On the net encoding of asynchronous interactions

Fabio Gadducci

University of Pisa

Joint work with
Paolo Baldan (Univ. Padova), Filippo Bonchi (CWI Amsterdam)

Overview

General Theme

Relating calculi with asynchronous communication and Petri nets

Asynchronous calculi

Asynchronous process calculi

Formal models of distributed and concurrent systems with asynchronous communication [Honda,Tokoro'91], [Boudol'92]:

- no handshake between sender and receiver
- non-blocking send
- the message is sent, it travels to destination and it is (possibly) received

Observations

Only message sending is observable, reception is not

Asynchronous calculi

Asynchronous process calculi

Formal models of distributed and concurrent systems with asynchronous communication [Honda,Tokoro'91], [Boudol'92]:

- no handshake between sender and receiver
- non-blocking send
- the message is sent, it travels to destination and it is (possibly) received

Observations

Only message sending is observable, reception is not

Asynchronous CCS

CCS fragment of asynchronous pi-calculus

Petri nets

Petri Nets

Widely used model of concurrent and distributed systems:

- formal semantics
- intuitive graphical representation

Asynchrony in Petri nets

Tokens are first generated by some transition and then consumed by others

Relating asynchronous calculi and Petri nets

Can this intuitive correspondence between asynchronous calculi and Petri nets made formal?

Open Petri nets

Open Petri nets

Generalising Petri nets with composition and reactivity for modelling “open” systems

- interface / interaction with the environment through some designated places
- composition between nets (using an interface)

Open Petri nets

Open Petri nets

Generalising Petri nets with composition and reactivity for modelling "open" systems

- interface / interaction with the environment through some designated places
- composition between nets (using an interface)

Related ...

- Compositional semantics for Petri nets (SCONE, Petri box calculus, Petri Net algebra)
- Petri nets as reactive systems in the sense of Leifer, Milner ([Milner], [Sassone,Sobocinski])
- Workflows and web-service models (e.g., [van der Aalst])

Results: Encoding asynchronous CCS into open nets

Encoding bounded asyncronous CCS into open nets

- it preserves structural congruence
- message exchanges as interactions at open places
- operational semantics: CCS reductions $\leftrightarrow$ PN firings
- it preserves and reflects weak and strong bisimilarity

Results: Technology transfer on Expressiveness

Intimate connection between the two formalisms, useful for some technology transfer on expressiveness

Results: Technology transfer on Expressiveness

Intimate connection between the two formalisms, useful for some technology transfer on expressiveness

Undecidability of bisimilarity

(Strong/weak) bisimilarity for bounded asynchr. CCS is undecidable

$$\downarrow$$

(Strong/weak) bisimilarity for open nets is undecidable

Results: Technology transfer on Expressiveness

Intimate connection between the two formalisms, useful for some technology transfer on expressiveness

Undecidability of bisimilarity

(Strong/weak) bisimilarity for bounded asynchr. CCS is undecidable

↓

(Strong/weak) bisimilarity for open nets is undecidable

Decidability of convergence

Reachability is decidable for open Petri nets

↓

Reachability/convergence is decidable for bounded asynchr. CCS

Asynchronous CCS

[Amadio,Castellani,Sangiorgi]

Syntax

$P ::= M, \ \ \bar{a}, \ \ (\nu a)P, \ \ P_1 \mid P_2, \ \ !_a.P$ (Processes)

$M ::= 0, \ \ \mu.P, \ \ M_1 + M_2$ (Sums)

Asynchronous CCS

[Amadio,Castellani,Sangiorgi]

Syntax

$P ::= M,\ \ \bar{a},\ \ (\nu a)P,\ \ P_1 \mid P_2,\ \ !_a.P$ (Processes)

$M ::= 0,\ \ \mu.P,\ \ M_1 + M_2$ (Sums)

Reduction semantics

$a.P + M \mid \bar{a} \to P \qquad \tau.P + M \to P \qquad !_a.P \mid \bar{a} \to P \mid !_a.P$

(+ usual structural axioms)

Asynchronous CCS: behavioral equivalences

Barb

Equivalence based on the notion of barb

$$P \downarrow \bar{a} \qquad \textit{if } P \equiv \bar{a} \mid Q$$

Asynchronous CCS: behavioral equivalences

Barb

Equivalence based on the notion of barb

$$P \downarrow \bar{a} \qquad \textit{if } P \equiv \bar{a} \mid Q$$

Barbed equivalence

A barbed bisimulation is a symmetric relation $R \subseteq Proc \times Proc$ s.t. whenever $(P, Q) \in R$ then

1. if $P \downarrow \bar{a}$ then $Q \downarrow \bar{a}$,
2. if $P \rightarrow P'$ then $Q \rightarrow Q'$ and $(P', Q') \in R$.

