

# **SISTEMI OPERATIVI**

Gestione della Memoria Centrale  
Tecniche di Base di Secondo Livello

## **Lezione 2 – Segmentazione**

**Vincenzo Piuri**

---

Università degli Studi di Milano

# Sommario

- Tecnica della segmentazione della memoria centrale
- Obiettivi
- Gestione
- Supporti hardware
- Frammentazione della memoria

# Problemi

- Stessi problemi affrontati dalla paginazione
- La paginazione vede  
lo spazio di indirizzamento logico  
come un unico vettore di indirizzi  
di memoria omogenei
- La paginazione non permette di tipizzare  
le varie porzioni di spazio di indirizzamento logico  
(ad esempio: codice, dati globali, heap, stack)
- Non permette la condivisione semplice  
ed efficiente di porzioni di memoria  
(ad esempio: codice)

## **Obiettivi** (1)

- Stessi obiettivi della paginazione  
(eccetto la dimensione identica delle porzioni)

## **Obiettivi (2)**

- Supportare la visione dello spazio di indirizzamento dei processi dal punto di vista dell'utente
- Dare una visione semantica alle porzioni di spazio di indirizzamento dei processi
- Supportare la tipizzazione di porzioni dello spazio di indirizzamento logico e il controllo degli accessi e delle operazioni ammissibili in base al tipo

## **Obiettivi** (3)

- Supportare la condivisione di porzioni dello spazio di indirizzamento tra diversi processi

# Segmentazione (1)

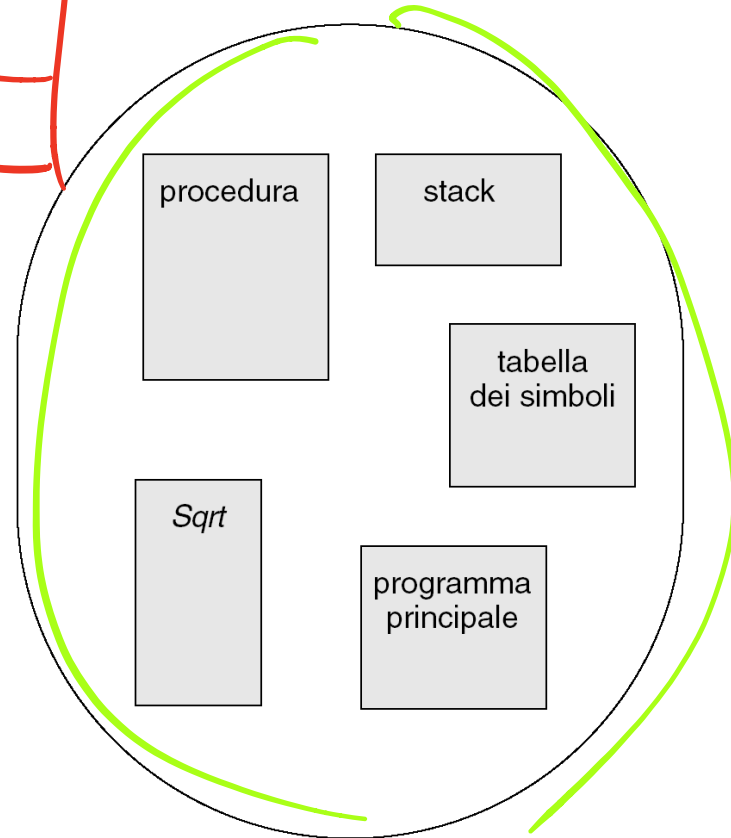
- Memoria centrale fisica divisa in **segmenti fisici** (frame)

struttura  
monodimensionale  
di memoria



- Spazio di indirizzamento del processo diviso in **segmenti logici** (segmenti)

struttura  
bidimensionale  
di memoria



# Segmentazione (2)

- I segmenti contengono informazioni di tipo diverso  
(tipizzazione dei segmenti)
- I segmenti possono avere dimensioni diverse
- Un segmento è caricato  
in un frame di ugual dimensione

