Lezione 11

Cenni ai sistemi operativi distribuiti
3. Coordinazione distribuita

- Ordinamento degli eventi
- Mutua esclusione
- Deadlock
- Algoritmi di elezione
- Raggiungimento di un accordo

Ordinamento degli eventi

- Un sistema monoprocessore
  - Unico clock
  - Unica memoria

Possibile stabilire un ordinamento degli eventi

- Un sistema multicomputer/distribuito
  - Non ha memoria comune
  - Non ha unico clock

Impossibile stabilire un ordinamento completo degli eventi

Relazione "verificato prima"

- Riferita ad eventi
- Denotata con $\rightarrow$
- Definizione :

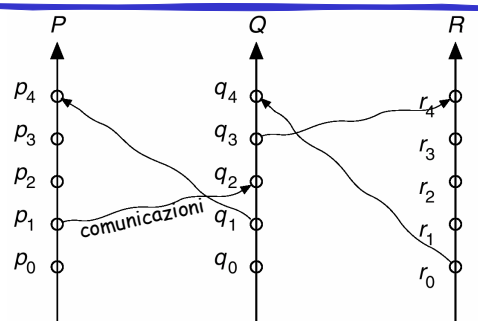
A si e' verificato prima di B se
($A \rightarrow B$)

  - *A* e *B* sono eventi dello stesso processo e A e' stato eseguito prima di B.
  - *A* e' l'evento di inviare un messaggio da un processo e *B* e' l'evento di ricevere lo stesso messaggio da un altro processo.
  - $A \rightarrow C$ e $C \rightarrow B$ (proprieta' transitiva).

Osservazioni

- l'evento A non puo' avvenire prima dell'evento stesso (la relazione "verificato prima" non e' riflessiva)
- se l'evento A si e' verificato prima di B allora non puo' valere il viceversa (la relazione "verificato prima" non e' simmetrica)
- Se non e' possibile stabilire una relazione di "verificato prima" tra due eventi A e B, allora i due eventi devono essere considerati concorrenti

esempio



Eventi in relazione "verificato prima"

$p1 \rightarrow q2$
$r0 \rightarrow q4$
$q3 \rightarrow r4$
$p1 \rightarrow q4$

Eventi concorrenti

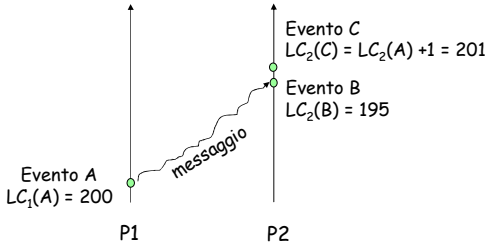
q0 e p2
r0 e q3
r0 e p3
q3 e p3

Implementazione di $\rightarrow$

- Si associa una marca temporale ad ogni evento del sistema e si richiede che per ogni coppia di eventi A e B, se $A \rightarrow B$, allora la marca temporale di A deve essere minore della marca temporale di B.
- In ogni processo Pi la marca temporale e' realizzata mediante il valore di un orologio logico LC_i. L'orologio logico e' un contatore che viene incrementato ad ogni evento all'interno del sistema.
- se un processo riceve un messaggio con marca temporale minore, esso sincronizza il suo orologio logico

esempio

A invio del messaggio da P1
B arrivo del messaggio in P2
C correzione dell'orologio logico in P2



$A \rightarrow B$ implica che $LC_1(A) < LC_2(B)$
Nel nostro caso: $A \rightarrow C$ e $B \rightarrow C$

Mutua esclusione distribuita (DME)

- Il problema della mutua esclusione e' un altro problema che richiede un coordinamento con decisioni che devono essere prese in maniera distribuita
- Assunzioni
  - Il sistema ha n processi; ogni processo risiede in un differente processore.
  - Ogni processo ha una sezione critica che richiede la mutua esclusione.
- richieste
  - Se P_i e' dentro la propria sezione critica allora nessun altro processo deve essere dentro la propria sezione critica

