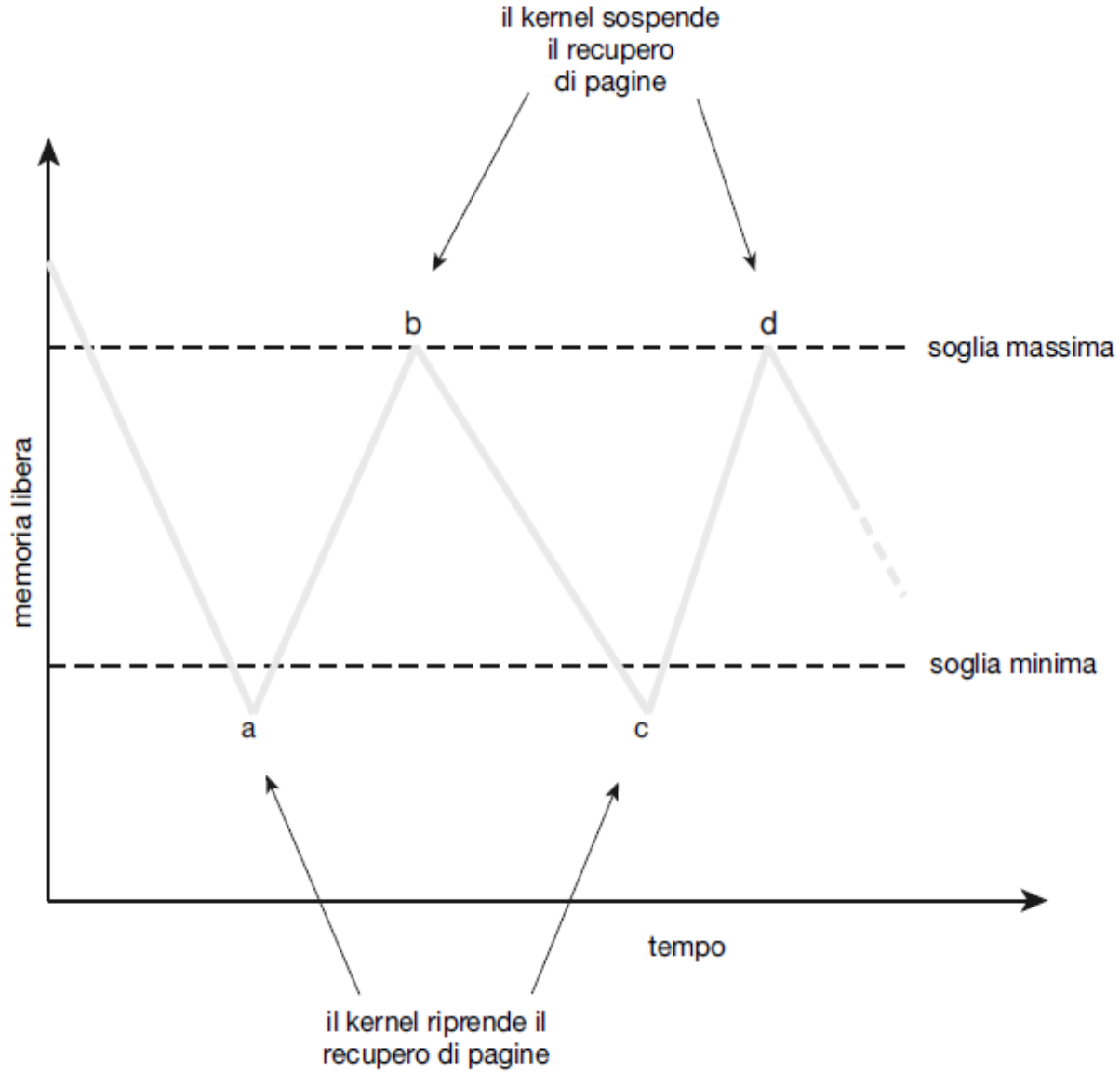


Figura 10.18 Recupero di pagine.

Sistemi con accesso non uniforme in memoria NUMA

Quando un processo incorre in un page fault in un sistema NUMA (Non Uniform Memory Access), se il sistema operativo è conscio dell'architettura NUMA, allora assegna a quel processo un frame il più possibile vicino alla CPU su cui è in esecuzione.

