Capitolo 3

Risorse e Stallo

3.1. Risorse
3.2. Introduzione
3.3. L'algoritmo dello struzzo
3.4. Riconoscimento e eliminazione dello stallo
3.5. Impedire lo stallo
3.6. Evitare lo stallo
3.7. Problemi correlati

Risorse (1)

- Esempi di risorse
  - stampanti
  - nastri
  - tabelle
- I processi devono accedere alle risorse in un ordine ragionevole
- Supponiamo che un processo occupi la risorsa A e che richieda la risorsa B
  - Se allo stesso istante un altro processo occupa B e richiede A
  - Entrambi i processi restano bloccati

Risorse (2)

- Lo stallo avviene se …
  - i processi hanno accesso esclusivo ai dispositivi
  - In generale riferiremo questi dispositivi come risorse
- Risorse prerilasciabili
  - Possono essere cedute da un processo senza provocarne il fallimento
- Risorse non prerilasciabili
  - Un processo fallisce se viene forzato a cederle prima che ne abbia terminato l'uso

Risorse (3)

- Sequenza di azioni necessarie per usare una risorsa
  1. richiesta della risorsa
  2. uso della risorsa
  3. rilascio della risorsa

- Se la richiesta di acquisizione viene rifiutata il processo richiedente deve aspettare
  - il processo viene messo in stato di attesa oppure
  - può fallire con un corrispondente codice di errore

Introduzione allo stallo

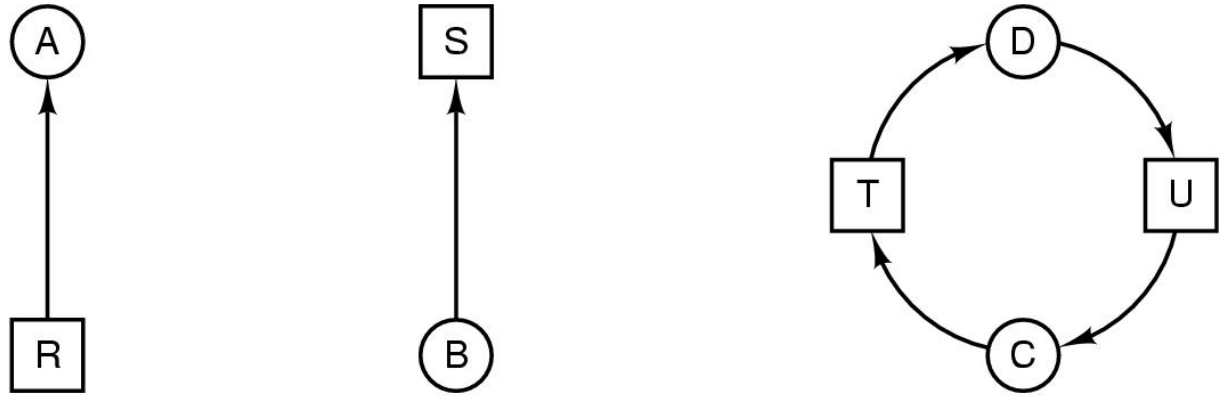
- Definizione formale :
  Un insieme di processi è in stallo se ogni processo nell'insieme sta aspettando un evento che solo un'altro processo nello stesso insieme può generare.
- Generalmente l'evento è il rilascio di una risorsa
- Nessuno dei processi in stallo può …
  - Passare in esecuzione
  - rilasciare risorse
  - Essere riattivato

Quattro condizioni per lo stallo

1. Le risorse sono seriali
   - Ogni risorsa è assegnata ad un processo o è libera
2. Condizione di "hold & wait"
   - Un processo che vincola una risorsa ne può richiedere altre
3. Risorse non prerilasciabili
   - I processi non possono essere forzati a rilasciare in anticipo le risorse acquisite
4. Attesa circolare
   - Almeno due processi
   - Ogni processo aspetta una risorsa occupata da un altro processo in attesa circolare

Modelli per lo Stallo (1)

- Modello con grafi orientati



(a) (b) (c)