Algoritmo centralizzato

- Uno dei processi nel sistema e' scelto come coordinatore
- Un processo che vuole entrare nella sezione critica manda un messaggio di richiesta al coordinatore
- Il coordinatore decide se puo' entrare, in caso affermativo risponde con un messaggio, altrimenti mette in attesa il processo richiedente
- Quando il processo esce dalla sezione critica, manda un messaggio di rilascio al coordinatore e procede con la sua esecuzione
- Lo schema garantisce la mutua esclusione e evita l'attesa indefinita
- Richiede 3 comunicazioni per ogni accesso alla sezione critica

Approccio distribuito

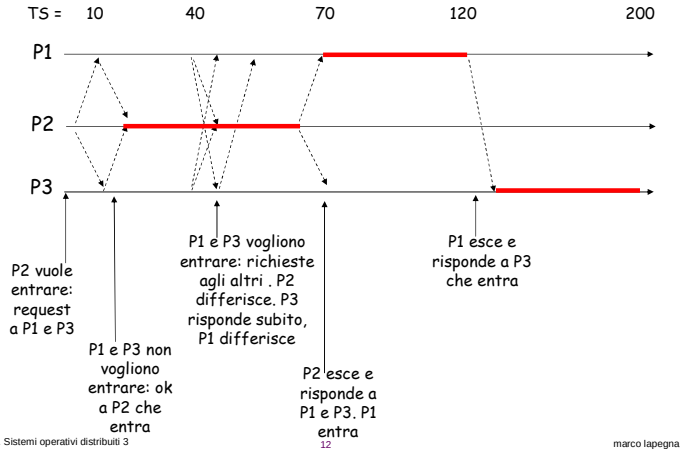
- Basato sul concetto di marca temporale
- Quando un processo P_i vuole entrare nella sezione critica genera una nuova marca temporale *TS*, e manda un messaggio di richiesta (P_i, TS) a tutti gli altri processi del sistema
- Quando un processo P_j riceve un messaggio di richiesta puo'
  - Rispondere immediatamente
  - Differire la risposta
- Quando il processo P_i riceve la risposta da tutti gli altri processi, puo' entrare nella sezione critica e differisce tutte le eventuali richieste in arrivo
- All'uscita dalla sezione critica il processo risponde a tutte le richieste sospese

Algoritmo di Agrawala e Ricart per la mutua esclusione

problema

- da cosa e' definito il comportamento di un processo P_j che riceve un messaggio di richiesta (P_i, TS) ?
  - Se P_j e' nella propria sezione critica allora differisce la risposta a P_i.
  - Se P_j non vuole entrare nella propria sezione critica allora risponde immediatamente
  - Caso limite: P_j aveva gia' inviato una richiesta ed e' in attesa di una risposta. P_j confronta le marche temporali delle due richieste:
    - Se la propria marca temporale e' maggiore di TS allora risponde immediatamente
    - Se la propria marca temporale e' inferiore di TS differisce la risposta ed entra nella sezione critica
    - Se sono uguali da' la precedenza al processo con identificativo inferiore

Esempio: 3 processi P1, P2 e P3



Algoritmo di Agrawala e Ricart

- Aspetti positivi
  - Garantita l'assenza di deadlock
  - Garantita l'assenza di starvation (si entra secondo le TS).
  - Numero di messaggi per entrare nella sezione critica = $2 \times (n - 1)$.
- Aspetti negativi
  - Necessita' di conoscere l'identita' di tutti gli altri processi e comunicazioni globali
  - Se un nodo cade tutto lo schema fallisce.
  - Processi che non devono entrare nella sezione critica sono interrotti frequentemente e inutilmente ➔ adatto a pochi processi.

Metodo con passaggio del contrassegno

- I processi sono organizzati logicamente in una struttura ad anello.
- Uno speciale messaggio, il ***contrassegno***, viene fatto circolare lungo l'anello
  - chi lo possiede può entrare nella sezione critica e trattiene il contrassegno
  - Quando il processo esce, il contrassegno viene rimesso in circolazione.
  - Se il processo non vuole entrare nella sezione critica, lo passa al successivo
- Se l'anello è unidirezionale non vi sono situazioni di attesa indefinita.

