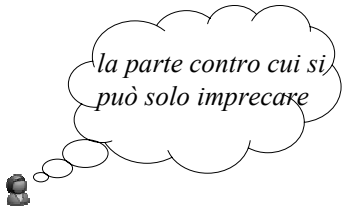
Il software

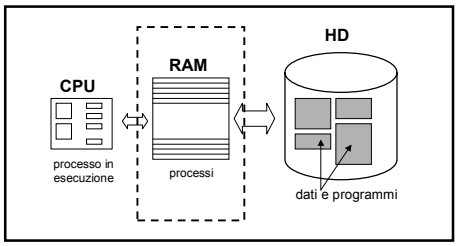


(continua)

Funzioni principali del sistema operativo

- Avvio dell'elaboratore
- Gestione del processore e dei processi in esecuzione
- Gestione della memoria principale
- Gestione della memoria virtuale
- Gestione della memoria secondaria
- Gestione dei dispositivi di input / output
- Interazione con l'utente

Gestione della memoria principale

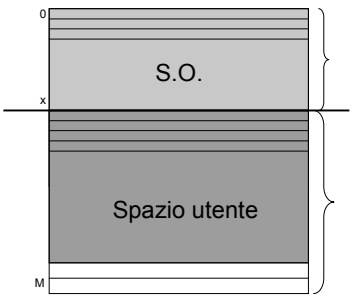


Gestione della memoria principale

- Più processi "contemporaneamente" in esecuzione
  1) Come possono **condividere** l'uso della memoria principale?
  2) Come possiamo garantire che i processi **non si "danneggino"** tra loro?
  3) Quanti processi possono essere caricati in memoria contemporaneamente?
  4) Esiste una **dimensione massima** per i processi?

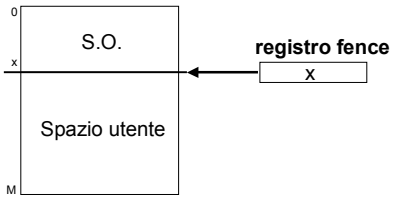
Sistema monoprogrammato

Sistema monoprogrammato

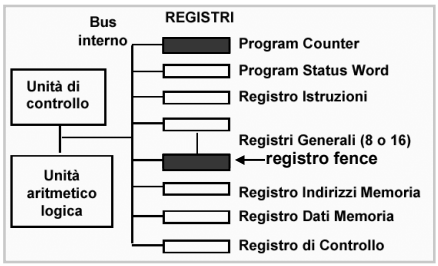


Sistema monoprogrammato

- I processi si alternano nell'uso della memoria che occupano fino a quando non terminano la loro computazione
- Come possiamo garantire che durante la loro esecuzione non danneggino il S.O.?



Riprendiamo la struttura della CPU …



Sistema monoprogrammato

- Il Program Counter (PC) contiene sempre l'indirizzo della prossima istruzione da eseguire
- Per ogni accesso in memoria da parte di un processo utente si verifica

```
if (val[PC] <= val[fence])
then <ERRORE e gestione S.O.>
else <OK, esecuzione istruzione>
```

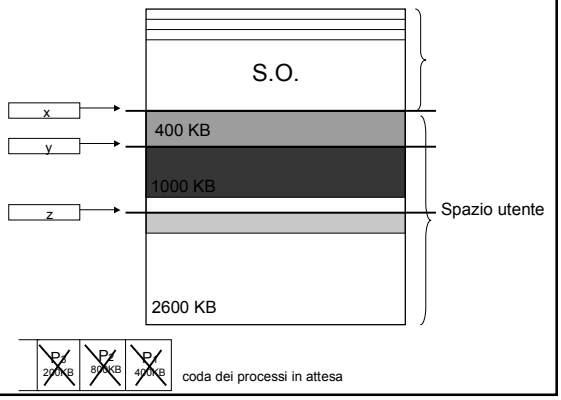
Sistema multiprogrammato

- Più programmi utente possono essere eseguiti "contemporaneamente"
- Si devono quindi studiare delle tecniche per condividere la memoria principale
  - **Allocazione contigua**
  - **Allocazione non contigua**

