Gestione della memoria

- Background
- Cenni ai Sistemi Mono-programmati
- Partizioni Fisse
- Swapping (Partizioni variabili)
- Memoria Virtuale
  - Paginazione

Basato parzialmente sulle slide originali di Silberschatz, Galvin and Gagne

Background

- Gerarchia di memoria
  - Cache
    - Molto veloce, dispendiosa e volatile
  - Memoria principale
    - Velocità media, prezzo medio, volatile
  - Memoria secondaria (disco)
    - Lenta, poco costosa, non volatile
- Gestore della memoria
  - Si occupa di gestire opportunamente la gerarchia di memoria
  - Alloca e dealloca i processi in memoria principale
  - Tiene traccia dello spazio libero
  - Scarica qualche (parte di) processo su disco quando lo spazio diventa insufficiente

Background

Nei sistemi in cui esistono

- Job Queue
  - contiene processi in attesa di memoria
- Schedulatore a lungo termine
  - Selezione il prossimo processo della job queue a cui allocare la memoria

Il Gestore della memoria opera in collaborazione con lo schedulatore a lungo termine

Classificazione

- Gestione **senza swapping**
  - I processi ottengono lo spazio di memoria all'inizio dell'esecuzione e lo mantengono per tutta la durata dell'esecuzione
- Gestione **con swapping**
  - I processi possono entrare ed uscire dalla memoria
  - Un processo (o parte di esso) può essere deallocato per mancanza di spazio
  - Permette un maggior grado di multiprogrammazione
  - Richiede meccanismi di rilocazione dinamica

Rilocazione dinamica (1)

- I programmi contengono istruzioni di salto
  - Chiamate di procedure, ritorno al chiamante
  - Istruzioni condizionali, ripetitive, di salto
  - ....
- L'assemblatore traduce queste istruzioni come

  Goto <indirizzo>
- L'assemblatore traduce il programma sorgente assumendo che inizi dall'indirizzo iniziale 0…0
- Gli indirizzi riferiti nel codice dipendono dall'indirizzo di partenza
  - Indirizzi delle istruzioni di salto
  - indirizzi degli operandi

Rilocazione dinamica (2)

- Il programma non viene generalmente caricato a partire dall'indirizzo 0….0
- Indirizzo effettivi
  - Raramente sono noti a tempo di compilazione
  - Possono essere conosciuti al momento del caricamento
  - Può cambiare durante l'esecuzione
    - Sistemi che usano swapping
    - Sistemi con memoria virtuale
- Gli indirizzi devono essere rilocati
  - Ad eccezione del caso in cui gli indirizzi effettivi sono noti a tempo di compilazione (codice assoluto)

Indirizzo logico e fisico

- Indirizzo logico (o virtuale)
  - Indirizzo generato dal programma (dalla CPU)
- Spazio di indirizzamento logico
  - insieme degli indirizzi logici
- Indirizzo fisico
  - Indirizzo visto dalla memoria
- Spazio di indirizzamento fisico
  - insieme degli indirizzi fisici

Binding degli indirizzi

Traduzione dell'indirizzo logico nell'indirizzo fisico

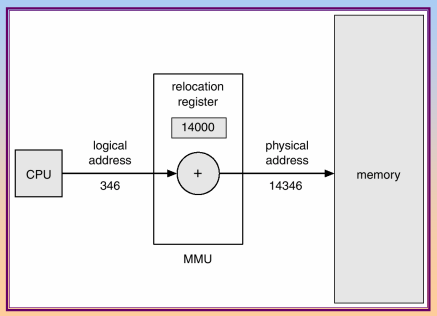
- **Fase di Compilazione**
  - il compilatore genera codice assoluto.
  - Deve essere ricompilato se cambia lo spazio di memoria
- **Fase di Caricamento**
  - La rilocazione viene fatta poco prima di caricare il programma in memoria
- **Fase di esecuzione**
  - Il processo può essere spostato da un segmento di memoria ad un altro durante l'esecuzione
  - Necessita di supporti hw (e.g., registro base e registro limite).