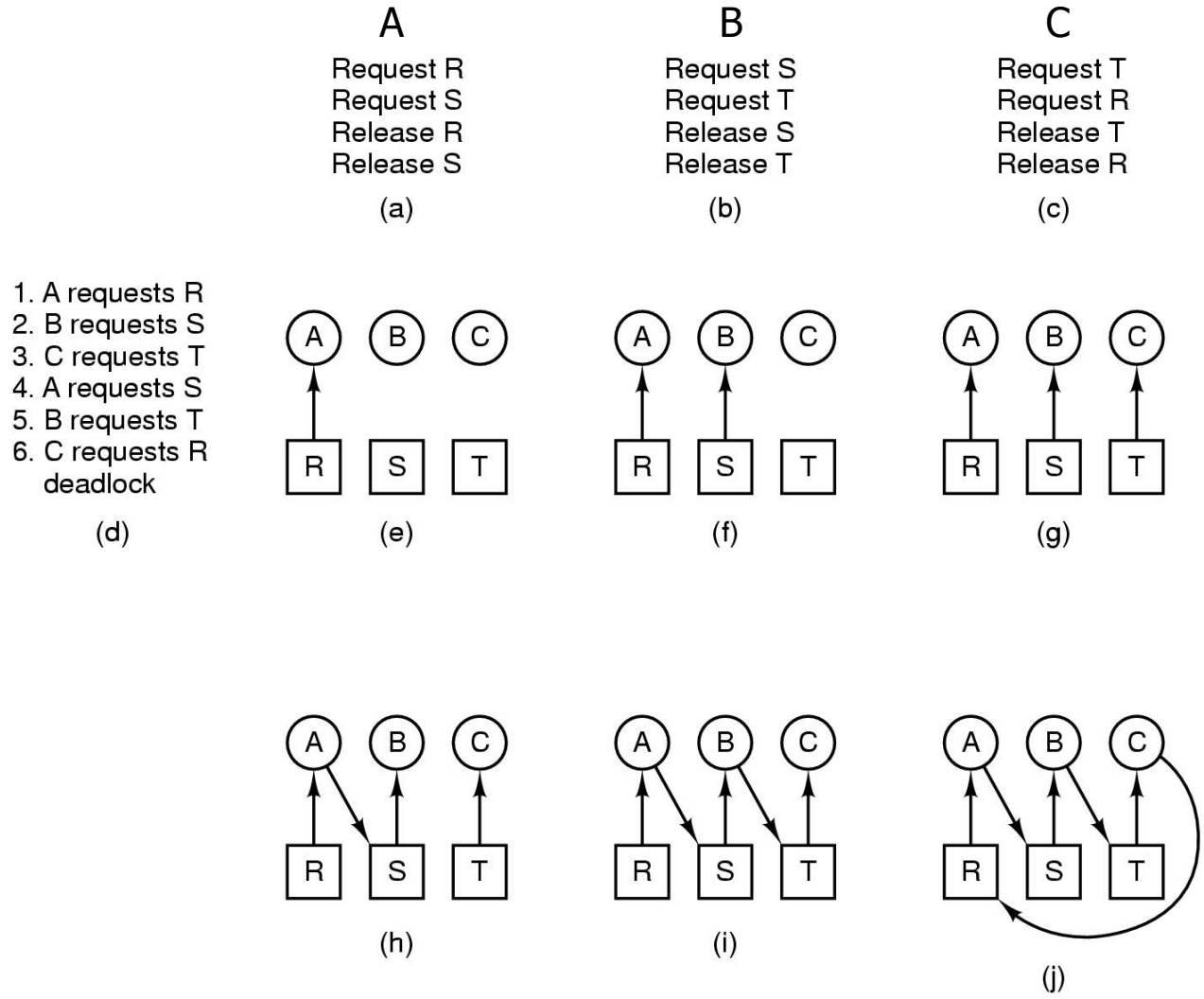
  - La risorsa R è assegnata al processo A
  - Il processo B richiede/aspetta la risorsa S
  - I processi C e D sono in stallo sulle risorse T e U

Modelli per lo Stallo (2)

Strategie per affrontare il problema dello stallo

1. Ignorare il problema
2. Riconoscerlo ed eliminarlo
3. Impedirlo
   - Con specifiche politiche di allocazione
4. Evitarlo
   - Rimuovere una delle quattro condizioni

Modelli per lo Stallo (3)



