Lezione 20 MSC
Introduzione al model checking

Roberto Gorrieri

Oltre all'equivalence-checking (1)

- Data una implementazione *Impl* (di solito scritta in CCS), per verificare se questa è corretta bisogna:
  - Definirsi una specifica *Spec*, di solito scritta anch'essa in CCS, che prescriva il comportamento astratto che il sistema deve soddisfare.
  - Identificare una opportuna equivalenza ≈
  - Verificare che valga *Spec* ≈ *Impl*
- Ma non è sufficientemente flessibile per descrivere proprietà "puntuali/parziali" di un sistema. Ad esempio: "il sistema può fare subito l'azione a ma non l'azione b", oppure "comunque faccia a ora, poi può subito fare b", oppure anche "il sistema non può mai andare in deadlock".

Il solito semplice protocollo

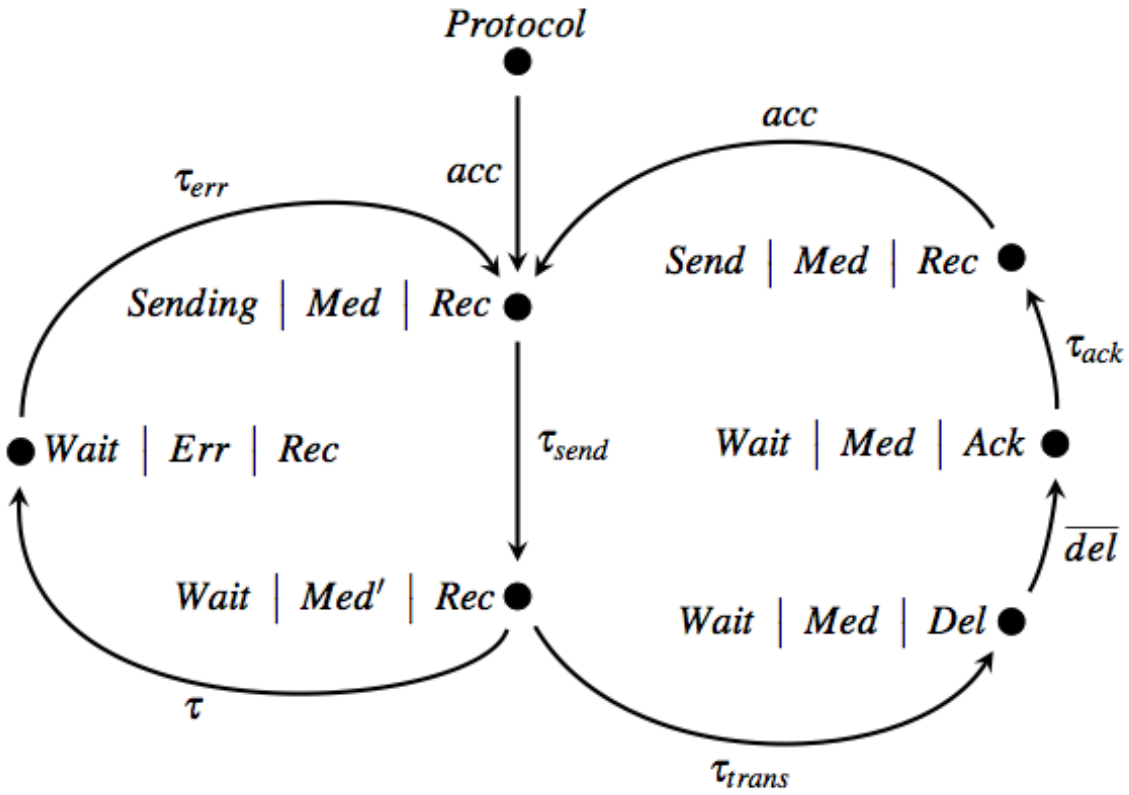

Fig. 3.8 The behaviour of the simple communication protocol.

Specifica ed equivalence checking

- Specifica sequenziale:

ProtSpec = acc.’del.ProtSpec

- Implementazione distribuita:

Protocol quello del lucido precedente

- Equivalence checking:

ProtSpec ≈ Protocol vale

Oltre all'equivalence-checking (2)

- Data *Impl*, descritta ad esempio in CCS, per verificare se questa è corretta, uno potrebbe
  - definire un set di proprietà, descritte attraverso una qualche logica, che rappresenterebbe la specifica del sistema
  - definire la relazione di soddisfacimento $\models$ *Impl* $\models$ F dove F è una proprietà nel set della specifica.
  - verificare che la formula F sia davvero soddisfatta da *Impl*.
- Questo è il cosiddetto Model Checking

Hennessy-Milner Logic (HML)

- Logica modale: esprime “cosa può accadere ora”, ovvero possibilità <-> e necessità [-]
- Ad esempio posso esprimere queste proprietà:
- Protocol $\models$ <acc >tt posso ricevere un msg all’inizio
- Protocol $\models$ [[‘del]]ff non posso inoltrare inizialmente, nemmeno facendo un po’ di tau prima e/o dopo
- Protocol $\models$ [acc]<<‘del>>tt se ricevo un msg, potrò inoltrarlo eventualmente facendo un po’ di tau prima e/o dopo
- Protocol $\models$ [acc][[acc]]ff non potrò ricevere due msg di seguito, nemmeno se ammetto un po’ di tau fra le due azioni

Hennessy-Milner Theorem

Dato un lts image-finite, due stati P e Q sono bisimili se e soltanto se soddisfano le stesse formule della logica HML.

Relazione precisa fra equivalence-checking e model-checking.

HML con ricorsione

- HML esprime proprietà semplici, di possibilità e necessità (logica modale)
- Può essere utile avere logiche che esprimono proprietà temporali (logica temporale), del tipo:
  - "la vending machine non ruba mai i soldi" (safety property: something bad can never happen)
  - "la vending machine, dopo l'inserzione di una moneta, darà prima o poi una bevanda" (liveness property: something good will happen eventually)
- Necessità di parlare di minimi e massimi punti fissi

HML con ricorsione (2)

- Logica temporale: esprime “cosa può accadere in (un) futuro (arbitrariamente lontano)”
- Ad esempio posso esprimere queste proprietà:
- Protocol $\models$ X

$\langle Act\rangle tt = \langle a_1\rangle tt \vee \ldots \vee \langle a_n\rangle tt$

$[Act]F = [a_1]F \wedge \ldots \wedge [a_n]F$

$X =_{max} \langle Act\rangle tt \wedge [Act]X$

no reachable state is a deadlock

- Protocol $\models$ Y

$Y =_{max} F \wedge [acc](\langle\langle \text{‘del}\rangle\rangle Y \wedge [[acc]]\, ff)$

$F = \langle acc\rangle tt \wedge [[\text{‘del}]]ff$

che garantisce che un’azione di acc può sempre essere seguita da una ‘del (cominciando con acc), in alternanza