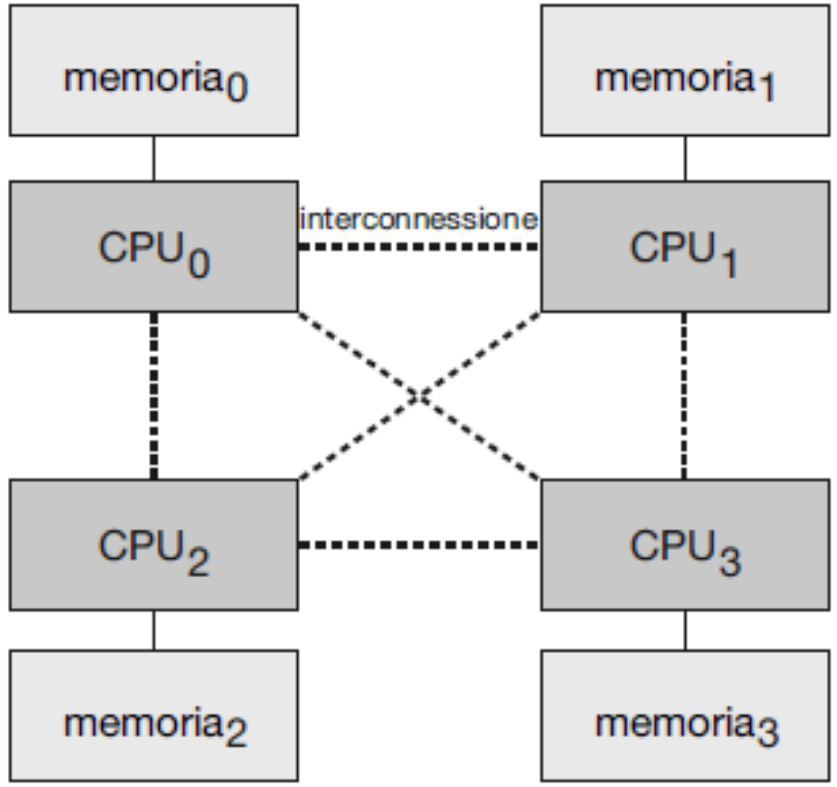


Figura 10.19 Architettura multiprocesso NUMA.

Trashing

Il **thrashing** si verifica quando un sistema spende più tempo per la paginazione rispetto al tempo destinato all'esecuzione.

1. Immaginiamo che un processo non disponga di un numero sufficiente di frame

2. Il processo incorrerà in un page fault e bisognerà sostituire una pagina del processo

3. Se tutte le pagine sono attive, si dovrà scegliere una pagina che molto probabilmente sarà di nuovo necessaria. Pertanto, si verificherà presto un nuovo page fault e si ripeterà il punto 2

Cause del trashing

Il thrashing causa notevoli problemi di prestazioni.

La multiprogrammazione può portare a problemi di trashing → se si supera una certa soglia di multiprogrammazione, l'attività di paginazione degenera e fa crollare l'utilizzo della CPU

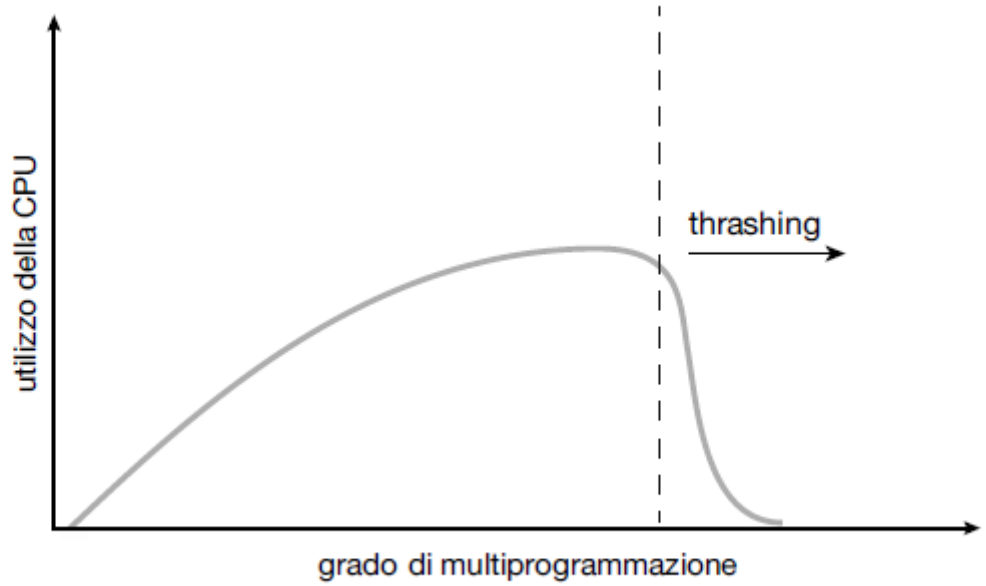


Figura 10.20 Thrashing.

Soluzione al trashing

La soluzione al trashing consiste nel fornire a un processo tutti i frame di cui necessita.

Ma come possiamo farlo?