Barbed bisimilarity $\sim$ is the largest barbed bisimulation

Asynchronous CCS: equivalences

Barbed congruence

$P \sim_b Q$ if $P \mid S \sim Q \mid S$ for all processes $S \in Proc$

Asynchronous CCS: equivalences

Barbed congruence

$P \sim_b Q$ if $P \mid S \sim Q \mid S$ for all processes $S \in Proc$

1-bisimilarity

A 1-bisimulation is a symmetric relation $R \subseteq Proc \times Proc$ s.t. whenever $(P, Q) \in R$ then

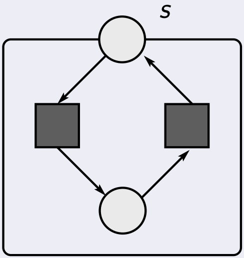
1. if $P \rightarrow P'$ then $Q \rightarrow Q'$ and $(P', Q') \in R$,
2. $\forall a \in \mathcal{N}. (P \mid \bar{a}, Q \mid \bar{a}) \in R$,
3. if $P \equiv P' \mid \bar{a}$ then $Q \equiv Q' \mid \bar{a}$ and $(P', Q') \in R$.

Strong 1-bisimilarity $\sim_1$ is the largest strong 1-bisimulation

Open nets

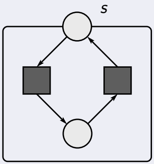
Interface of the net

- open places
- the enviroment can put/remove tokens



Open nets: Behaviour

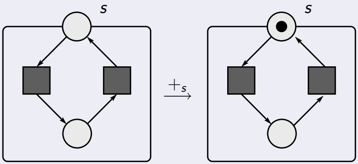
Interactions at the interfaces / internal firing



Weak and strong bisimilarities are totally standard

Open nets: Behaviour

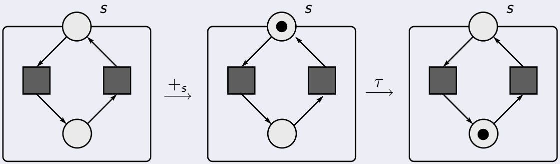
Interactions at the interfaces / internal firing



Weak and strong bisimilarities are totally standard

Open nets: Behaviour

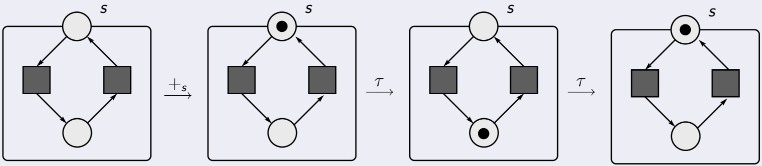
Interactions at the interfaces / internal firing



Weak and strong bisimilarities are totally standard

Open nets: Behaviour

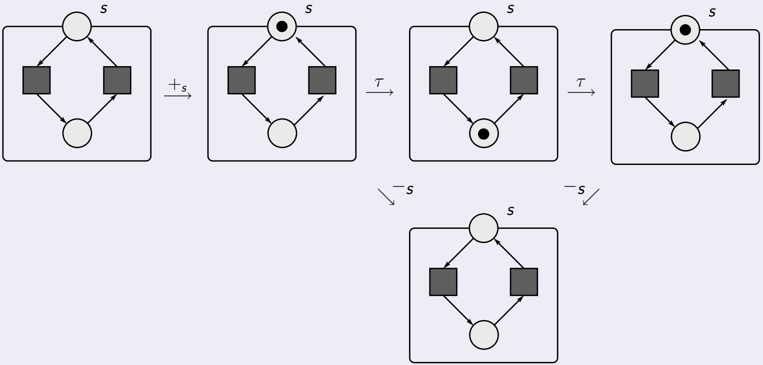
Interactions at the interfaces / internal firing



Weak and strong bisimilarities are totally standard

Open nets: Behaviour

Interactions at the interfaces / internal firing



Weak and strong bisimilarities are totally standard

Encoding asynchronous CCS into open nets

Bounded asynchronous CCS processes

The encoding is restricted to bounded processes: restriction never occurs under the scope of replication

$$!_a.(\ldots(\nu b)(\ldots)\ldots) \quad \text{NO!!}$$

Encoding asynchronous CCS into open nets

Bounded asynchronous CCS processes

The encoding is restricted to bounded processes: restriction never occurs under the scope of replication

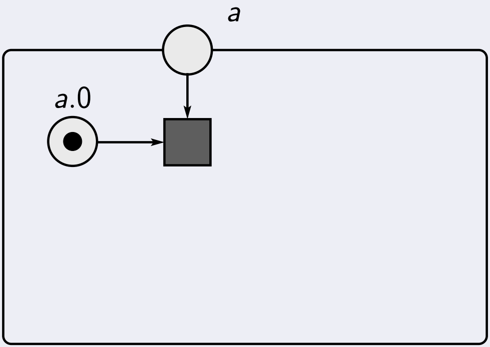
$$!_a.(\ldots(\nu b)(\ldots)\ldots) \qquad \text{NO!!}$$

Idea

- open places represent the free channels of a process
- messages represented by tokens in places
- transitions encode the control flow