Binding degli indirizzi (2)

- Indirizzo logico == indirizzo fisico
  - In sistemi senza rilocazione dinamica
- In sistemi senza rilocazione dinamica l'indirizzo logico viene inviato direttamente alla memoria
- In sistemi con rilocazione dinamica l'indirizzo logico viene inviato alla MMU per la traduzione
- MMU (Memory Management Unit)
  - Esegue la rilocazione dinamica
  - Necessaria perché il programma può essere caricato in punti diversi della memoria durante l'esecuzione

Rilocazione dinamica

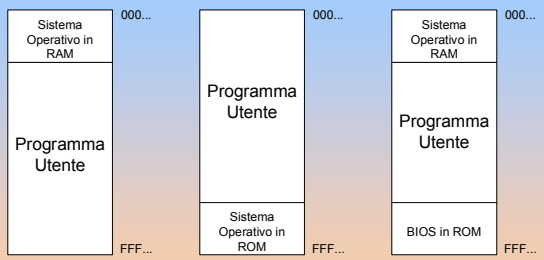
Basata su registro di rilocazione



Gestione della memoria

- Background
- **Cenni ai Sistemi Mono-programmati**
- Partizioni Fisse
- Swapping (Partizioni variabili)
- Memoria Virtuale
  - Paginazione

Sistemi mono-programmati



Usato in passato in mainframe e mini computer

Usato attualmente in sistemi embedded

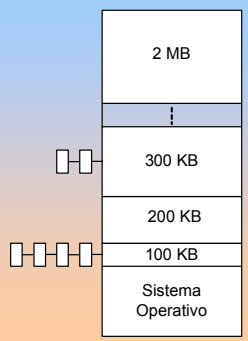
Usato in passato in Personal Computer (DOS)

Sistemi Multi-programmati

- Partizioni fisse
- Swapping (Partizioni Variabili)
- Memoria Virtuale

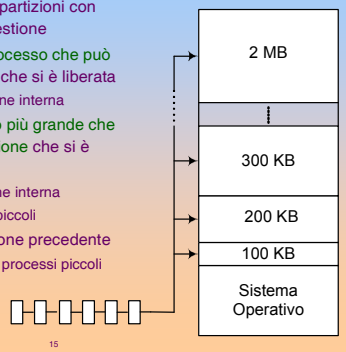
Partizioni Fisse (1)

- Partizioni di dimensione pre-fissata
- Code separate per ciascuna partizione
- Processi accodati nella coda della partizione più piccola in grado di soddisfare la richiesta di memoria
- Frammentazione interna
  - inevitabile
- Partizioni inutilizzate e processi in per partizioni più piccole
- Soluzione: coda unica



Partizioni fisse (2)

- Coda unica per tutte le partizioni con diverse alternative di gestione
- Viene scelto il primo processo che può entrare nella partizione che si è liberata
  - Elevata frammentazione interna
- Viene scelto il processo più grande che può entrare nella partizione che si è liberata
  - Minore frammentazione interna
  - Sfavorisce i processi piccoli
- Integrazione alla soluzione precedente
  - Partizione riservata ai processi piccoli
  - Meccanismo di **aging**



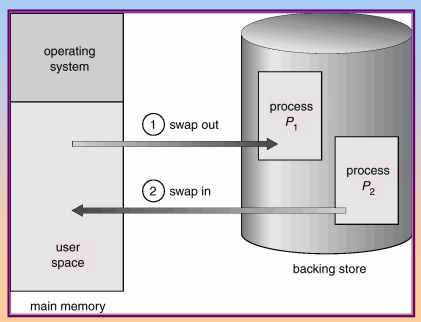
Allocazione dinamica della memoria

- Nei sistemi batch un job rimane in memoria per tutta la sua esecuzione
- Nei sistemi interattivi e nei PC a volte non c'è sufficiente memoria per contenere tutti processi attivi
- I processi in eccesso vengono scaricati su disco temporaneamente e ricaricati successivamente
- Swapping
  - Un processo viene eseguito per qualche tempo, poi scaricato nuovamente su disco, quindi ricaricato, ….
- Memoria virtuale
  - Consente di eseguire processi caricati in memoria solo parzialmente

Swapping

- A process can be *swapped* temporarily out of memory to a *backing store*, and then brought back into memory for continued execution.
- Backing store (swap space)
  - fast disk large enough to accommodate copies of all memory images for all users
  - must provide direct access to these memory images.
- *Roll out, roll in* – swapping variant used for priority-based scheduling algorithms; lower-priority process is swapped out so higher-priority process can be loaded and executed.
- Major part of swap time is transfer time; total transfer time is directly proportional to the *amount* of memory swapped.
- Modified versions of swapping are found on many systems, i.e., UNIX, Linux, and Windows.