Località

Una **località** è un set di pagine usate attivamente insieme

La Figura 10.21 illustra il concetto di località e come la località di un processo cambia nel tempo

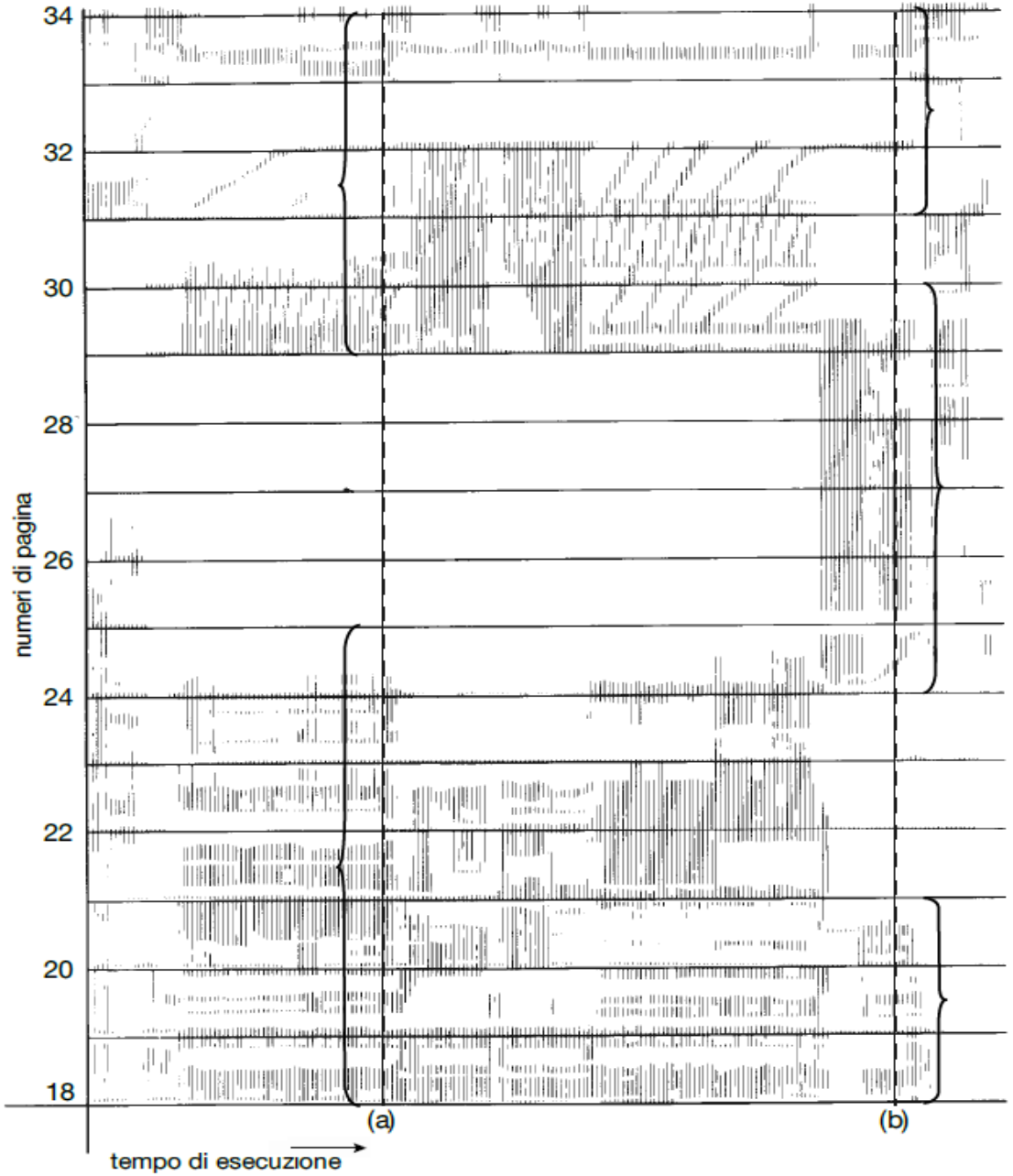


Figura 10.21 Località dei riferimenti alla memoria.

Modello del working set

L'insieme di pagine nei più recenti Δ riferimenti è il **working set** → l'*insieme di pagine utilizzate da un processo in un dato istante*.

Se una pagina è in uso attivo si trova nel working set; se non è più usata esce dal working set Δ unità di tempo dopo il suo ultimo riferimento. Quindi, il working set non è altro che un'approssimazione della località del programma.

