UNIVERSITÀ CA' FOSCARI DI VENEZIA
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Specialistica in Informatica

Corso di Sistemi Distribuiti
Approfondimento

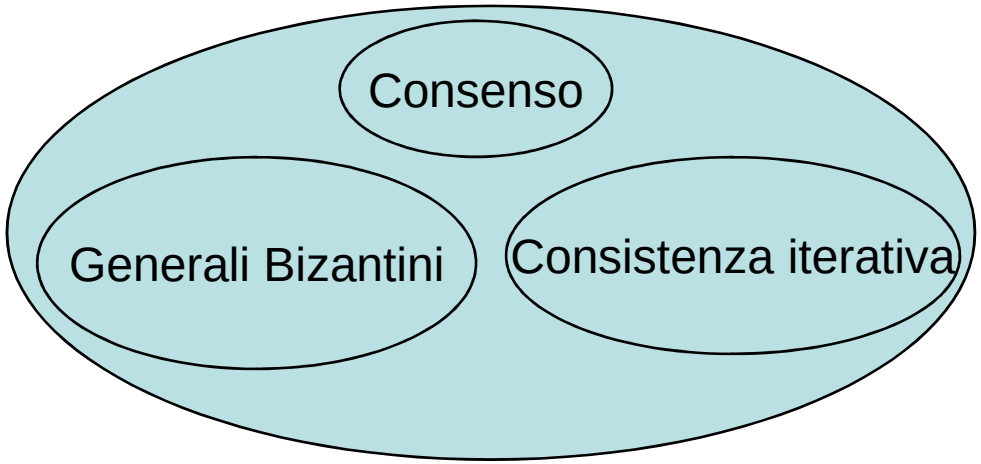
Problema del consenso e relativi problemi

Studente: Marco Lionello

Anno Accademico 2005-2006

Consenso e problemi correlati

Problemi di agreement



• Problemi di “agreement”

• Trovare un’ accordo su di una variabile dopo che uno o più processi propongono un valore per quella variabile”

•Esempi:

- •Procedere o non procedere su una determinata operazione
- •Transazioni bancarie
- •Mutua esclusine
- •Multicast totale ordinato

Problema del consenso

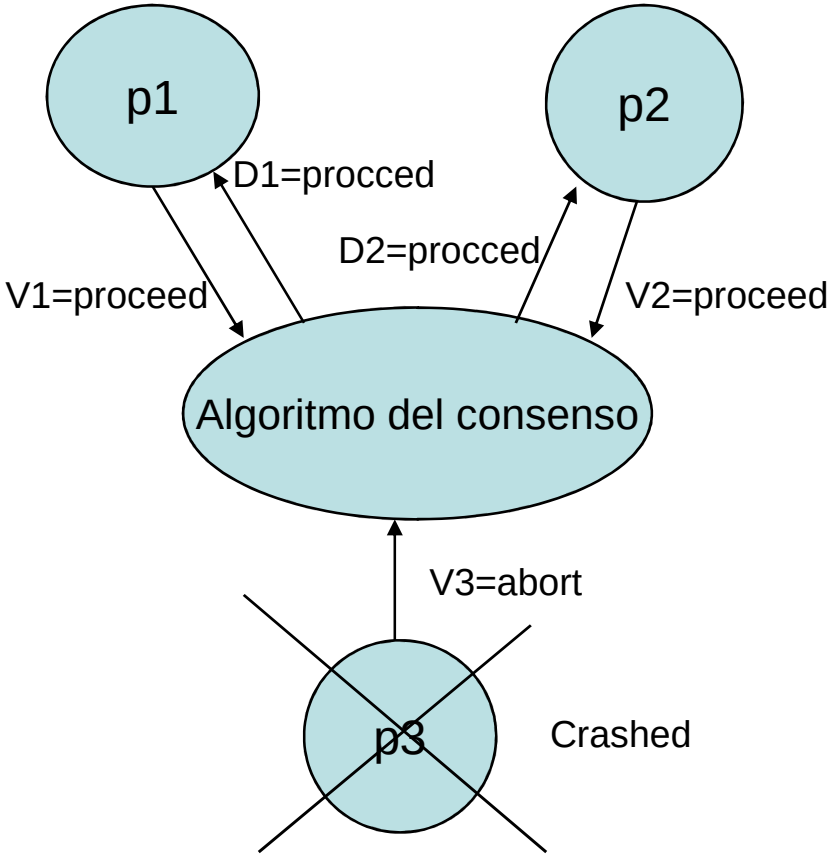
- Ogni processo p_i è all'inizio nello stato undecided e propone un valore v_i appartenente a D