Prevenzione dei deadlock

- Alcuni approcci per la prevenzione dei deadlock nei sistemi monoprocessori possono essere utilizzati nei sistemi multicomputer/distribuiti
- Ordinamento globale delle risorse
  - Si assegna un unico identificativo alle risorse.
  - Un processo puo' acquisire la risorse numero i solo se non e' gia' in possesso di una risorsa j con i > j.
  - Semplice da implementare e con basso overhead
- Algoritmo del banchiere
  - Si designa uno dei processi come banchiere, che mantiene le strutture dati (approccio centr.)
  - Gli altri processi contattano il banchiere per sapere se possono acquisire le risorse
  - Anche implementato facilmente ma richiede un maggiore overhead

Schema basato su priorita'

- Ad ogni processo P_i si assegna un unico numero di priorita'
- Es. Nel grafo di attesa $P_i \rightarrow P_j$ indica che P_i ha priorita' maggiore di P_j
- Le priorita' sono usate per decidere se un processo P_i puo' attendere un processo P_j; in caso contrario le richieste di P_i sono annullate
- Lo schema previene il deadlock, in quanto il grafo non puo' contenere cicli
- Puo' provocare attesa indefinita

Schema di attesa/morte basato su marche temporali

- Tecnica senza prelazione.
- Se P_i richiede una risorsa correntemente in possesso di P_j, allora P_i puo' attendere solo se la sua richiesta ha una marca temporale piu' piccola di quella di P_j (P_i e' piu' vecchio di P_j).
- In caso contrario la richiesta di P_i e' annullata
- esempio: P_1, P_2, and P_3 fanno richieste con marche temporali 5, 10, e 15.
  - se P_1 richiede una risorse di P_2, allora P_1 puo' aspettare.
  - se P_3 richiede una risorsa di P_2, allora la richiesta di P_3 e' annullata

Schema di ferita/attesa basato su marche temporali

- Tecnica con prelazione.
- Se P_i richiede una risorsa di P_j, allora P_i puo' attendere solo se la sua richiesta ha una marca temporale maggiore di quella di P_j (P_i piu' giovane di P_j).
- In caso contrario la richiesta di P_j e' annullata
- esempio: P_1, P_2, e P_3 fanno richieste con marche temporali 5, 10, e 15.
  - Se P_1 richiede una risorsa di P_2, allora la risorsa sara' sottratta a P_2.
  - Se P_3 richiede una risorsa di P_2, allora P_3 aspetta

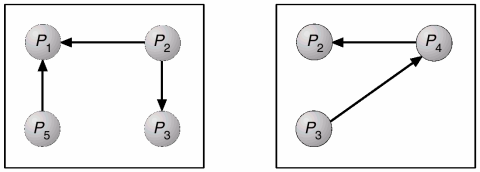
osservazioni

- Si possono avere numerosi annullamenti di richieste prima che un processo possa acquisire la risorsa
- Negli schemi di prevenzione dello stallo si possono sottrarre risorse a processi anche se non c'e' nessuno stallo
- Necessari anche metodi di rilevazione dello stallo

Rilevazione dello stallo

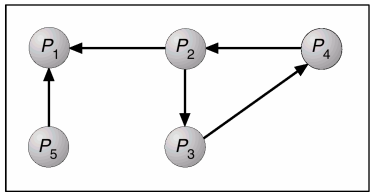
- Il problema principale in un sistema distribuito e' decidere come e dove mantenere il grafo di attesa
- Usualmente si mantiene in ogni sito un grafo di attesa locale.
- I nodi del grafo corrispondono a tutti i processi (anche di altri siti) ai quali e' stata allocata una risorsa locale
- Non e' sufficiente verificare l'assenza di cicli all'interno dei singoli grafi locali