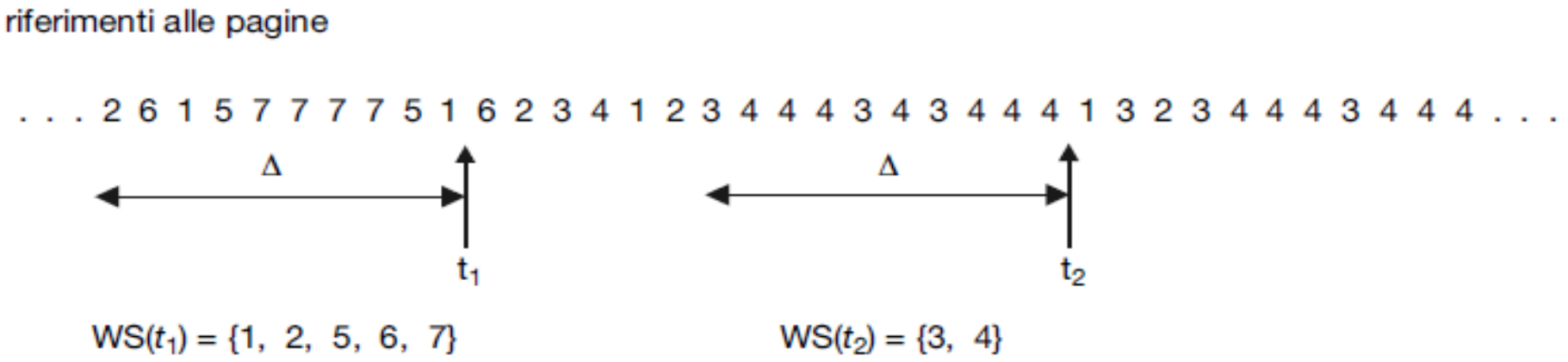


Figura 10.22 Modello del working set.

Frequenza dei page fault

Si può fissare un *limite inferiore* e un *limite superiore* per la frequenza desiderata dei page fault. Se la frequenza effettiva dei page fault per un processo oltrepassa il limite superiore, occorre allocare a quel processo un altro frame; se la frequenza scende sotto il limite inferiore, si sottrae un frame a quel processo → prevenire il thrashing

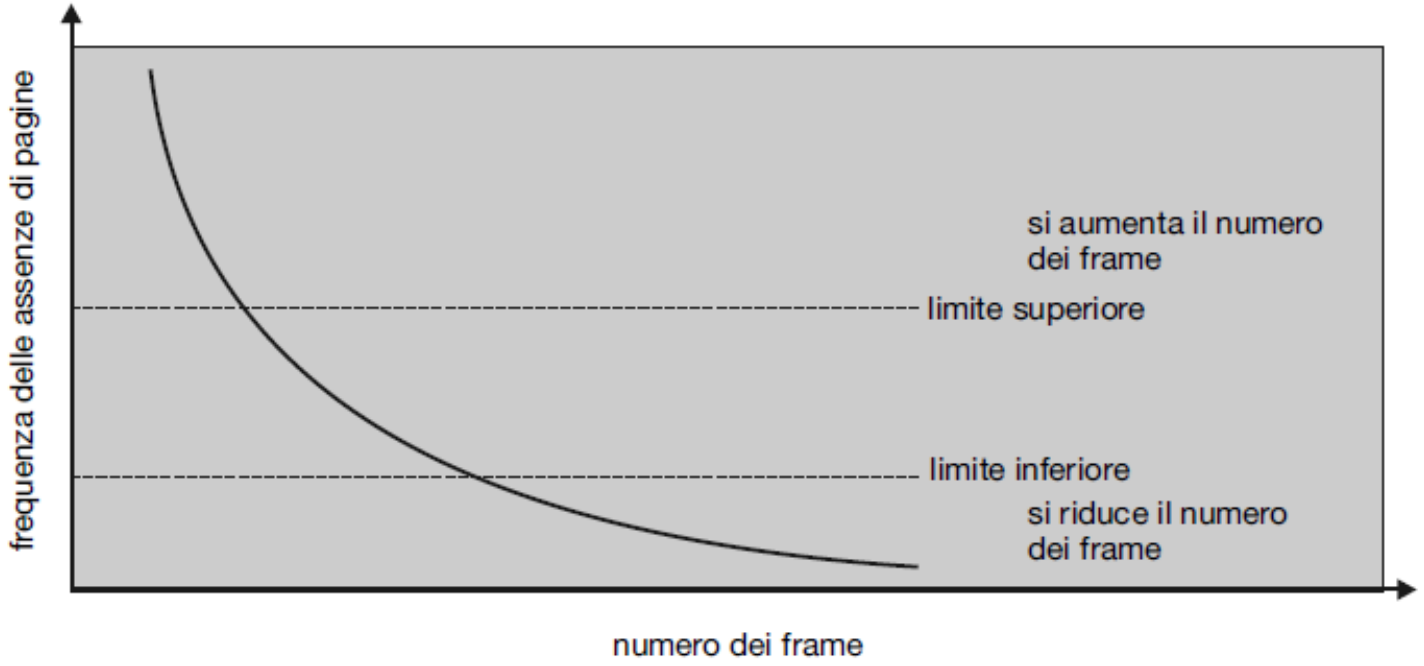


Figura 10.23 Frequenza dei page fault.