Condizioni fondamentali nel consenso:

•Terminazione: Ogni processo corretto pone la sua variabile di decisione

•Accordo: La variabile di decisione di tutti i processi corretti è la stessa

•Integrità: se tutti i processi corretti propongono lo stesso valore, allora ogni processo corretto nello stato di decisione ha scelto quel valore



NOTA: possibile introduzione di una soglia sulla nozione di integrità

Problema dei generali bizantini

- Tre o più generali devono accordarsi sull'attaccare o sul ritirarsi
- Comandante dà i comandi, i tenenti devono decidere sul da farsi
- Uno o più generali può essere perfido:
  - Se è il comandante dice a un tenente di attaccare e all'altro di ritirarsi
  - Se è un tenente comunica ai peers ad alcuni di attaccare ad altri di ritirarsi

Differenza dal consenso: un processo distinto fornisce un valore che gli altri devono concordare, invece che tutti propongano il valore.

Condizioni fondamentali nel problema del generale bizzantino:

Terminazione: Ogni processo corretto pone la sua variabile di decisione
Accordo: La variabile di decisione di tutti i processi corretti è la stessa
Integrità: se il comandante è corretto, allora tutti i processi corretti decidono sul valore che il comandante ha deciso

Nota: l'integrità implica l'accordo solo se il comandante è corretto

Problema della consistenza iterativa

- Variante del consenso in cui tutti i processi propongono un valore
- Goal dell'algoritmo è per i processi corretti di accordarsi in un vettore di valori uno per ogni processo "decision vector"
- Esempio ottenere le stesse informazioni riguardo ai rispettivi stati

Condizioni fondamentali nel problema del generale bizantino:

- Terminazione: Ogni processo corretto pone il suo "decision vector"
- Accordo: il "decision vector" di tutti i processi corretti è lo stesso
- Integrità: se p_i è corretto, allora tutti i processi corretti decidono su v_i come i-esimo componente del loro vettore.

Relazionare i tre problemi (1/2)

- Frammentazione dei problemi → sincroni
- Frammentazione dei problemi → asincroni
- Soluzione dei problemi C, BG, IC:
  - Ci(v1,v2,…,vn) ritorna la “decision value” di p_i nella soluzione di una esecuzione del problema del consenso, dove v_i sono i valori che i processi hanno proposto
  - BGi(j,v) ritorna la “decision value” di pi nella soluzione di una esecuzione dl problema dei generali bizantini, dove p_j il comandante propone il valore v
  - ICi(v1,…,vn)[j] ritorna il j-esimo valore nel “decision vector” di p_i nella soluzione dell’esecuzione del problema della consistenza iterativa, dove v1,….,vn sono valori che i processi propongono

Relazionare i tre problemi (2/2)

- IC da BG: Costruiamo una soluzione a IC da BG facendo eseguire BG N volte una per ogni processo pi usandola come comandante:
  ICi(v1,…,vn)[j]=BGi(j,vj) (i,j=1….N)

- C da IC: Per il caso in cui majority() dei processi è corretta, si costruisce una soluzione di C da IC eseguendo IC per produrre un vettore di valori a ogni processo, che applicato alla funzione appropriata nel vector's value per derivare un singolo valore
  Ci(v1,…,vn)=majority(ICi(v1,..,vn)[1],....., ICi(v1,..,vn)[N]) per ogni i

- BG da C: Costruiamo una soluzione di BG da C come descritto qui sotto:
  - Il comandante pj manda una proposta di valore v a se stesso e a ogni altro processo
  - Tutti i processi lanciano C con i valori v1,v2,….,vn che ricevono (pj potrebbe essere caduto)
  - I processi derivano BGi(j,v)=C(v1,…..,vn) per ogni i

Algoritmo (1/2)

- In sistemi con crash risolvere il problema è equivalente a isolvere un problema di multicast certo ordinato.
- Implementazione del consenso con RTO-multicast → diretto

Algoritmo

- Ogni processo fa RTO-Multicast di (g,vi) per f+1 giri
- Ogni processo pi sceglie di=mi dove mi è il primo valore che pi trasporta

- La terminazione, accordo, integrità garantiti dal RTO-Multicast
- Dimostrazione di Chandra and Toueg [1996]

Algoritmo (2/2)