Esempio: 2 siti S1 e S2



site S_1 site S_2

Apparentemente non c'e' deadlock nei due siti

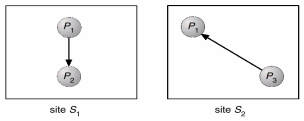


L'unione dei due grafi mostra la presenza di un ciclo

Approccio centralizzato

- L'unione di tutti i grafi e' mantenuta da un singolo processo: il coordinatore per il rilevamento
- differenti opzioni per la costruzione del grafo di attesa :
  1. Ogni volta che si inserisce o si rimuove un arco in uno dei grafi locali (dispendioso).
  2. periodicamente, ad es. quando un numero minimo di cambi sono stati fatti nei grafi locali.
  3. Solo quando il coordinatore decide di verificare la presenza di un deadlock
- Un problema: falsi cicli di deadlock (deadlock fantasma)

Esempio : opzione 3 (rilevazione a richiesta)

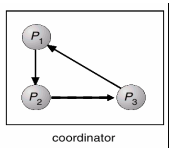


site S_1 site S_2

In questo istante il coordinatore chiede i grafi locali ai siti (non c'e' stallo)

- Supponiamo che avvenga la seguente sequenza di eventi
  - Il messaggio del coordinatore arriva a S1 che risponde (P1 → P2)
  - P2 rilascia le risorse in S1 e chiede una risorsa posseduta da P3 in S2 (P2 → P3). Non c'e' stallo e in S2 e' presente P2 → P3 → P1.
  - Il messaggio del coordinatore arriva a S2 che risponde P2 → P3 → P1
  - Il coordinatore costruisce il grafo globale

Deadlock fantasma



coordinator

Soluzione

- Per evitare false segnalazioni di stallo, le segnalazioni dai vari siti sono marcate con marche temporali

Gestione dei grafi locali

- Quando un processo P_i, nel sito S1, richiede una risorsa dal processo P_j, nel sito S2, il messaggio e' accompagnato da una marca temporale TS
  - L'arco $P_i \rightarrow P_j$ con marca temporale TS e' inserito nel grafo di attesa di S1.
  - L'arco e' inserito anche nel grafo di S2 solo se S2 non puo' soddisfare immediatamente la richiesta

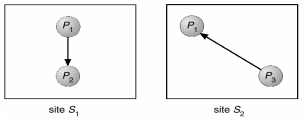
Gestione del grafo globale del coordinatore

1. Il controllore invia le richieste ai siti per acquisire i grafi locali.
2. Al ricevimento ogni sito invia il proprio grafo locale

3. Il grafo globale e' costruito a partire da quelli locali:
   (a) Il grafo globale contiene un vertice per ogni processo del sistema
   (b) Nel grafo globale e' presente l'arco $P_i \rightarrow P_j$ se e solo se
      - E' presente l'arco $P_i \rightarrow P_j$ senza marca temporale in uno dei grafi locali oppure
      - L'arco $P_i \rightarrow P_j$ con la marca temporale TS e' presente in piu' di un grafo locale

Se il grafo globale contiene un ciclo ⇒ deadlock.

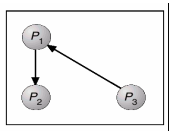
Esempio



In questo istante il coordinatore chiede i grafi locali ai siti (non c'e' stallo)

- Supponiamo che avvenga la seguente sequenza di eventi
  - Il messaggio del coordinatore arriva a S1 che risponde (P1 → P2)
  - P2 rilascia le risorse in S1 e chiede una risorsa posseduta da P3 in S2 (**P2 → P3 in entrambi i siti con TS**). Non c'e' stallo . In S2 e' presente P2 → P3 → P1.
  - Il messaggio del coordinatore arriva a S2 che risponde P2 → P3 → P1
  - Il coordinatore costruisce il grafo globale dove P2 → P3 non viene inserito (non e' presente in entrambi i grafi, perche S1 aveva gia' risposto)