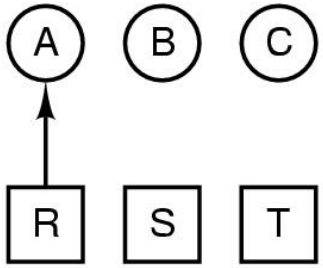
Come avviene lo stallo

Modelli per lo Stallo (5)

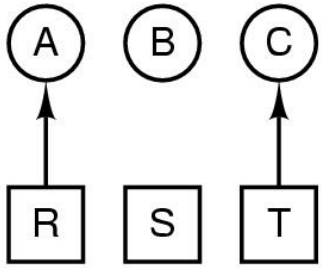
1. A requests R
2. C requests T
3. A requests S
4. C requests R
5. A releases R
6. A releases S

no deadlock

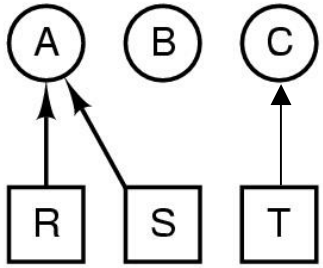
(k)



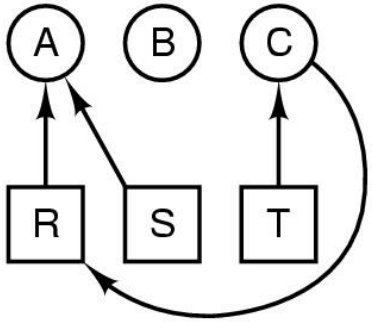
(l)



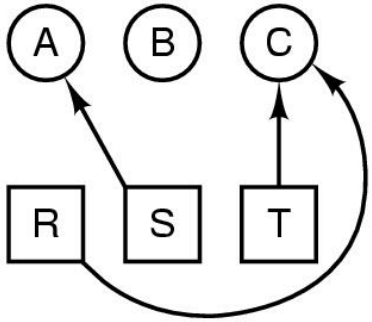
(m)



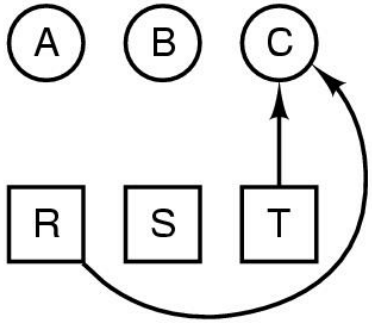
(n)



(o)



(p)



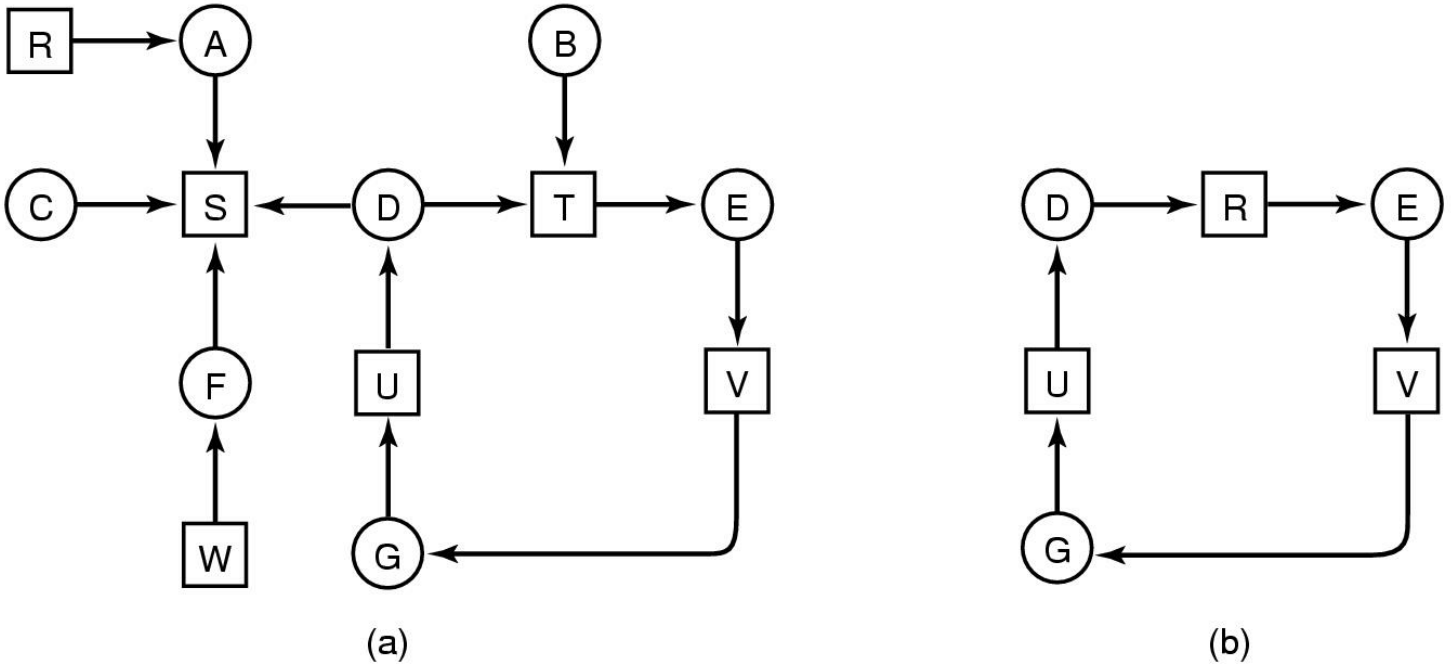
(q)