- Al max f difetti
- Garanzia che al passo f+1 ci sia accordo
- Attiya e Welch [1998]
- Sistema sincrono → durata del giro settata da un timeout

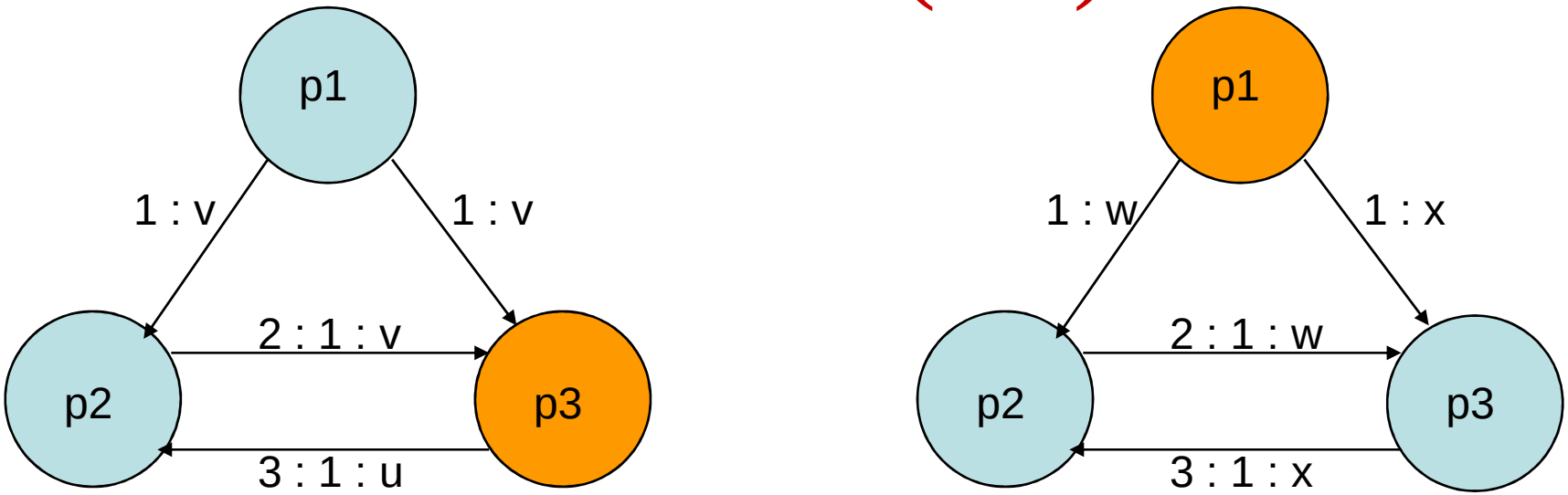
Dimostrazione

- Per assurdo due processi alla fine dell'algoritmo differiscono pi e pj
- Unica spiegazione crash di un terzo processo che gestiva la spedizione di v da pi a pj
- Crash nel giro precedente, al max un crash per giro
- Ma f crash e f+1 giri → Assurdo!!!

Generale bizantino in sistema sincrono (1/2)

- Presenza di difetti arbitrari (messaggi senza valore, valore sbagliato)
- Processi corretti capiscono quando quando c'è assenza di messaggio ma non possono concludere un crash
- Assunzione canale di comunicazione privato
- Processi esaminano più messaggi per trovare eventuali inconsistenze
- Nessun processo può inniettare messaggi maliziosi
- Lamport[1982] non esiste soluzione nel caso di tre processi con uno fallimentare.
- Generalizzazione N<=3f

Generale bizantino in sistema sincrono (2/2)