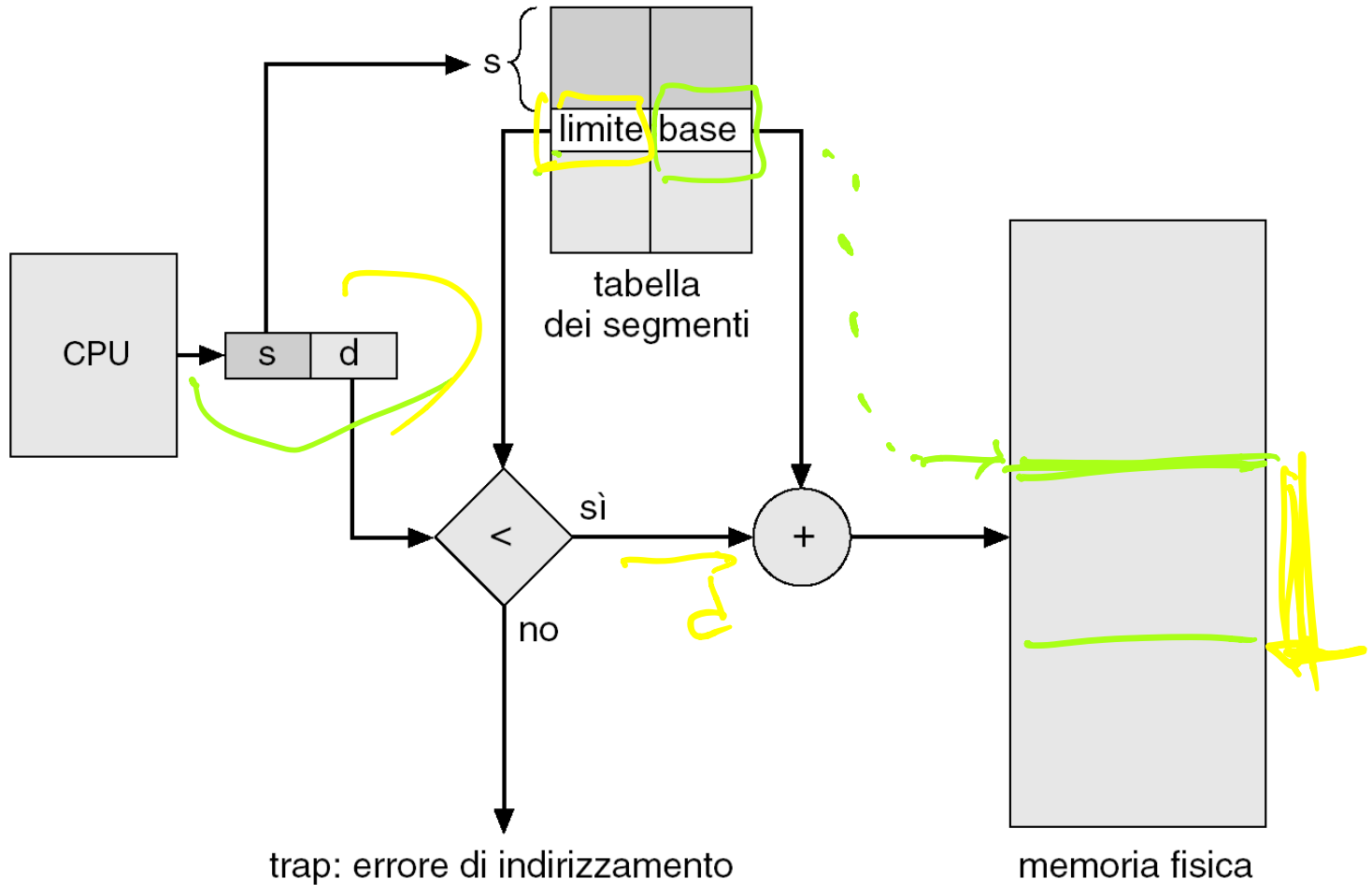


## Segmentazione (3)

- La **Tabella dei segmenti** di un processo definisce la corrispondenza tra i segmenti e i frame del processo considerato
- $\text{TabellaSegmenti}[\text{Segmento}] =$   
 $(\text{IndirizzoBaseFrame}, \text{DimensioneSegmento})$   