Impedire lo stallo

L'Algoritmo dello struzzo

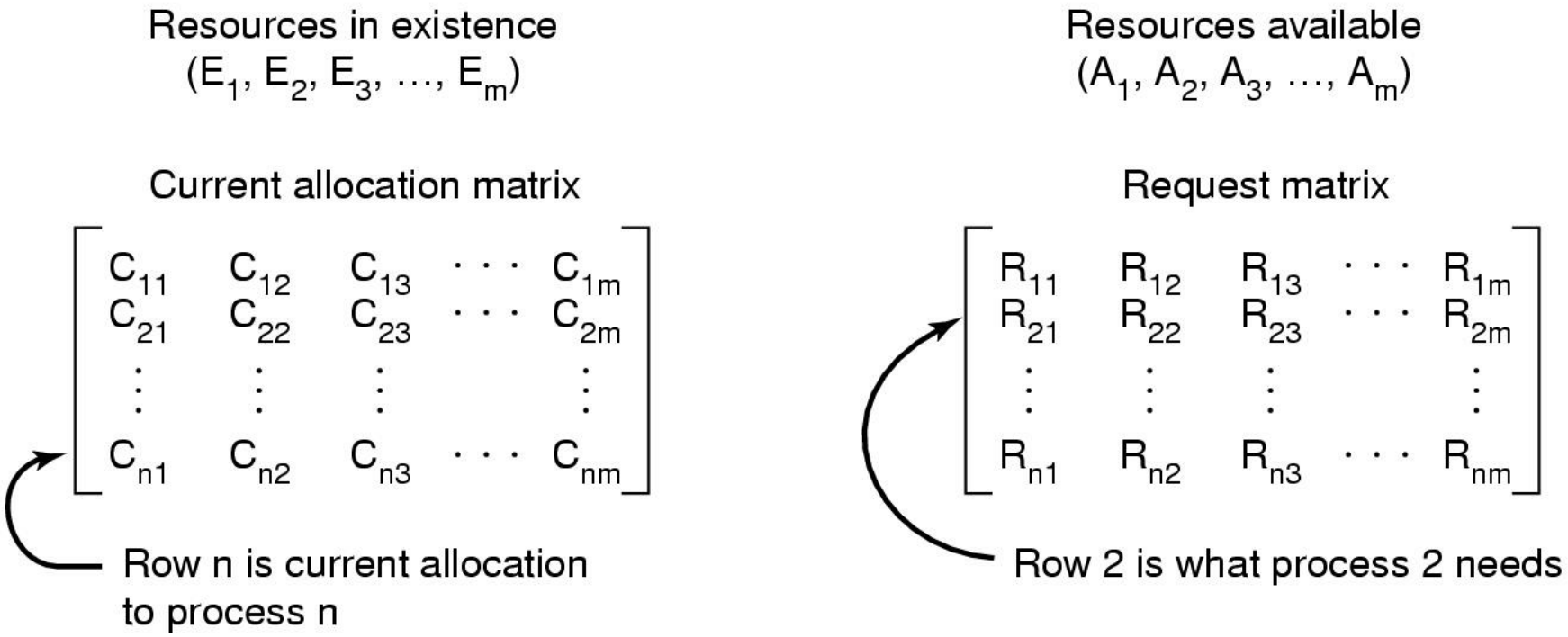
- Pretende che il problema non esista
- E' una soluzione ragionevole se:
  - lo stallo capita raramente
  - il costo per evitare lo stallo è troppo elevato
- UNIX e Windows usano questo approccio
- C'e' un compromesso tra
  - convenienza
  - correttezza