Allocazione contigua

- La memoria viene **suddivisa in parti** che devono contenere le immagini dei processi
  - **Partizioni fisse:** si decide quali e quante saranno le partizioni della memoria al momento della configurazione del S.O.
  - **Partizioni variabili:** le partizioni vengono create dinamicamente quando i processi vengono caricati in memoria

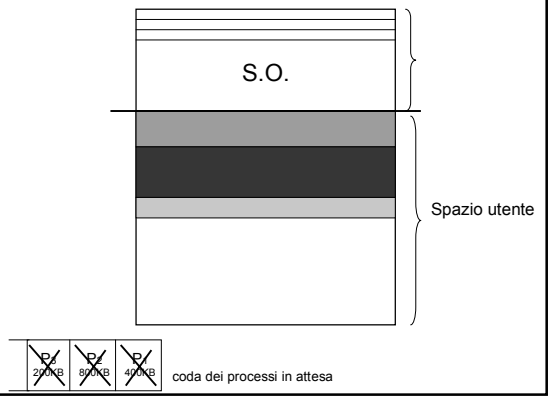
Allocazione contigua: partizioni fisse



Partizioni fisse: problemi

- Limite sulla dimensione massima dei processi
- Come si possono dimensionare le partizioni?
- Spreco di spazio all'interno delle partizioni: **frammentazione interna**

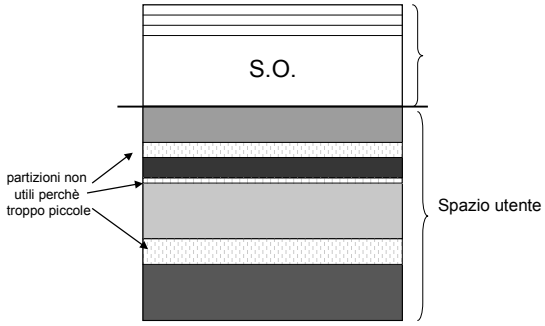
Allocazione contigua: partizioni variabili



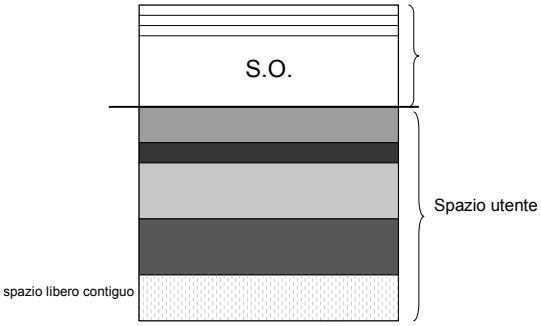
Partizioni variabili: problemi

- Quando un processo termina rimane uno spazio libero. Il prossimo processo dove deve essere caricato?
- Si deve **minimizzare la frammentazione esterna**, ovvero la creazione di spazi troppo piccoli che non potranno contenere nessun processo

Frammentazione esterna

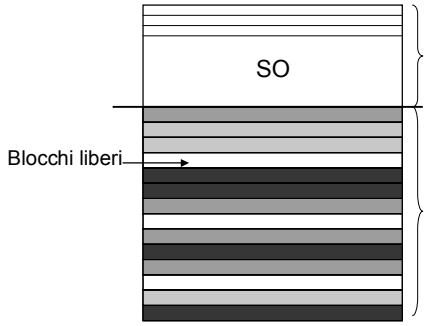


Compattazione



Allocazione non contigua

- I processi vengono caricati in RAM "a pezzi"



Ritorniamo alla tabella dei processi

- Per ogni processo vi è un **descrittore** nel quale sono memorizzate molte informazioni, tra cui
  - L'identificatore del processo
  - L'identificatore dell'utente proprietario
  - Lo stato del processo
  - Il contenuto del registro Program Counter e degli altri registri
  - Informazioni sui file e sulle risorse in uso
  - …
  - **Dove sono caricate in RAM le sue pagine**

