

Università degli Studi di Padova

Criteri di sincronizzazione



Anno accademico 2004/5
Corso di Sistemi Concorrenti e Distribuiti

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

Corso di Laurea Specialistica in Informatica, Università di Padova 1/17

Università degli Studi di Padova

Criteri di sincronizzazione

Criteri di valutazione - 1

- ❑ **I costrutti di un linguaggio di programmazione sono spesso valutati da 2 punti di vista**
 - **Potere espressivo (criterio oggettivo)**
 - Misura la capacità del linguaggio di soddisfare direttamente i bisogni caratterizzanti del dominio applicativo
 - **Usabilità (criterio soggettivo)**
 - Misura il grado di interazione (efficacia) e di integrazione (coerenza) dei costrutti in esame, tra di loro e verso il resto del linguaggio
- ❑ **Tale schema di valutazione può essere utile applicato ai costrutti di sincronizzazione**
 - 1979, Toby Bloom, "Evaluating synchronisation mechanisms", Proc. 7th ACM Symposium on Operating System Principles, pp. 24-32

Corso di Laurea Specialistica in Informatica, Università di Padova 2/17

Università degli Studi di Padova

Criteri di sincronizzazione

Criteri di valutazione - 2

- ❑ **Toby Bloom ha identificato i problemi chiave che devono essere affrontati nel progettare forme di sincronizzazione**
 - Tipo di servizio richiesto → canale tipato
 - Ordine di arrivo delle richieste → politiche di accodamento
 - Stato del servente (anche come conseguenza della sua storia d'uso) → guardie [*avoidance synchronisation*]
 - Priorità del cliente
 - Parametri della richiesta

Quale soluzione?

Corso di Laurea Specialistica in Informatica, Università di Padova 3/17

Università degli Studi di Padova

Criteri di sincronizzazione

L'allocazione delle risorse - 1

- ❑ **Problema assolutamente centrale ad ogni forma di programmazione concorrente**
 - Coinvolge tutte le dimensioni identificate da Toby Bloom
 - È particolarmente difficile da risolvere con il solo uso di guardie

Esempio: Si costruisca un controllore di risorse che debba allocare risorse ad un gruppo di clienti, ove le risorse siano in numero finito ed i clienti sono in competizione tra loro per averle. Ciascun cliente C_i può richiedere $N_i \geq 1$ risorse alla volta. Il protocollo di assegnazione sancisce che la richiesta, se accettata, debba essere soddisfatta integralmente, altrimenti sia tenuta in sospenso ed il cliente bloccato sino a quando ne diventi possibile il soddisfacimento.

Corso di Laurea Specialistica in Informatica, Università di Padova 4/17

Università degli Studi di Padova

Criteri di sincronizzazione

L'allocazione delle risorse - 2

- ❑ **È immediato capire come la dimensione della richiesta di un cliente debba essere un parametro del messaggio**
- ❑ **Per leggere il parametro (e determinare la soddisfaccibilità della richiesta) occorre accettare la comunicazione**
 - Che fare, però, se la richiesta non fosse soddisfacibile in quel momento?
 - Di sicuro non vogliamo ricorrere all'uso del "busy wait"!
 - Questione di usabilità e non di potere espressivo

Corso di Laurea Specialistica in Informatica, Università di Padova 5/17

Università degli Studi di Padova

Criteri di sincronizzazione

L'allocazione delle risorse - 3

- ❑ **L'uso delle guardie di Dijkstra comporta *avoidance synchronization***
 - Meccanismo che consente di evitare sincronizzazione quando questa non sia considerata utile
- ❑ **L'uso delle variabili di condizione del *monitor* di Hoare consente di ottenere sincronizzazione condizionale (*condition synchronization*)**
 - Ciò che ci servirebbe in questo caso: prima valutare il parametro e poi, se necessario, imporre attesa di sincronizzazione
 - Abbiamo però visto che il limite strutturale del *monitor* è di richiedere programmazione esplicita della sincronizzazione!