Non c'e' deadlock fantasma



Approccio distribuito

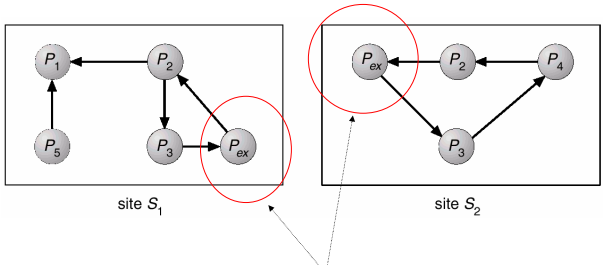
- Tutti i siti condividono la responsabilita' di rilevare lo stallo
- Ogni sito costruisce un grafo di attesa locale che e' parte del grafo globale

- Ad ogni grafo locale si aggiunge un nodo P_{ex} per fare riferimento a risorse di altri siti.
  - Se un grafo locale contiene un ciclo che non involve P_{ex}, allora il sistema e' in uno stato di deadlock (lo stallo e' interno al sito)
  - Se un grafo locale contiene un ciclo che involve P_{ex} allora c'e' possibilita' di deadlock.

problema

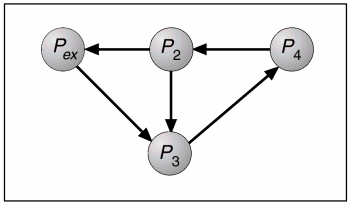
- Come fare a vedere se c'e' effettivamente uno stallo?

- Esempio: 2 siti S1 e S2



soluzione

- Se S1 trova un ciclo che coinvolge P_{ex} nel sito S2 allora manda le informazioni del proprio grafo a S2, che aggiorna il proprio grafo con le nuove informazioni



site S_2

- Ciclo che non coinvolge P_{ex} ➔ deadlock

osservazioni

- Situazione analoga se S2 manda a S1 le sue informazioni
- Nel caso di piu' siti puo' essere necessario passare le informazioni da sito a sito
- In un numero di passi al piu' pari al numero di siti viene rilevato lo stallo

Algoritmi di elezione (leader election)

- Molti algoritmi alla base dei sistemi operativi distribuiti richiedono un coordinatore (o master, leader)
  - Per la mutua esclusione
  - Per mantenere grafi globali
  - Controllo di dispositivi di I/O
- In caso di caduta del processo o di interruzione della comunicazione e' necessario

Eleggere un nuovo coordinatore

- Algoritmi basati sul concetto di priorita' (per semplicita' priorita' di Pi = i)
- Il coordinatore e' il processo con priorita' maggiore
- Si assume nel seguito una corrispondenza 1-1 tra processi e siti

Algoritmo dello spaccone

Se un processo **Pi** invia una richiesta e il coordinatore non risponde in un intervallo di tempo **T**, allora **Pi** assume che il coordinatore non è piu' attivo e cerca di eleggere se stesso come nuovo coordinatore :

Pi invia un messaggio a ogni processo **Pj** con un numero di priorità più alto (j > i) ed aspetta per un tempo **T'** la risposta da tutti gli altri processi

- Se non riceve risposta, **Pi** assume che tutti i processi **Pj** non sono attivi ed elegge se stesso come coordinatore. **Pi** fa partire una copia del coordinatore e manda un messaggio a tutti i processi con priorità minore di **i**, informandoli che lui è il nuovo coordinatore
- Se riceve una risposta, **Pi** aspetta di ricevere un messaggio che lo informi quale processo con priorità più alta della sua è stato eletto.

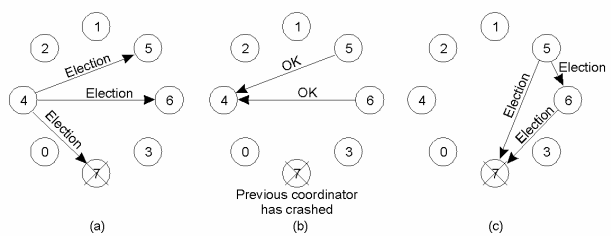
Algoritmo dello spaccone (Cont.)