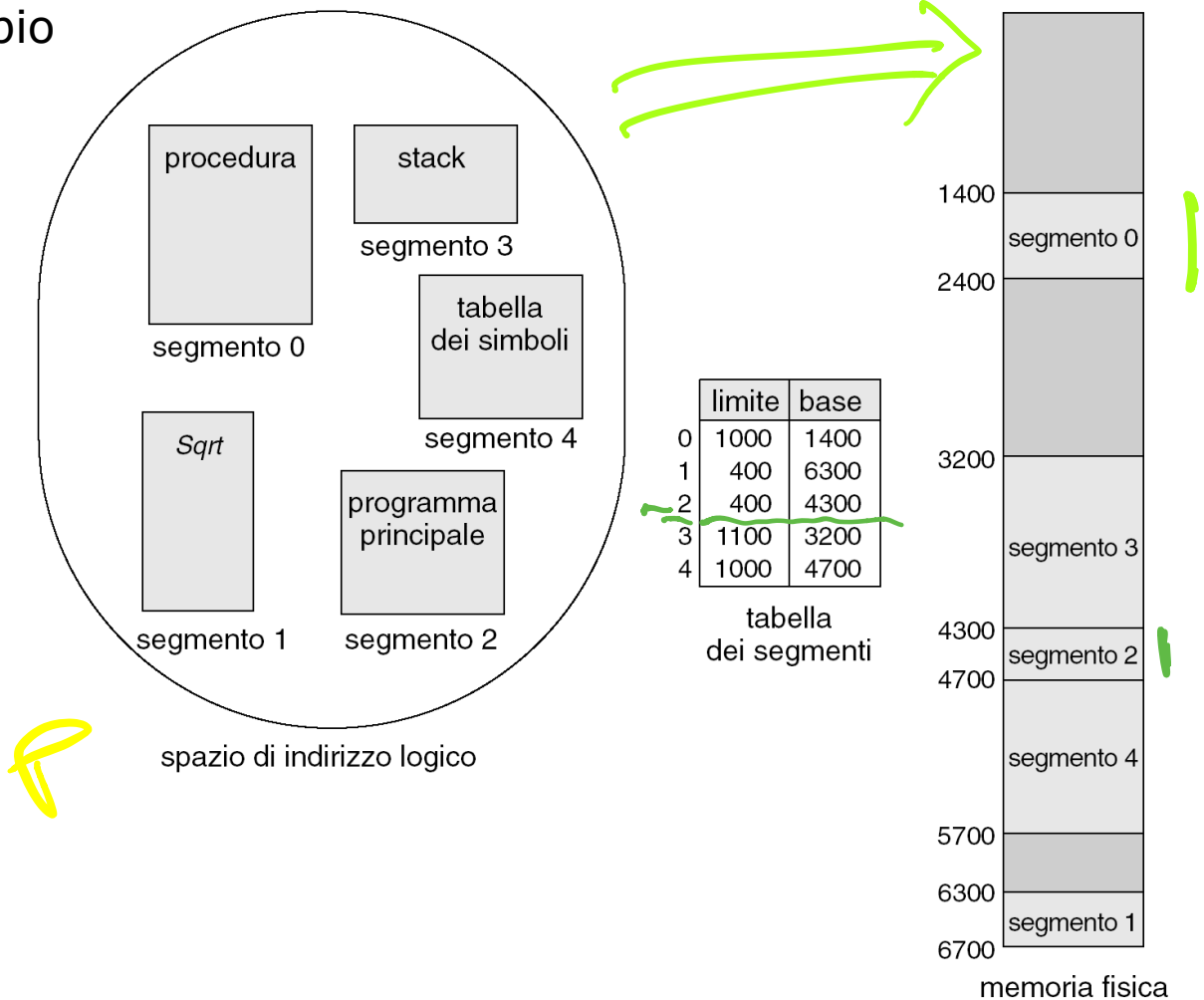
---
se caricato  
se non caricato
- Indirizzo logico =  
 $(\text{numero di segmento } \mathbf{s}, \text{spiazzamento nel segmento } \mathbf{d})$
- Indirizzo fisico =  
 $(\text{indirizzo di base del frame } \mathbf{bf}, \text{spiazzamento nel frame } \mathbf{d})$

# Segmentazione (4)



# Segmentazione (5)

Esempio



# Gestione della segmentazione (1)

- I segmenti necessari nell'immediato futuro alla computazione di processi nello stato di pronto vengono caricati in frame
- I segmenti di un processo possono essere caricati in frame non contigui in memoria centrale fisica

## **Gestione della segmentazione (2)**

- I segmenti non caricati sono conservati nell'area di swap
- I frame modificati vengono salvati in area di swap prima di essere rimossi dalla memoria centrale fisica

# Gestione della segmentazione (3)

- Il programmatore deve configurare la divisione del processo in segmenti

La configurazione è effettuata implicitamente mediante la strutturazione del programma in moduli compilati separatamente e nell'uso di librerie

- Il compilatore e il linker generano
  - un segmento codice per modulo sorgente
  - un segmento dati globali per modulo
  - un segmento per la tabella dei simboli
  - un segmento di stack
  - un segmento di heap