Corso di Laurea Specialistica in Informatica, Università di Padova 6/17

Criteri di sincronizzazione

Università degli Studi di Padova Criteri di sincronizzazione

L'allocazione delle risorse - 4

❑ Per affrontare il problema sono possibili almeno 2 estensioni del protocollo di *avoidance synchronization*

- Consentire all'espressione di guardia di avere accesso ai parametri in ingresso della richiesta
 - Linguaggio SR (*Synchronizing Resource*), <http://www.cs.arizona.edu/sr/>
- Consentire al processo servente di trasferire il cliente accettato ma non soddisfacibile su un'altra coda d'attesa
 - Dove il cliente attenderà il verificarsi di condizioni più propizie
 - Consentendo così al canale di poter accogliere nuove richieste

Corso di Laurea Specialistica in Informatica, Università di Padova 7/17

Università degli Studi di Padova Criteri di sincronizzazione

L'allocazione delle risorse - 5

Forma base (1 risorsa per richiesta) Soluzione SR per il nostro problema

```

protected Controller is
  entry Allocate (R : out Resource);
  procedure Release (R : Resource);
private
  Free : Natural := Full_Capacity;
  ...
end Controller;
protected body Controller is
  entry Allocate (R : out Resource)
  when Free > 0 is
  begin
    Free := Free - 1;
  ...
  end Allocate;
  procedure Release (R : Resource) is
  begin
    Free := Free + 1;
  end Release;
end Controller;
    
```

```

type Request is range 1..Max_Requests;
protected Controller is
  entry Allocate
  (R : out Resource;
   Amount : in Request);
  procedure Release
  (R : Resource;
   Amount : Request);
private
  Free : Request := Request_Last; ...
end Controller;
protected body Controller is
  entry Allocate
  (R : out Resource;
   Amount : in Request)
  when Amount <= Free is
  begin
    Free := Free - Amount;
  end Allocate;
  procedure Release (...) is ...
  end Controller;
    
```

La guardia usa un parametro in di ingresso

Corso di Laurea Specialistica in Informatica, Università di Padova 8/17

Università degli Studi di Padova Criteri di sincronizzazione

L'allocazione delle risorse - 6

❑ Il difetto principale dell'approccio SR è il suo eccessivo costo realizzativo

- Legare direttamente la sincronizzazione (e dunque l'accodamento) al valore di un parametro della richiesta comporta che tutte le richieste accodate debbano essere rivalutate ogni volta che la condizione possa essere cambiata!

❑ L'alternativa è trasferire la richiesta attualmente non soddisfacibile ad un'altra coda

- Senza doverne rivalutare l'eventuale guardia
- Con una singola operazione atomica (senza *race condition*)

Corso di Laurea Specialistica in Informatica, Università di Padova 9/17

Università degli Studi di Padova Criteri di sincronizzazione

L'allocazione delle risorse - 7

❑ Vediamo come il trasferimento di coda risolve il nostro problema

- **Esercizio 10**
Migliorare la soluzione proposta evitando di percorrere inutilmente la coda sul canale interno Assign (ossia quando le risorse disponibili non fossero ancora disponibili per soddisfare alcuna richiesta pendente)

Corso di Laurea Specialistica in Informatica, Università di Padova 10/17

Università degli Studi di Padova Criteri di sincronizzazione

Semantica del trasferimento di coda - 1

❑ Il trasferimento di coda (*requeue*) non è una normale chiamata di procedura

The diagram illustrates two scenarios. On the left, 'Caso normale' shows a process P calling a procedure in process Q, which then returns. On the right, 'Trasferimento di coda' shows a channel C calling a procedure in process E1. Instead of returning to C, E1 performs a 'requeue' operation to process E2, which then returns to E1. A 'NO' label is placed between E1 and E2, indicating that the call is not a standard procedure call.

