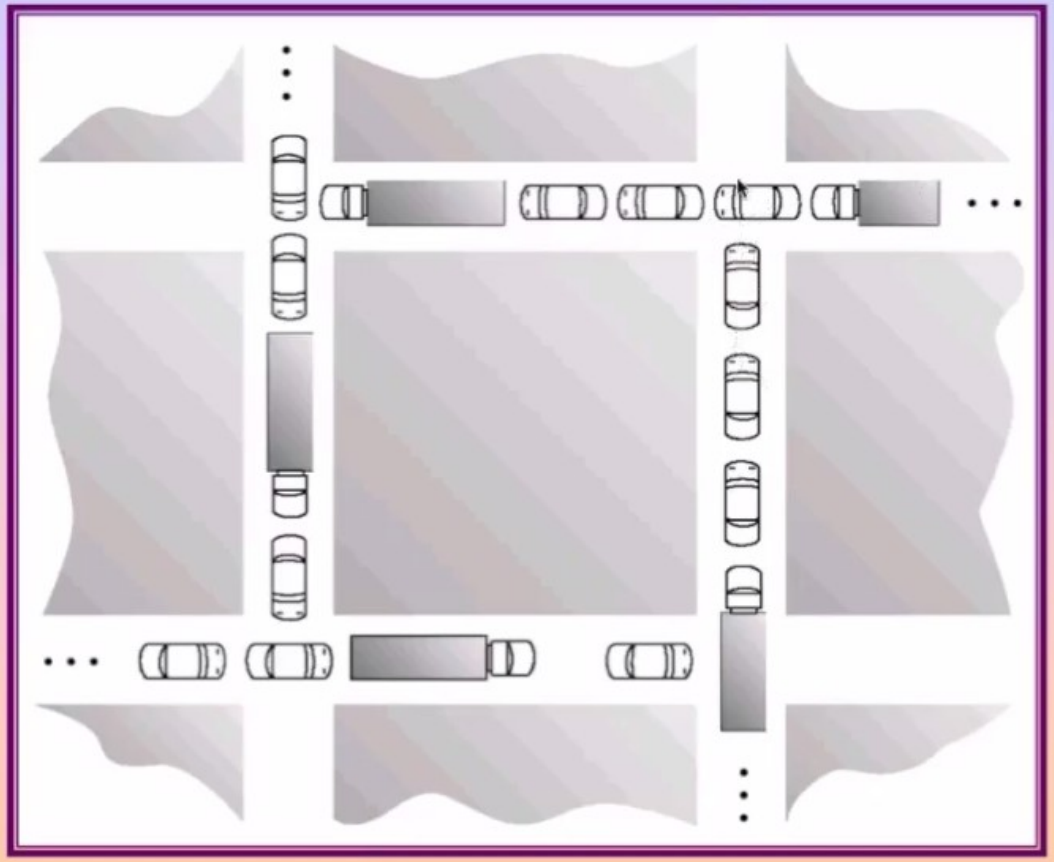
Capitolo 7: Stallo dei processi

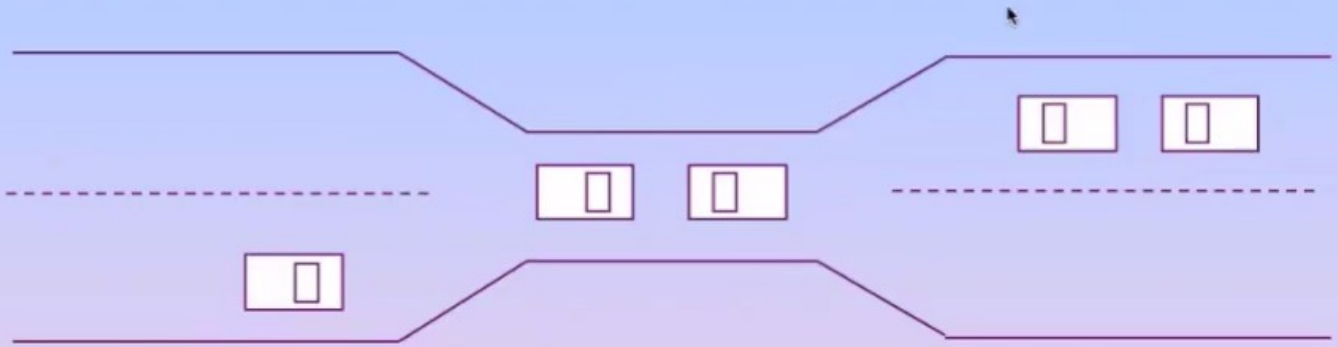
- Modello del sistema
- Caratterizzazione delle situazioni di stallo
- Metodi per la gestione delle situazioni di stallo
- Prevenire le situazioni di stallo
- Evitare le situazioni di stallo
- Rilevamento delle situazioni di stallo
- Ripristino da situazioni di stallo

Stallo (*Deadlock*)

Un insieme di processi e' in stallo se ciascun processo nell'insieme è in attesa di un evento che può essere causato solo da un altro dei processi nell'insieme.