Individuare lo stallo con una risorsa di ogni tipo (1)



- Un ciclo nel grafo denota uno stallo

Individuare lo stallo con più risorse di ogni tipo (1)



Strutture dati necessarie per individuare lo stallo

Individuare lo stallo con più risorse di ogni tipo (2)

Tape drives Plotters Scanners CD Roms

E = (4 2 3 1)

Tape drives Plotters Scanners CD Roms

A = (2 1 0 0)

Current allocation matrix

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Request matrix

$$R = \begin{bmatrix} 2 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 \end{bmatrix}$$

Esempio di strutture dati per l'algoritmo di individuazione dello stallo

Individuare lo stallo con più risorse di ogni tipo (3)

- Notazioni:
  - Sia R_i l'i-esima riga della matrice R
  - Dati due vettori $X=<x_1,...,x_m>$ e $Y=<y_1,...y_m>$, $X \leq Y$ *sse* $x_i \leq y_i$ *per ogni* $i=1,\ldots,m$

Algoritmo di individuazione dello stallo:

1. Inizialmente ogni processo P_i è **non marcato**
2. Per ogni processo non marcato P_i tale che $R_i \leq A$
   - $A=A+C_i$;
   - marca P_i;
3. Se ci sono processi non marcati:

   I processi non marcati sono in stallo

Eliminare lo stallo (1)

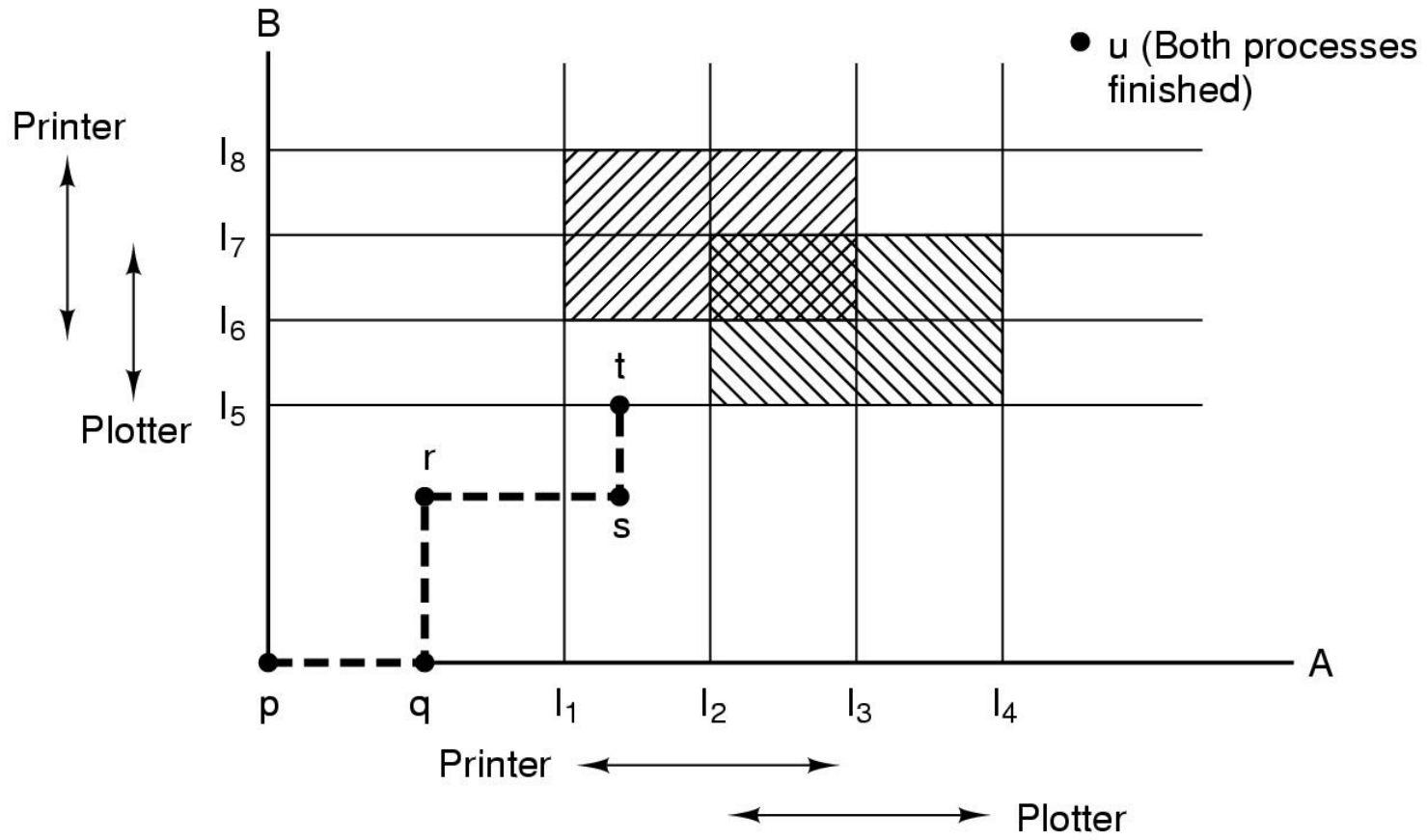
- Tramite prerilascio
  - Forzare il prerilascio di una risorsa da parte di un altro processo
  - dipende dalla natura della risorsa
- Tramite “rollback”
  - Salvataggio periodico dello stato dei processi
  - In caso di stallo:
    - Si ripristina il processo all’ultimo stato salvato
    - Si fa ripartire il processo