Corso di Laurea Specialistica in Informatica, Università di Padova 11/17

Università degli Studi di Padova Criteri di sincronizzazione

Semantica del trasferimento di coda - 2

❑ Il trasferimento pone 2 problemi importanti

- Verso quale coda di canale (*entry*) possa avvenire
 - La coda di destinazione (*target*) può essere una qualsiasi coda di canale, sia di processo che di risorsa protetta, anche se, metodologicamente, è preferibile restringerne l'ambito alla stessa entità di partenza
 - Il trasferimento verso coda in un'altra entità comporta però perdita di possesso dell'entità di partenza, il che potrebbe non essere desiderabile
 - Il canale destinazione deve essere compatibile con quello di partenza → o nessun parametro o esattamente gli stessi della chiamata esterna
 - La direzione del trasferimento determina il destinatario finale del "lucchetto"
- Come trattare l'eventuale *time-out* posto dal cliente sulla richiesta
 - Il trasferimento deve indicare esplicitamente se applicare il *time-out* al canale destinazione oppure considerarlo soddisfatto con l'attuale accettazione

Corso di Laurea Specialistica in Informatica, Università di Padova 12/17

Criteri di sincronizzazione

Università degli Studi di Padova Criteri di sincronizzazione

Semantica del trasferimento di coda - 3

Che succede in questo caso?

Vi sono 2 possibilità:

- 1) La chiamata B.E1 **non** viene accolta entro il tempo T1 → chiamata annullata
- 2) La chiamata viene accolta entro E1, ove esegue per un tempo T2, ma verrà annullata se E2 non venisse accolta entro T2+T1

```

-- A
select
  B.E1;
or
  delay T1;
end select;

-- B
select
  accept E1 do
    ... -- T2 seconds
    requeue E2 [with abort];
  end E1;
or
  ...
end select;
    
```

Questa clausola preserva l'eventuale *time-out* posto dal chiamante sulla coda di destinazione

Corso di Laurea Specialistica in Informatica, Università di Padova 13/17

Università degli Studi di Padova Criteri di sincronizzazione

Esercizio 11

Sull'utilità del trasferimento verso altra coda

Esercizio 11: Un *router* di rete può instradare pacchetti in ingresso verso 3 linee di comunicazione L_{a-c} , distinte ma funzionalmente equivalenti. La linea L_a rappresenta la scelta preferenziale, ma le altre linee (prima L_b e poi L_c) sono usate quando la precedente alternativa risulti sovraccarica. Si realizzi un programma concorrente che realizzi questa politica, considerando il diagramma qui riportato come possibile fonte di ispirazione.

Corso di Laurea Specialistica in Informatica, Università di Padova 14/17

Università degli Studi di Padova Criteri di sincronizzazione

Un esempio complesso - 1

- La possibilità di trasferimento tra code consente un grande potere espressivo, che aiuta a rappresentare situazioni complesse
- Vogliamo simulare il comportamento di un sistema di trasporto viaggiatori su linea circolare (p.es. una linea ferroviaria di tipo TAF)
 - N stazioni su linea circolare → una risorsa protetta per stazione, presso la quale ciascun viaggiatore in partenza si blocca in attesa del treno
 - 1 treno a capienza finita → entità attiva
 - K viaggiatori che si recano alla loro stazione di partenza avendo una stazione di arrivo → un'entità attiva per ogni viaggiatore

Corso di Laurea Specialistica in Informatica, Università di Padova 15/17

Università degli Studi di Padova Criteri di sincronizzazione

Un esempio complesso - 2

- Il trasferimento di coda è un modo pratico di simulare il trasporto dei viaggiatori
 - Quando il treno arriva ad una stazione, il programma trasferisce i viaggiatori in attesa in quella stazione (ma solo fino alla capienza massima del treno) nella coda della loro stazione di destinazione, dalla quale saranno rilasciati (= arrivati) quando il treno vi farà sosta

Corso di Laurea Specialistica in Informatica, Università di Padova 16/17

Università degli Studi di Padova Criteri di sincronizzazione

Un esempio complesso - 3

- Vediamo come programmare questa soluzione ...
- Esercizio 12 Realizzare la stessa semantica della soluzione mostrata usando un altro linguaggio a piacere

Corso di Laurea Specialistica in Informatica, Università di Padova 17/17