Esempio: attraversamento di un ponte



- In ogni istante, sul ponte possono transitare autoveicoli solo in una direzione.
- Ciascuna sezione del ponte può essere vista come una risorsa.
- Se si verifica una situazione di stallo, questa può essere risolta se un'auto torna indietro (rilascia la risorsa ed esegue un ritorno ad uno stato "sicuro" - *rollback*).
- In caso di deadlock, può essere necessario che più auto debbano tornare indietro.
- È possibile si verifichi attesa indefinita (starvation).

Il problema del deadlock

- In un ambiente multiprogrammato più processi possono entrare in competizione per cercare di ottenere risorse condivise.
- Se una risorsa non è correntemente disponibile, il processo richiedente entra in stato di attesa.
- Se le risorse sono trattenute da altri processi, a loro volta in stato di attesa, il processo potrebbe non poter più cambiare il proprio stato.
- Esempio
  - Nel sistema sono presenti solo due unità a nastri
  - P_1 e P_2 hanno già avuto assegnate un'unità ciascuno
  - Entrambi si bloccano e richiedono l'assegnazione di una seconda unità

Risorse riutilizzabili e risorse non riutilizzabili

- Risorse riutilizzabili:
  - Questo tipo di risorsa può essere usata in modo sicuro da un processo alla volta,ed essere riutilizzata in seguito da un altro processo
  - Esempi di risorse riutilizzabili : processori, canali I/O, memoria centrale e secondaria, basi di dati, semafori, file, dispositivi
- Risorse non riutilizzabili:
  - Questo tipo di risorsa dopo essere stata creata,viene consumata (distrutta)
  - Esempi di risorse non riutilizzabili : interrupt, segnali, messaggi, dati contenuti nel buffer I/O, etc.

Modello di sistema

- Risorse di tipo $R_1, R_2, \ldots, R_m$

  (Cicli di CPU, spazio di memoria, file, dispositivi di I/O)

- Sono disponibili W_i istanze di ciascuna risorsa di tipo R_i.
- Nelle ordinarie condizioni di funzionamento un processo può servirsi di una risorsa solo se rispetta la seguente sequenza:
  - **Richiesta** — se la richiesta non può essere soddisfatta immediatamente, perché la risorsa non è libera, il processo richiedente deve attendere fino a che non può acquisire la risorsa.
  - **Utilizzo** — il processo può operare sulla risorsa.
  - **Rilascio** — il processo rilascia la risorsa.
- Richiesta e rilascio di risorse sono realizzati con apposite system call:
  - *request/release device*, *open/close file*, *allocate/free memory*, o *wait* e *signal* su semafori.

Caratterizzazione dei deadlock

Una situazione di deadlock può verificarsi *solo se* valgono ***tutte e quattro*** le seguenti condizioni simultaneamente :

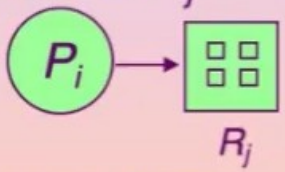
- **Mutua esclusione:** almeno una risorsa deve essere "non condivisibile", cioè può essere usata da un solo processo alla volta. Se un altro processo richiede questa risorsa, si deve bloccare il processo richiedente fino al rilascio della risorsa.
- **Possesso ed attesa:** un processo, in possesso di almeno una risorsa, attende di acquisire ulteriori risorse già in possesso di altri processi.
- **Impossibilità di prelazione:** non esiste un diritto di prelazione. Una risorsa può essere rilasciata dal processo che la possiede solo volontariamente, al termine del suo utilizzo.
- **Attesa circolare:** deve esistere un insieme $\{P_0, P_1, \ldots, P_n\}$ di processi in attesa, tali che P_0 è in attesa di una risorsa che è posseduta da P_1, P_1 è in attesa di una risorsa posseduta da P_2, …, P_{n-1} è in attesa di una risorsa posseduta da P_n, e P_n è in attesa di una risorsa posseduta da P_0 .

