Università degli Studi di Padova

Sincronizzazione

SCD

Anno accademico 2003/4
Corso di Sistemi Concorrenti e Distribuiti

Tullio Vardanega, tullio.vardanega@math.unipd.it

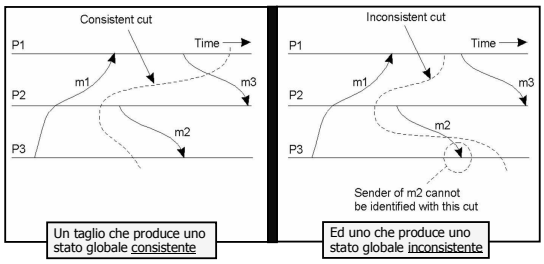
Stato del sistema - 1

- **Stato globale di un sistema distribuito**
  - **Stato locale di ciascuno dei suoi processi**
    - Solo ciò che è considerato importante ai fini dello stato globale
  - **L'insieme dei messaggi in transito (non ancora consegnati)**
- **Conoscere lo stato globale consente di**
  - **Verificare se il sistema è globalmente attivo oppure no (nessun messaggio in transito)**
  - **Nel 2o caso, analizzarne le cause: normale terminazione oppure stallo?**

Stato del sistema - 2

- **Istantanea distribuita (*distributed snapshot*)**
  - **Riflette uno stato globale consistente come potrebbe essere stato**
    - Esempio di stato inconsistente: P ha ricevuto un messaggio da Q, il cui invio non risulta dallo stato
  - **Rappresenta un "taglio" (*cut*) nell'evoluzione temporale dei processi del sistema**
    - Fissa ciò che appartiene allo stato globale e ciò che ne è escluso
    - È il "percorso" del taglio (causato dall'algoritmo usato) che determina la consistenza dello stato

Stato del sistema - 3



Stato del sistema - 4

- **Istantanea distribuita: algoritmo 1/2**
  - **Sistema visto come un insieme di processi connessi da canali di comunicazione punto a punto unidirezionali**
  - **Qualunque processo può iniziare la cattura dello stato**
    - Più istantanee possono trovarsi in corso simultaneamente
  - **Il processo iniziatore salva il proprio stato ed invia un *marker* su ogni suo canale in uscita, richiedendo al destinatario di partecipare alla cattura dello stato**
    - Il marker identifica il processo iniziatore e la versione dell'istantanea

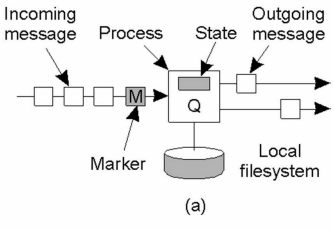
Stato del sistema - 5

- **Istantanea distribuita: algoritmo 2/2**
  - **Il processo che riceve un *marker* da un suo canale C**
    - Se non ha ancora salvato il suo stato locale, lo salva ed invia il *marker* su tutti i suoi canali in uscita
    - Se già lo ha salvato, inizia a salvare lo stato del canale C, ossia l'insieme dei messaggi ricevuti su C a partire dall'ultimo salvataggio del sistema
  - **Un processo ha completato la sua parte dell'algoritmo quando abbia trattato tutti i *marker* ricevuti da tutti i suoi canali in ingresso**
  - **Quando tutti i processi coinvolti dall'iniziatore hanno completato, l'iniziatore può raccogliere lo stato globale**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

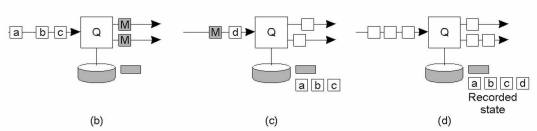
Stato del sistema - 6



Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Stato del sistema - 7



... Salva il proprio stato locale ed invia il *marker* su ciascuno dei suoi canali in uscita ...

... e comincia a salvare anche lo stato del suo canale in ingresso ...