Eliminare lo stallo (2)

- Tramite terminazione dei processi
  - È il modo più semplice e drastico per eliminare lo stallo
  - Si forza la terminazione di uno dei processi nel ciclo dello stallo
  - Gli altri processi acquisiscono le risorse del processo terminato
  - Si sceglie un processo che può essere fatto ripartire dall'inizio

Impedire lo stallo

traiettorie nell'uso delle risorse



Traiettorie nell'uso delle risorse da parte di due processi

Stati Sicuri e Insicuri (1)

	Has	Max
A	3	9
B	2	4
C	2	7

Free: 3

(a)

	Has	Max
A	3	9
B	4	4
C	2	7

Free: 1

(b)

	Has	Max
A	3	9
B	0	–
C	2	7

Free: 5

(c)

	Has	Max
A	3	9
B	0	–
C	7	7

Free: 0

(d)

	Has	Max
A	3	9
B	0	–
C	0	–

Free: 7

(e)

Dimostrazione che lo stato (a) è sicuro

Stati Sicuri e Insicuri (2)

	Has	Max
A	3	9
B	2	4
C	2	7

Free: 3

(a)

	Has	Max
A	4	9
B	2	4
C	2	7

Free: 2

(b)

	Has	Max
A	4	9
B	4	4
C	2	7

Free: 0

(c)

	Has	Max
A	4	9
B	—	—
C	2	7

Free: 4

(d)

Dimostrazione che lo stato (b) è insicuro

L'algoritmo del banchiere per risorsa singola

	Has	Max
A	0	6
B	0	5
C	0	4
D	0	7

Free: 10

(a)

	Has	Max
A	1	6
B	1	5
C	2	4
D	4	7

Free: 2

(b)

	Has	Max
A	1	6
B	2	5
C	2	4
D	4	7

Free: 1

(c)

- Tre stati di allocazione delle risorse
  - (a) sicuro
  - (b) sicuro
  - (c) insicuro

Algoritmo del banchiere per risorse multiple (1)

$C=$

Process	Tape drives	Plotters	Scanners	CD ROMs
A	3	0	1	1
B	0	1	0	0
C	1	1	1	0
D	1	1	0	1
E	0	0	0	0

Resources assigned

$R=$

Process	Tape drives	Plotters	Scanners	CD ROMs
A	1	1	0	0
B	0	1	1	2
C	3	1	0	0
D	0	0	1	0
E	2	1	1	0

Resources still needed

E = (6342)
P = (5322)
A = (1020)