- Un processo **Pi** in ogni istante puo' ricevere un messaggio da **Pj** del tipo
  - **Pj** e' il nuovo coordinatore (j > i). **Pi** registra tale informazione
  - **Pj** ha iniziato l'elezione di un nuovo coordinatore (j < i). **Pi** manda una risposta a **Pj** e inizia il suo algoritmo di elezione

- Quando un processo caduto viene ripristinato , inizia immediatamente l'algoritmo di elezione
- Se non ci sono processi attivi con priorita' maggiore, il processo ripristinato si impone come coordinatore, anche se gia' in esecuzione un altro coordinatore

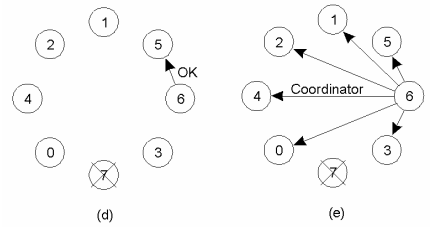
- Richiede al piu' n^2 messaggi per l'elezione

Esempio



a) Il processo 4 si accorge che 7 non risponde e indice le elezioni
b) I processi 5 e 6 rispondono e dicono a 4 di fermarsi
c) Ora 5 e 6 indicono le elezioni

Esempio

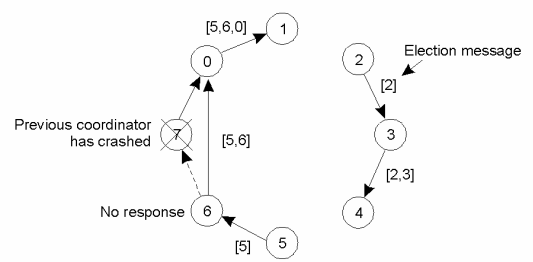


d) Il processo 6 dice a 5 di fermarsi
e) Il processo 6 non riceve risposte, vince e lo comunica a tutti

Algoritmo ad anello

- I processi hanno un unico ID e sono organizzati logicamente in un anello
- Ogni processo conosce i suoi vicini
  - Il coordinatore e' il processo con ID più alto
- Si invia un messaggio di ELECTION all'insieme di nodi più vicini che sono ancora vivi
  - Si invia il messaggio ad ogni successore finché viene trovato un nodo vivo
- Ogni processo marca il proprio ID sul messaggio
- Chi inizia raccoglie il nodo con l'ID più alto e invia un messaggio comunicando il coordinatore
- Vi possono essere elezioni multiple
  - Si ha un consumo notevole di banda

Algoritmo di elezione ring: esempio



- 7 è in crash
- 6 non ha risposta e indice una elezione
- 1 vota 6 e 4 vota 3

Raggiungimento di un accordo

- In alcune applicazioni un insieme di processi devono accordarsi su un valore comune (es. Algoritmi di elezione)
- Il raggiungimento dell'accordo puo' non avvenire a causa di :
  - Comunicazione inaffidabile
  - Processi inaffidabili
    - I processi inviano messaggi confusi o incorretti.
    - I processi possono cercare di distruggere l'integrita' dello schema.

Esempio di comunicazione inaffidabile

- P_i nel sito A, invia un messaggio a P_j nel sito B, e vuole conferma che il messaggio e' stato ricevuto
- Schema time-out.
  - P_i invia anche un intervallo di time out entro il quale vuole una risposta da P_j.
  - Quando P_j riceve il messaggio risponde subito a P_i ma non e' sicuro che la risposta arrivi. P_j oltre alla risposta invia un intervallo di time out
  - Quando P_i riceve il messaggio risponde subito a P_j ma non e' sicuro che la risposta arrivi. P_i oltre alla risposta invia un intervallo di time out
  - E cosi' via…

In un ambiente distribuito, due processi non possono accordarsi in maniera completamente non ambigua sul loro rispettivo stato

Problema dei generali bizantini

- N generali (i processi) assediano un campo nemico e possono comunicare solo attraverso messaggeri (messaggi)
- Problema: i generali possono raggiungere un accordo sul momento in cui attaccare tutti insieme?
- Risposta: ogni generale non puo' essere sicuro che gli altri attaccheranno nello stesso istante
  - I messaggeri possono essere catturati
  - Alcuni generali possono essere dei traditori