... All'arrivo del successivo *marker* sul suo canale in ingresso ha completato il suo contributo alla cattura dello stato globale

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Esempio d'uso - 1

- **Il processo Q che ha ricevuto un *marker* per la prima volta ne considera il mittente M sul canale in ingresso come suo predecessore**
  - **Q è dunque il successore di M**
- **Quando Q completa il suo contributo invia ad M il messaggio FINITO**
- **Lo stato globale è pronto quando l'iniziatore ha ricevuto messaggi FINITO da tutti i suoi successori**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Esempio d'uso - 2

- **Il processo Q invia il messaggio FINITO se e solo se**
  - **Tutti i suoi successori hanno inviato il messaggio FINITO**
  - **Q non ha ricevuto alcun messaggio successivo al completamento del suo stato**
- **Altrimenti, Q invia il messaggio CONTINUA**
- **L'iniziatore invia un nuovo *marker* finché non riceva solo messaggi FINITO**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Elezione del coordinatore - 1

- **La presenza di un coordinatore facilita la costruzione di algoritmi distribuiti**
- **Eleggere un coordinatore richiede un accordo distribuito**
  - **L'obiettivo dell'algoritmo di elezione è assicurarne la terminazione con l'accordo di tutti i partecipanti**
- **Prerequisiti**
  - **Tutti i processi del sistema hanno un identificatore unico ed ordinabile (maggiore, minore)**
  - **Ogni processo del sistema conosce gli identificatori di tutti gli altri processi**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

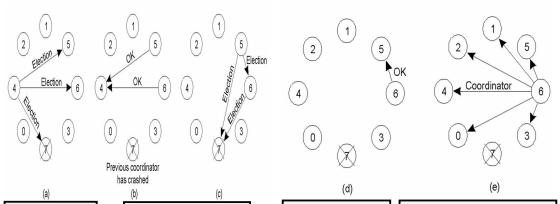
Elezione del coordinatore - 2

- **L'algoritmo del "bullo"**
  - **Il processo P che rilevi l'assenza del coordinatore promuove una nuova elezione**
    - P invia un messaggio ELEZIONE a tutti i processi di identificatore maggiore
    - Se nessuno risponde, P si proclama vincitore e diventa coordinatore
    - Un processo che riceva il messaggio ELEZIONE da un processo di identificatore minore risponde con il messaggio OK e ne rileva l'elezione
    - Se P riceve un messaggio OK ha finito il suo lavoro
  - **Un processo appena (ri-)creato non conosce il coordinatore e dunque promuove una elezione**
  - **L'algoritmo designa sempre come coordinatore il processo in vita con identificatore maggiore**
    - Il vincente informa tutti i processi del sistema che hanno un nuovo coordinatore

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Elezione del coordinatore - 3



Il processo **4** inizia una nuova elezione

I processi **5** e **6** rilevano l'elezione in parallelo

Il processo **6** rileva l'elezione del processo **5**

Il processo **6** non conosce processi di identificatore maggiore, quindi si proclama coordinatore

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Mutua esclusione - 1

- **Algoritmo centralizzato: facile, ma fragile**
  - **Tutte le richieste di accesso a risorse in mutua esclusione vengono inviate ad un coordinatore centrale**
    - Richiesta ENTER
  - **Se la risorsa è libera, il coordinatore informa il mittente che l'accesso è consentito**
    - Notifica GRANTED
  - **Altrimenti il coordinatore accoda la richiesta (FIFO) informando il mittente che l'accesso non è consentito**
    - Notifica DENIED (oppure nessun risposta per richieste sincrone)
  - **Quando l'utente rilascia la risorsa informa il coordinatore**
    - Notifica RELEASED
  - **Il coordinatore preleva la prima richiesta in attesa ed invia al mittente la notifica di accesso**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Mutua esclusione - 2