Compressione della memoria

La **compressione della memoria** è una tecnica di gestione della memoria che consiste nel comprimere un certo numero di pagine in una singola pagina

La **memoria compressa** è un'alternativa alla paginazione e viene utilizzata su sistemi mobili che non supportano la paginazione

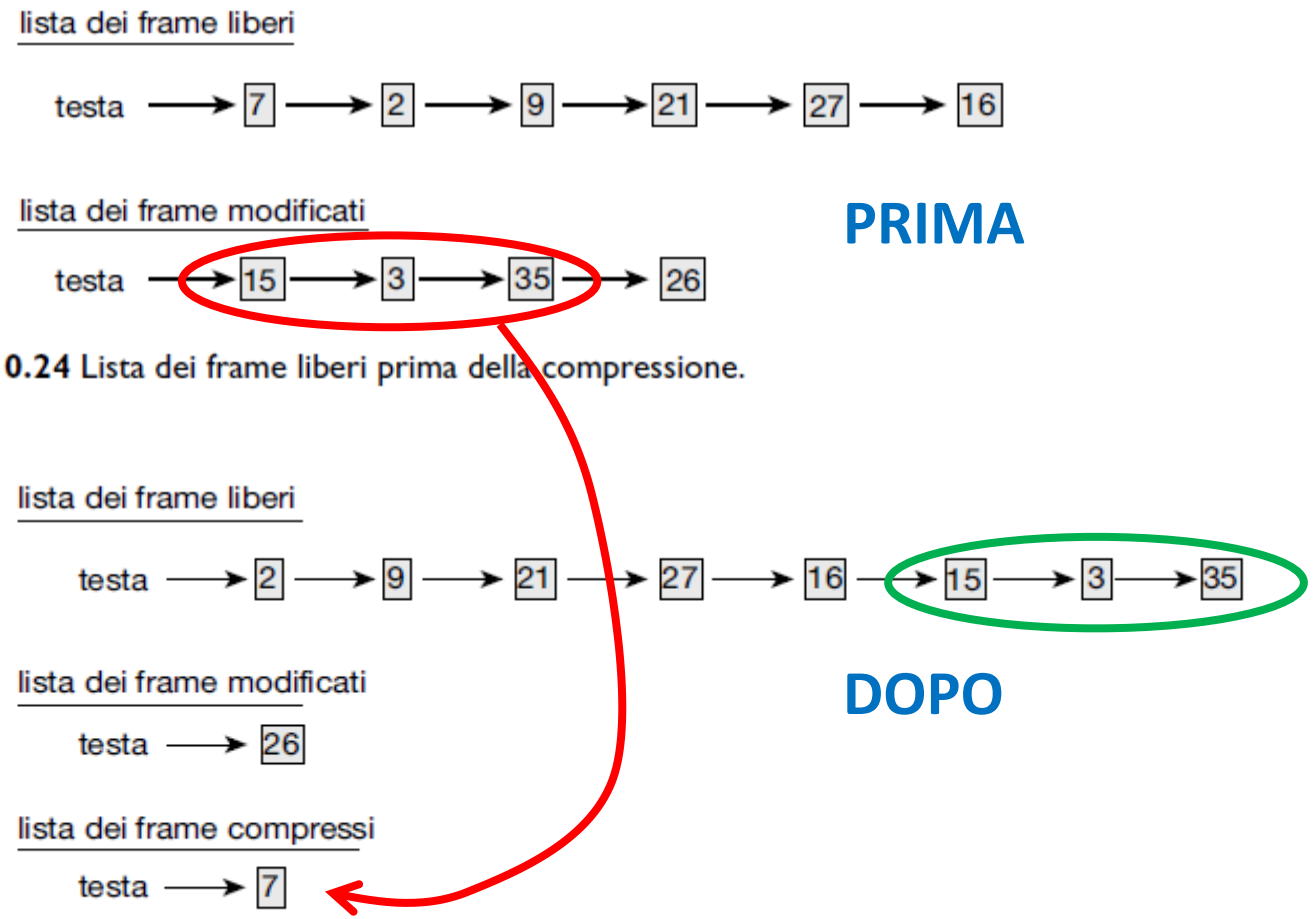


Figura 10.24 Lista dei frame liberi prima della compressione.

Figura 10.25 Lista dei frame liberi dopo la compressione.

Allocazione di memoria del kernel

La memoria del kernel è allocata in modo differente rispetto a quanto avviene per i processi in modalità utente, utilizzando

- blocchi contigui
- di dimensioni variabili.