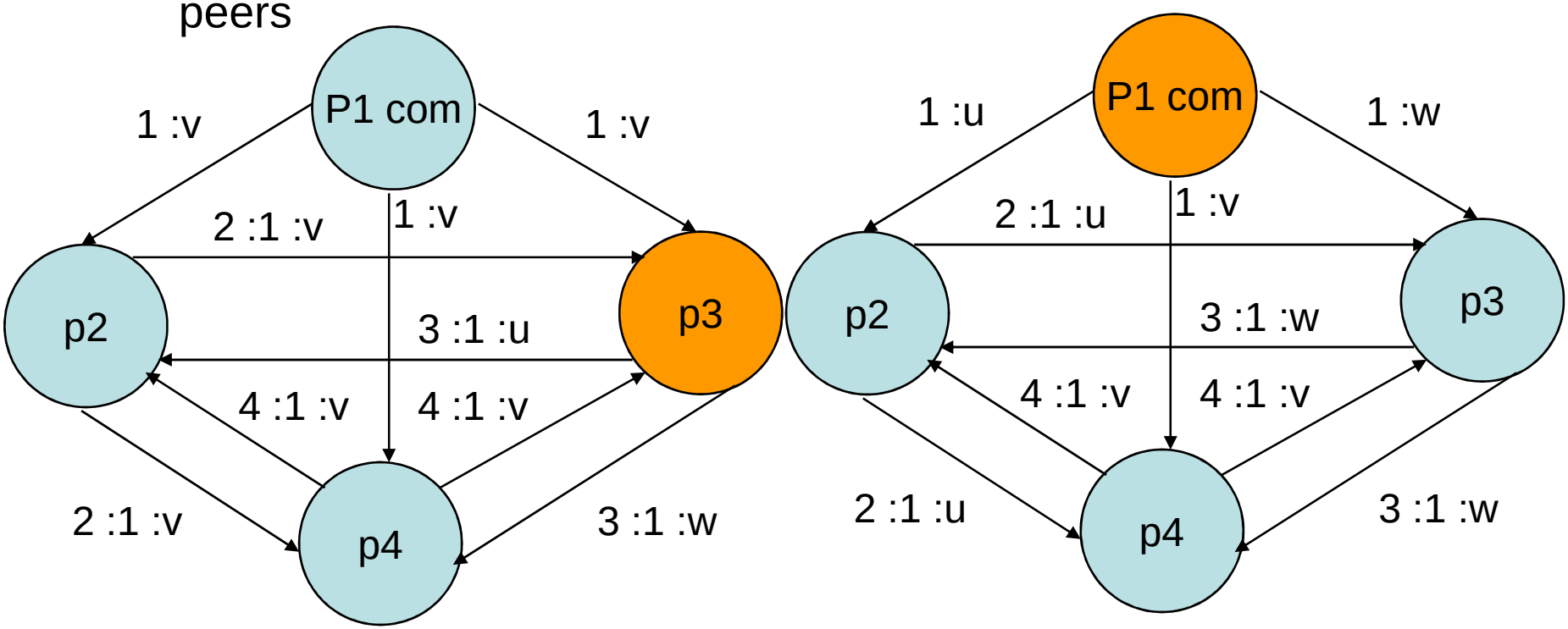


• dimostrazione Pease[1980]. L'algoritmo bizantino può essere raggiunto per i tre generali, con uno di questo difettoso, se il generale firma i suoi messaggi.
• Soluzione sse N>3f

Soluzione con un processo caduto(Istanza Pease N=4 f=1)

Due passi:

1. Nel primo giro ogni generale manda un valore ad ogni tenente
2. Nel secondo ogni tenente manda il valore che ha ricevuto ai suoi peers



Conclusioni

Misure di efficienza:

- Giri di messaggi → Tempo di esecuzione
- Numero di messaggi scambiati → Bandwith

- Lamport numero di giri = f+1 Fisher and Lynch [1982]
- Numero di messaggi O(N^f+1)

- Dolev and Strong

- Algoritmi applicabili quando la minaccia è grande
- Uso di firma elettronica

Bibliografia

- Pease, M., Shostack,R. and Lamport, L.(1980). Reaching agreement in the presence of faults. *Journal of the ACM*, Vol27, No. 2, April, pp. 228-34
- Dolev, D. and Strong, H. (1983). Authenticated algorithms for byzantine agreement. *SIAM Journal of Computing*, Vol. 12, No. 4, pp. 656-66
- Fisher, M. and Lynch, N. (1982). A lower bound for the time to assure interactive consistency. Inf. Process. Letters, Vol. 14, No 4,June, pp183-6
- Fisher, M.,Lynch, N. and Paterson, M. (1985) Impossibility of distribuited consensus with one faulty process. Journal of ACM, Vol. 32, No. 2 Apr., pp.34
- Attiya, H. and Welch, J. (1998). *Distribuited Computing Fundamentals, Simulations and Advanced Topics*. McGraw-Hill.
- Chandra, T. and Toueg, S. (1996). Unrealable failure detectors for reliable distributed system. *Journal of the ACM*, Apr., pp. 374-82
- Coulouris, G. and Dollimore, J. And Kindberg, T. (2005). *Distribuited System Concepts and Design*, 4th edition, Addison Wesley Masson