- $E=<e_0,\ldots,e_h>$; e_i=numero di risorse di classe i
- $P=<p_0,\ldots,p_h>$; p_i=numero di risorse di classe i occupate
- $A=<a_0,\ldots,a_h>$; a_i=numero di risorse di classe i disponibili ($A=E-P$)

Algoritmo del banchiere per risorse multiple (2)

Inizialmente ogni processo P_j è non marcato

While ∃ processi non marcati {

If (∃ una riga R_j di R tale che $R_j \leq A$) {

Marca il processo P_j;

$A = A + C_j$;

} else termina e segnala lo stallo di tutti i processi non marcati;

}

Termina con successo:

non ci sono processi in stallo

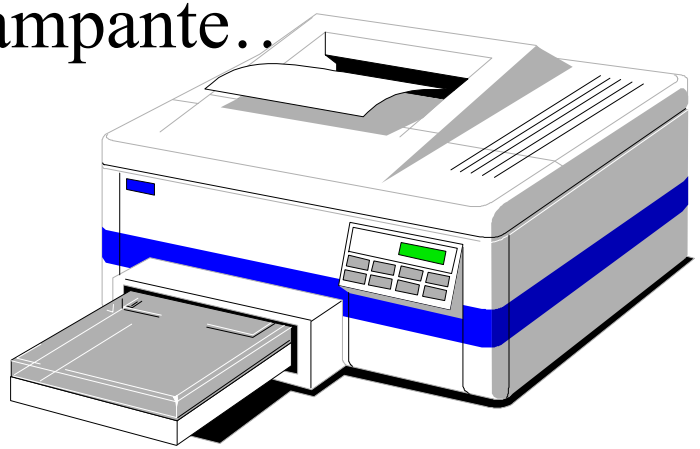
Evitare lo Stallo

Eliminare la condizione di risorse seriali

- Alcuni dispositivi (ad esempio le stampanti) possono essere gestiti con spool
  - Solo il gestore della stampante usa la stampante
  - Quindi lo stallo per la stampante è eliminabile
- Non tutti i dispositivo possono essere gestiti con spool
- Principio:
  - Evitare di assegnare una risorsa quando non strettamente necessario
  - Far in modo che il minor numero possibile di processi possa pretendere una risorsa

Eliminare la condizione di risorse non prerilasciabili

- Questa opzione non è percorribile
- Consideriamo ad esempio un processo che ottiene una stampante
  - … a metà della stampa
  - Viene forzato a cedere la stampante..
  - !!??



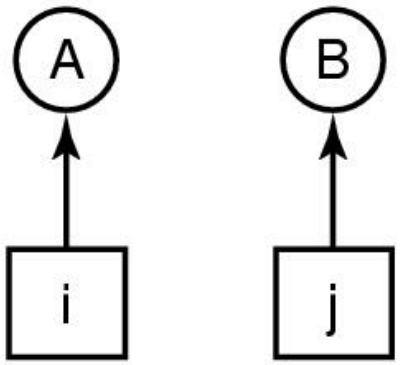
Eliminare la condizione di “hold & wait”

- I processi richiedono tutte le risorse necessarie prima di iniziare
  - Un processo non attende mai per le risorse
- Problemi
  - Può non sapere di quali risorse avrà bisogno
  - Vincola risorse che altri processi potrebbero usare
- Variazioni:
  - Un processo deve prima rilasciare tutte le risorse…
  - … e quindi richiedere immediatamente tutte quelle necessarie

Eliminare la condizione di attesa circolare (1)

1. Imagesetter
2. Scanner
3. Plotter
4. Tape drive
5. CD Rom drive

(a)



(b)

- Risorse ordinate
  - Ad ogni classe di risorse è assegnato un numero
  - Le risorse devono essere richieste secondo l'ordine stabilito
    - Esempio: non si può richiedere una risorsa di indice *j* dopo aver acquisito una risorsa di indice *i* con $i \geq j$
- Grafo delle risorse
  - Si dimostra che è aciclico

Eliminare la condizione di attesa circolare (2)

Condizione	Approccio
Serialità	Usare spool
“hold & wait”	Richiedere tutte le risorse inizialmente
Non prerilasciabilità	Forzare il prerilascio
Attesa circolare	Ordinare le risorse

Sommario degli approcci per evitare lo stallo