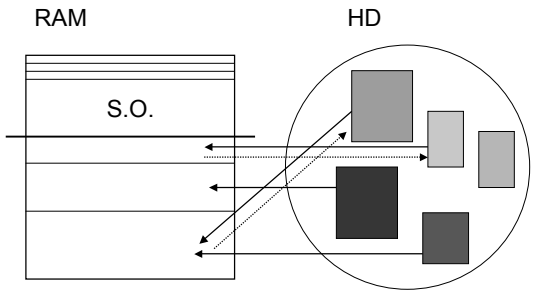
Memoria virtuale

- Visione **astratta** della memoria: fornisce agli utenti l'impressione che la memoria a disposizione sia più grande di quella reale
- Due tecniche principali
  - **Swapping**
  - **Demand paging**

Swapping

- Al momento del cambio di contesto, il processo in esecuzione
  - perde l'uso del processore
  - se necessario, perde l'uso della memoria principale e la sua immagine viene ricopiate sulla memoria secondaria

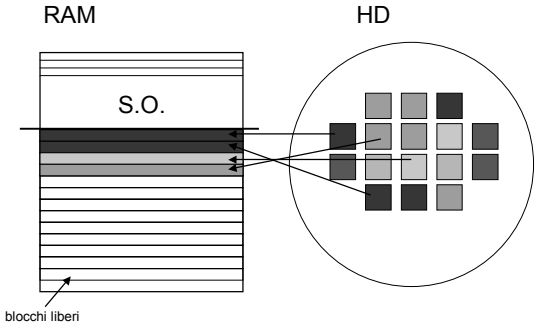
Swapping



Demand paging

- Lo swapping impone un limite massimo alle dimensioni dei processi
- Con il demand paging si possono avere programmi (processi) che sono indipendenti dalle dimensioni della RAM
- Le immagini dei processi vengono caricate in RAM a "pezzi" (paging) e su richesta (demand)

Demand paging



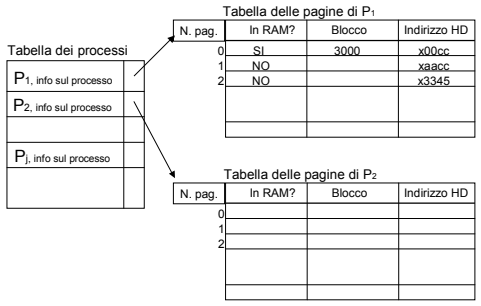
Demand paging

- Quando il processo in esecuzione fa riferimento ad una istruzione che si trova in una pagina che non è ancora stata caricata in memoria si ha un **page fault**
  - Il processo in esecuzione viene messo nello stato di **attesa**
  - Il sistema operativo si occupa del **caricamento della pagina** in memoria principale
  - Viene mandato in esecuzione **un altro processo**
  - Non appena la pagina è caricata in memoria principale il processo viene rimesso nello stato di **pronto**

Demand paging

- La gestione da parte del sistema operativo si complica: **tabella dei blocchi (frame)**
  - tiene traccia dei blocchi **liberi** e di quelli **occupati**
  - memorizza informazioni sull'uso dei blocchi (serve per decidere quali pagine rimpiazzare)
- Inoltre, per ogni processo, si deve avere anche una **tabella delle** sue **pagine**

Demand paging



NB: queste tabelle stanno nella parte di RAM riservata al Sistema Operativo

Indirizzi virtuali

- Ogni istruzione è caratterizzata da un **indirizzo virtuale** che tiene traccia
  - della **pagina** in cui si trova l'istruzione
  - della sua posizione nella pagina (**offset**)
- Quando la pagina viene caricata in RAM, gli indirizzi fisici vengono calcolati considerando l'indirizzo del blocco in cui viene caricata la pagina e l'offset dell'istruzione nella pagina

Gestione della memoria?