Grafo di allocazione risorse (II)

- Processo

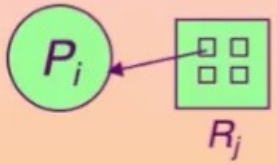
- Tipo di risorsa con 4 istanze

- P_i richiede un'istanza di R_j



Arco di richiesta

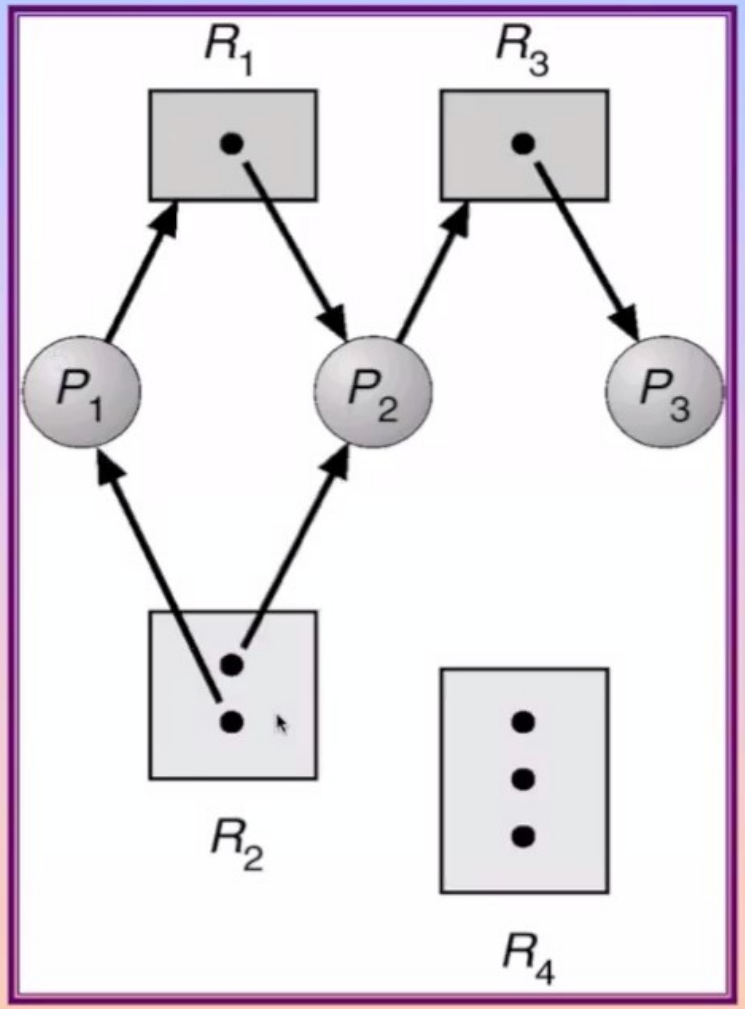
- P_i possiede un'istanza di R_j



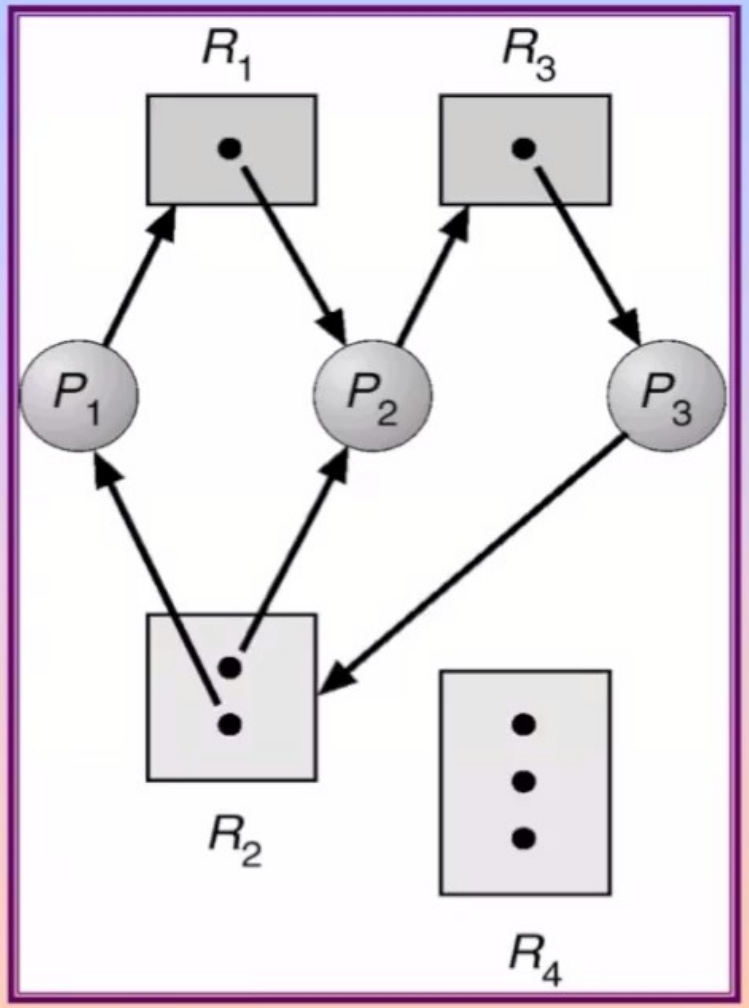
Arco di assegnazione

Esempi di grafo di allocazione risorse

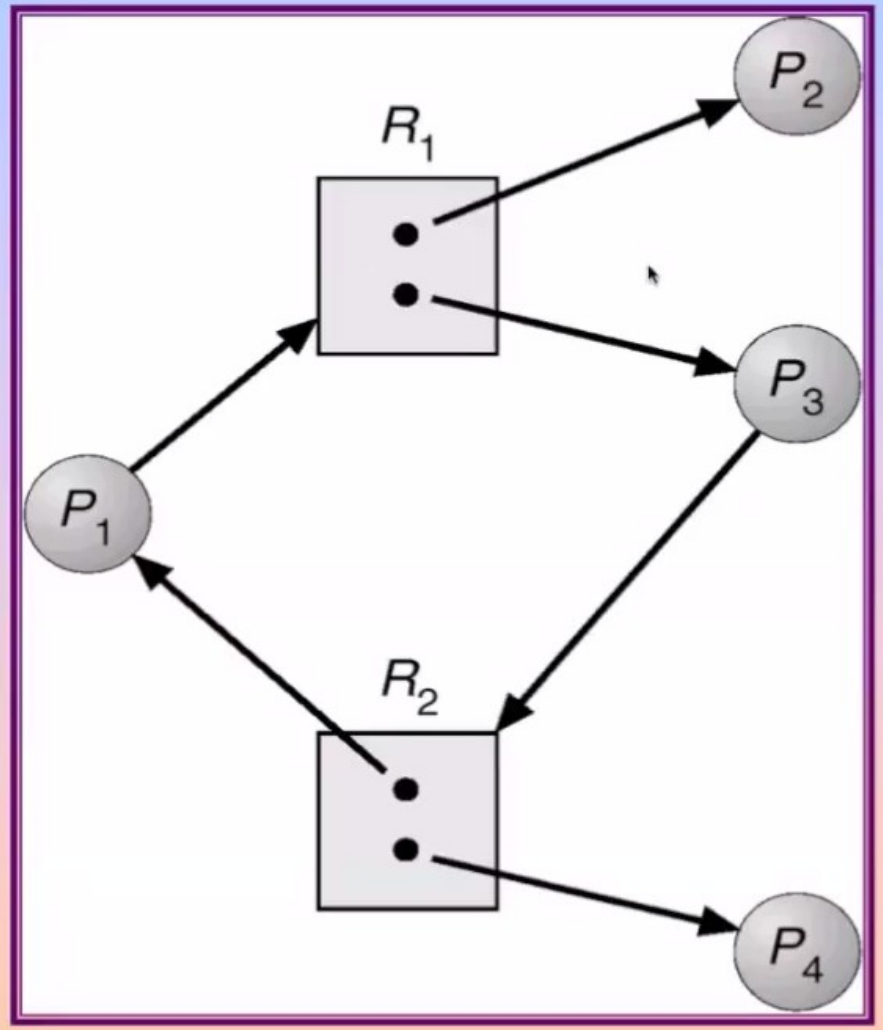
Grafo di allocazione risorse



Grafo di allocazione risorse con deadlock



Grafo di allocazione risorse (III)



- Se il grafo non contiene cicli ⇒ non ci sono deadlock.
- Se il grafo contiene un ciclo ⇒
  - Se vi è una sola istanza per ogni tipo di risorsa, allora si ha un deadlock.
  - Se si hanno più istanze per tipo di risorsa, allora il deadlock è possibile (ma non certo).
  - Nell'esempio abbiamo un ciclo ma non un deadlock.

Metodi per la gestione dei deadlock

- Assicurare che il sistema non entri *mai* in uno stato di deadlock.
  - **Prevenire i deadlock**:
    - ✓ evitare che contemporaneamente si verifichino mutua esclusione, possesso e attesa, impossibiltà di prelazione e attesa circolare ⇒ basso utilizzo risorse e throughput ridotto.
  - **Evitare i deadlock**:
    - ✓ evitare gli stati del sistema a partire dai quali si può evolvere verso un deadlock.
- Permettere al sistema di entrare in uno stato di deadlock, quindi ripristinare il sistema.
  - **Determinare la presenza di un deadlock**.
  - **Ripristinare il sistema da un deadlock**.
- Ignorare il problema e "fingere" che i deadlock non avvengano mai nel sistema; impiegato dalla maggior parte dei S.O., incluso UNIX.