Due tecniche comuni per l'**allocazione della memoria del kernel** sono:

- Sistema buddy
- Allocazione a lastre (slab)

Sistema buddy

Sistema buddy

Richiesta di memoria pari a 21KB

→

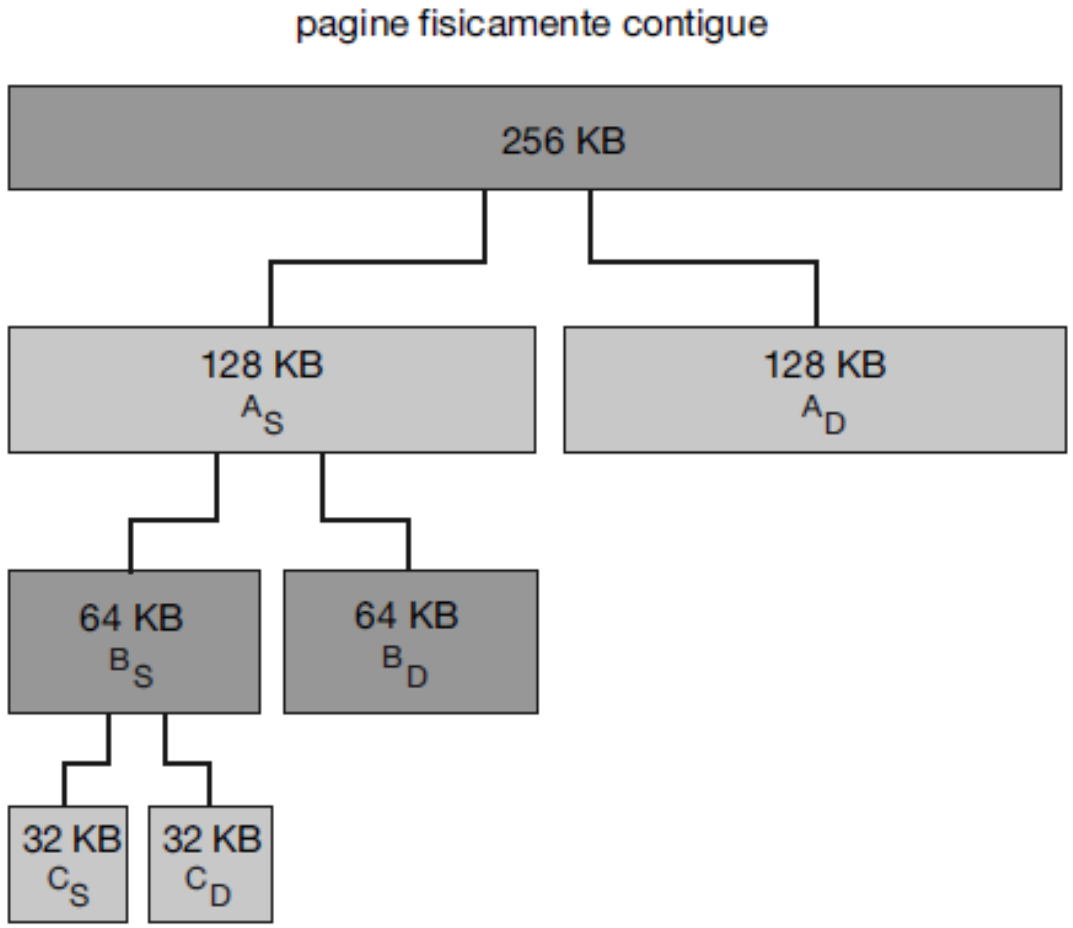


Figura 10.26 Sistema di allocazione buddy.

Allocazione buddy

- Buddy Allocator viene usato per allocare oggetti di dimensione variabile.
- Il buffer viene partizionato ricorsivamente in 2, creando di fatto un albero binario.
- La foglia più piccola che soddisfa la richiesta di memoria sarà utilizzata per soddisfare la richiesta.
- Il buddy associato a una foglia sarà l'altra regione ottenuta dalla divisione del parent.
- Se un oggetto è più piccolo della minima foglia che lo contiene, il restante spazio verrà sprecato.
- Quando un blocco viene rilasciato, esso verrà ricompattato con il suo buddy (se libero), risalendo fino al livello più grande non occupato.