- **Algoritmo distribuito** 1/2
  - **Richiede infrastruttura di comunicazione affidabile**
  - **Il processo P che voglia accesso esclusivo ad una risorsa R costruisce un messaggio M (GRANT?) contenente <P, R, C> e lo invia a tutti i processi del sistema**
    - C = ora locale di P
  - **Il processo che riceva M**
    - Se non sta usando R e non vuole usarla, risponde OK
    - Se invece sta usando R, non risponde ed accoda M presso di se
    - Se sta per accedere R ma non lo ha ancora fatto, confronta C con la sua ora locale: il valore più basso vince

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Mutua esclusione - 3

- **Algoritmo distribuito** 2/2
  - **P aspetta di aver ricevuto l'OK di tutti i processi**
    - Quando ciò avviene, P accede ad R in mutua esclusione
  - **Quando P rilascia R risponde OK a tutti i processi mittenti di richieste accodate presso di se e le rimuove dalla coda**
- **Da 1 SPF (*single-point failure*) nel coordinatore ad 1 SPF per ogni processo**
  - **Assenza di risposta interpretata come divieto d'accesso**
  - **I processi devono sempre rispondere (risposta DENIED) → in assenza di risposta il richiedente deve riprovare con *time-out* (sospendendo tra il primo DENIED ed il successivo OK)**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Mutua esclusione - 4

- **Algoritmo a gettone circolante (*token ring*)**
  - **Processi collegati in sequenza circolare ordinata e punto a punto, lungo la quale deve transitare un gettone**
  - **Il processo in posizione 0 riceve per primo il gettone**
  - **Il possesso del gettone consente al processo di accedere 1 risorsa in mutua esclusione, per poi passare il gettone al suo vicino**
  - **Se il processo non ha immediato bisogno di risorse passa subito il gettone al vicino**
    - il vicino conferma la ricezione, in cui mancanza viene rimosso dalla sequenza
  - **Nel caso peggiore un processo aspetta una intera rotazione del gettone**
- **Il gettone è 1 SPF → se perso, va rigenerato**

Università degli Studi di Padova

Sistemi distribuiti: sincronizzazione

Mutua esclusione - 5

- **I 3 algoritmi possono essere raffrontati in relazione a 3 criteri fondamentali**
  - **Numero di messaggi necessari al processo per poter operare sulla risorsa richiesta (ingresso ed uscita)**
  - **Il tempo necessario per perché la richiesta abbia successo**
  - **Le debolezze dell'algoritmo**
- **Questi 3 criteri possono essere applicati a varie classi di algoritmi distribuiti**

Mutua esclusione - 6

Algoritmo	# Messaggi per accesso e rilascio risorsa	Massima attesa di accesso (in trasmissione di messaggi → comunicazione >> lavoro)	Debolezze (SPF)
Centralizzato	**3** (ENTER, GRANTED, RELEASED)	**2** (ENTER, GRANTED)	**Guasto del coordinatore**
Distribuito	**2 (n – 1)** (GRANT?, RELEASED a tutti gli altri)	**2 (n – 1)**	**Guasto di qualsiasi processo**
Gettone circolante	**1 .. ∞** (se tutti o nessun processo sono interessati alla risorsa)	**0 .. n – 1** (gettone in possesso, gettone all'altro capo)	**Gettone perso, guasto di processo**

Raffronto degli algoritmi

Argomenti non trattati

- **Argomenti importanti per la problematica di questa lezione non trattati per mancanza di tempo**
  - **Sincronizzazione degli orologi fisici**
    - Il *middleware* di ogni nodo del sistema distribuito aggiusta il valore del suo orologio fisico in modo coerente con quello degli altri
  - **Sincronizzazione degli orologi logici**
    - Leslie Lamport ha mostrato che non è necessario l'accordo degli orologi fisici, ma solo l'ordinamento degli eventi (relazione "precede")
  - **Transazioni distribuite**
    - Come ottenere mutua esclusione ed operazioni atomiche su dati condivisi