# Gestione della segmentazione (4)

- Il sistema operativo gestisce automaticamente
  - la selezione dei segmenti da caricare in memoria centrale fisica
  - il caricamento in memoria centrale dei segmenti necessari ma non presenti
  - la selezione dei frame da scaricare dalla memoria centrale fisica
  - lo scaricamento dei frame di memoria centrale non più necessari

# Supporto hardware: MMU

- Hardware dedicato per il supporto alla segmentazione

## Memory Management Unit

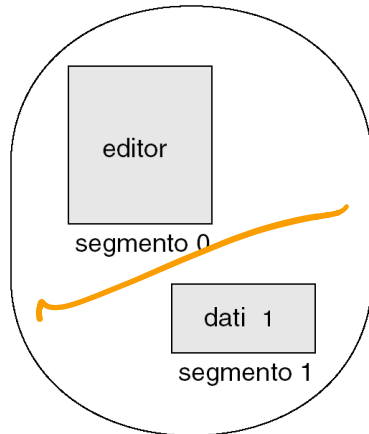
- Contiene la tabella dei segmenti o il suo indirizzo in memoria centrale
- Traduce l'indirizzo logico in indirizzo fisico



# Protezione dei segmenti

- Un processo può accedere solo ai suoi segmenti
  - La protezione dagli accessi di altri processi è implicita nella tabella dei segmenti
- Bit di protezione
  - Permettono di definire segmenti in
    - lettura/scrittura
    - sola lettura
    - sola esecuzione

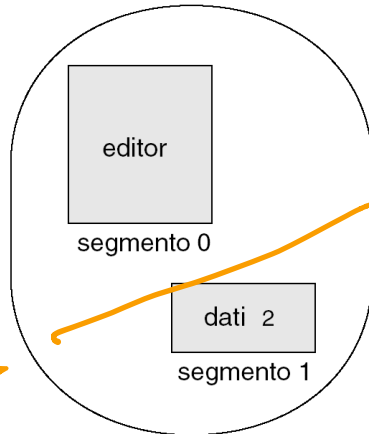
# Condivisione dei segmenti



memoria logica  
processo  $P_1$

|   | limite | base  |
|---|--------|-------|
| 0 | 25286  | 43062 |
| 1 | 4425   | 68348 |

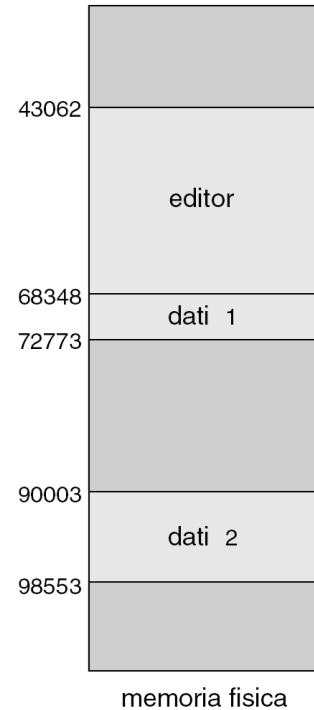
tabella dei segmenti  
processo  $P_1$



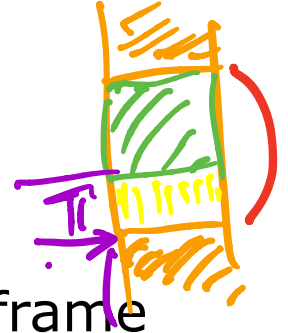
memoria logica  
processo  $P_2$

|   | limite | base  |
|---|--------|-------|
| 0 | 25286  | 43062 |
| 1 | 8850   | 90003 |

tabella dei segmenti  
processo  $P_2$



# Frammentazione della memoria



- Un segmento viene caricato in un frame di dimensione sufficiente a contenerlo
- Uno sfrido può rimanere inutilizzato nel frame
- Causa: dimensioni variabili dei segmenti
- Effetto: frammentazione esterna della memoria
- Soluzione: collezione degli sfridi di memoria (garbage collection) per creare frame liberi grandi, con eventuale rilocalizzazione dei segmenti caricati

# In sintesi

- Abbiamo visto:
  - tecnica della segmentazione della memoria centrale
  - obiettivi
  - gestione
  - supporti hardware
  - protezione
  - frammentazione della memoria
- Notiamo che la segmentazione
  - crea **spazio logico più grande dello spazio fisico** assegnato ad un processo nella memoria centrale fisica
  - è **configurata implicitamente dal programmatore**
  - è **gestita automaticamente dal sistema operativo**
  - è **efficiente** poiché sposta piccole porzioni di memoria