Allocazione a lastre (SLAB)

Allocazione a lastre (SLAB)

- Una cache consiste di una o più lastre
- Ogni cache è popolata da oggetti
- Ogni oggetto è una istanza di una struttura dati del kernel

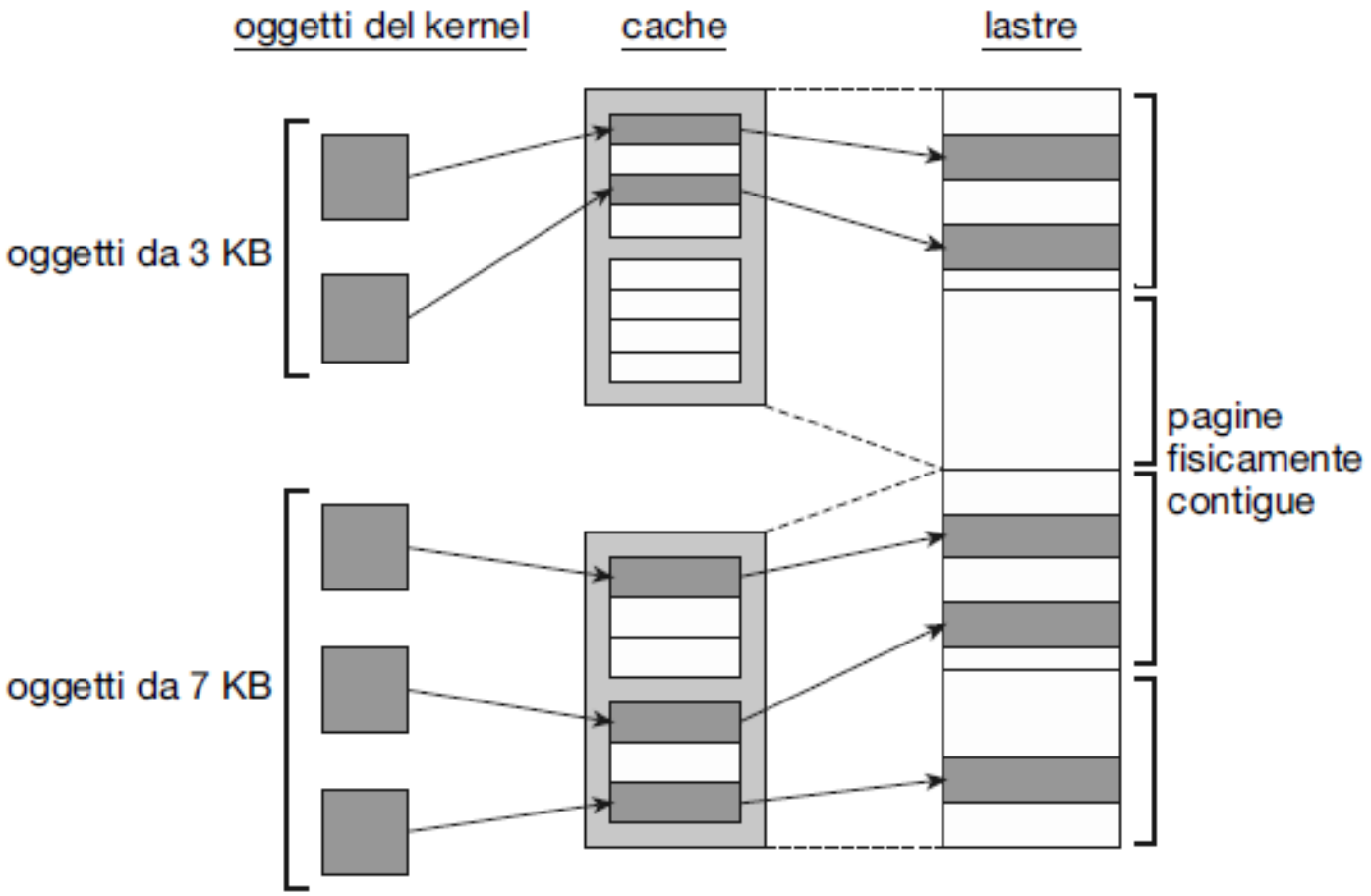


Figura 10.27 Allocazione a lastre (*slab*).

Allocazione SLAB

- Slab Allocator viene usato per allocare oggetti di dimensione fissa.
- Può allocarne fino ad un numero massimo fissato.

Portata del TLB

Tasso di successi (*hit ratio)* di un TLB → percentuale di traduzioni di indirizzi virtuali risolte dal TLB anziché dalla tabella delle pagine → proporzionale al numero di elementi del TLB

Portata del TLB → numero di elementi moltiplicato per la dimensione delle pagine

La **portata del TLB** esprime la quantità di memoria accessibile dal TLB

Una tecnica per aumentare la portata del TLB è aumentare la dimensione delle pagine.

Esempi di sistemi operativi

Realizzazione della memoria virtuale in:

Linux, Windows e Solaris gestiscono la memoria virtuale in modo simile, utilizzando, tra l'altro, la paginazione su richiesta e la copia su scrittura.

Ogni sistema utilizza anche una variante per approssimazione di LRU nota come algoritmo a orologio.