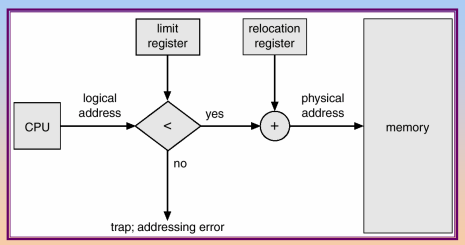
Schematic View of Swapping



Contiguous Allocation

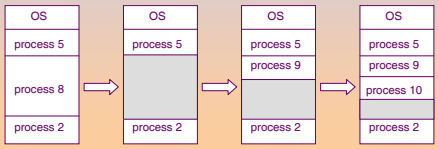
- Main memory usually into two partitions:
  - Resident operating system
  - User processes
- User-partition allocation
  - The second partition is allocated to user processes in a contiguous way
  - Relocation-register scheme used to protect user processes from each other, and from operating-system code and data.
  - Relocation register contains value of smallest physical address; limit register contains range of logical addresses – each logical address must be less than the limit register.

Hardware Support for Relocation and Limit Registers



Contiguous Allocation (Cont.)

- Multiple (variable size) partitions
- *Hole*
  - block of available memory;
  - holes of various size are scattered throughout memory.
- When a process arrives, it is allocated memory from a hole large enough to accommodate it.



Dynamic Storage-Allocation Problem

How to satisfy a request of size *n* from a list of free holes.

- **First-fit**: Allocate the *first* hole that is big enough.
- **Best-fit**: Allocate the *smallest* hole that is big enough
  - Must search entire list, unless ordered by size.
  - Produces the smallest leftover hole.
- **Worst-fit**: Allocate the *largest* hole
  - must also search entire list.
  - Produces the largest leftover hole.

First-fit and best-fit perform better than worst-fit in terms of speed and storage utilization.

Fragmentation

- **External Fragmentation**
  - total memory space exists to satisfy a request, but it is not contiguous.
- **Internal Fragmentation**
  - allocated memory may be slightly larger than requested memory;
  - this size difference is memory internal to a partition, but not being used.
- External fragmentation can be reduced by compaction
  - Shuffle memory contents to place all free memory together in one large block.
  - Compaction is possible *only* if relocation is dynamic, and is done at execution time.
  - May be expensive in terms of time overhead

Gestione dello spazio libero

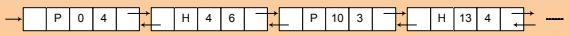
- Il sistema operativo deve mantenere informazioni su
  - Partizioni di memoria allocate
  - Buchi di memoria liberi
- Strutture dati
  - Mappa di bit
  - Lista concatenata

Gestione dello spazio libero

Mappa di bit

- La memoria viene suddivisa in tante unità di allocazione
- A ogni unità di allocazione viene associato un bit nella mappa
  - Esempio: 0 se l'unità è libera, 1 se occupato
- Per allocare un certo spazio di *K* unità di allocazione il gestore cerca nella mappa una sequenza di *K* bit 0 consecutivi
- Dimensione della unità di allocazione: compromesso fra
  - Frammentazione interna
  - Overhead

Gestione dello spazio libero

Lista concatenata

- Ogni elemento della lista rappresenta una partizione allocata a un processo oppure un buco libero
- Per allocare spazio di memoria si cerca un buco libero di dimensioni opportune.
- Il processo di allocazione comporta generalmente
  - Inserzione di un nuovo elemento (relativo al nuovo processo)
  - Modifica della dimensione del buco da cui si alloca
- Il processo di de-allocazione può comportare
  - Cambiare solo l'attributo da P a H
  - Fusione di due elementi vicini
  - Fusione di tre elementi vicini