Linux

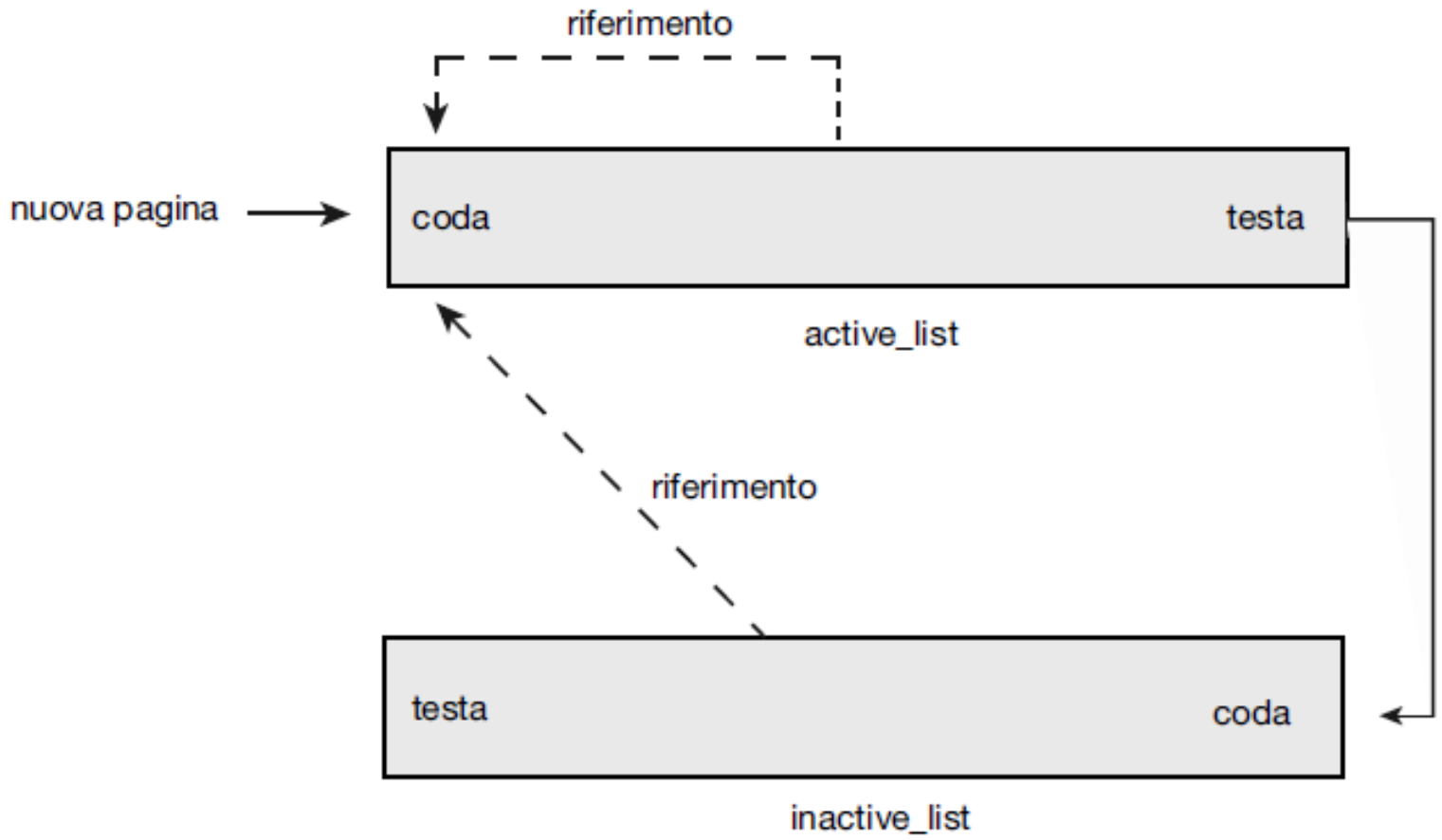
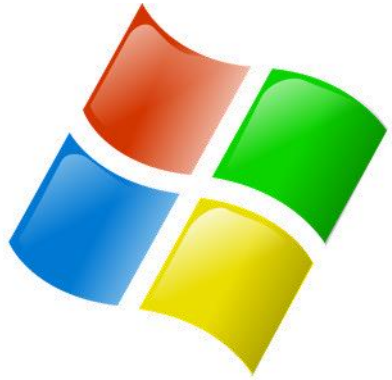


Figura 10.29 Le strutture `active_list` e `inactive_list` di Linux.

UNIVERSITÀ DEGLI STUDI DELLA BASILICATA



Corso di Sistemi Operativi

Allocazione della memoria

Docente:
Domenico Daniele Bloisi