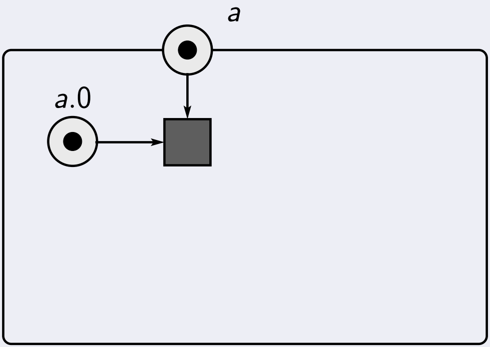
Encoding: Prefix, parallel, restriction

$a.0$



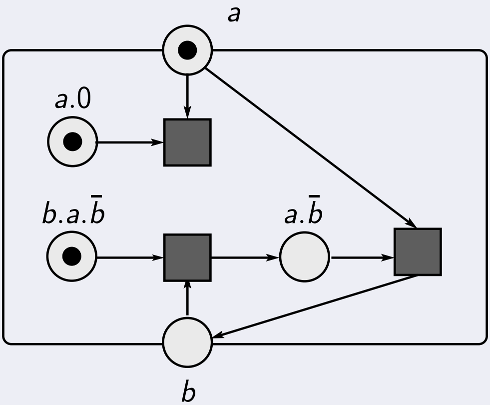
Encoding: Prefix, parallel, restriction

$\bar{a} \mid a.0$



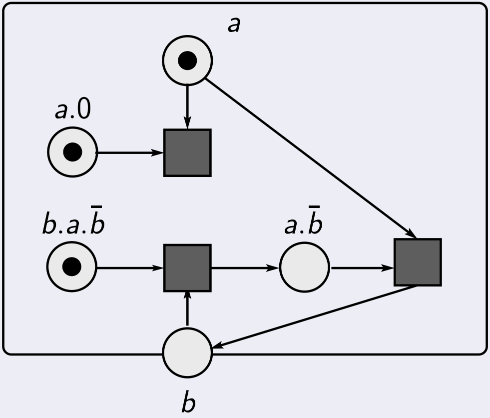
Encoding: Prefix, parallel, restriction

$\bar{a} \mid a.0 \mid b.a.\bar{b}$



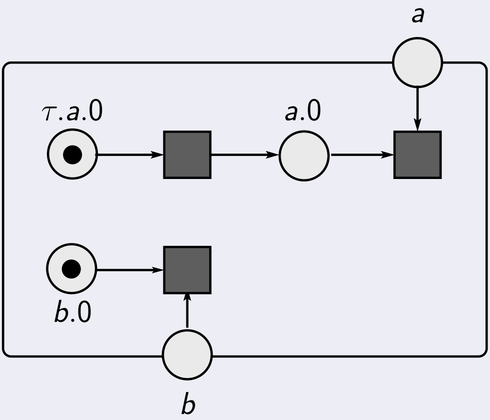
Encoding: Prefix, parallel, restriction

$(\nu a)(\bar{a} \mid a.0 \mid b.a.\bar{b})$



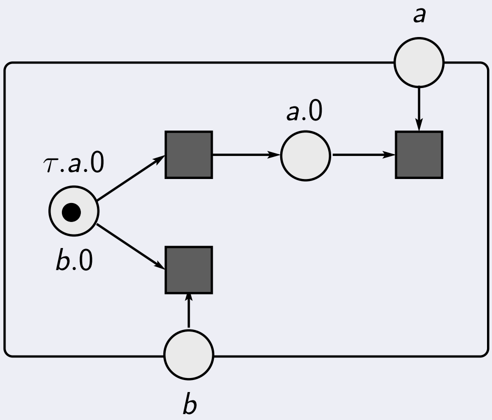
Encoding: Sum

$\tau.a.0 + b.0$



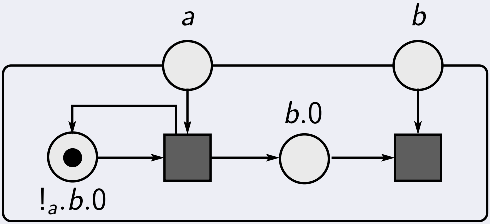
Encoding: Sum

$\tau.a.0 + b.0$



Encoding: Replication

$!_a.b.0$

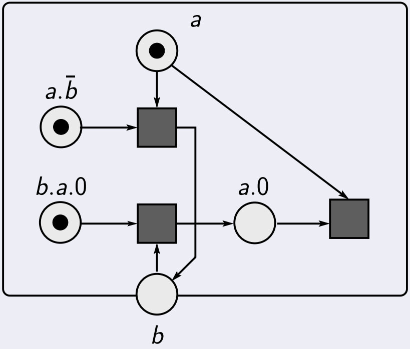


In general . . .

Any bounded asynchr. CCS process P encoded as an open net $[\![P]\!]$

Any Q such that $P \longrightarrow^* Q$ corresponds to a marking $\mathbf{m}(Q)$ of $[\![P]\!]$

$$(\nu a)(\bar{a} \mid a.\bar{b} \mid b.a.0)$$
$$\downarrow$$
$$(\nu a)(\bar{b} \mid b.a.0)$$
$$\downarrow$$
$$(\nu a)(a.0)$$



In general . . .

Any bounded asynchr. CCS process P encoded as an open net $[\![P]\!]$

Any Q such that $P \rightarrow^* Q$ corresponds to a marking $\mathbf{m}(Q)$ of $[\![P]\!]$

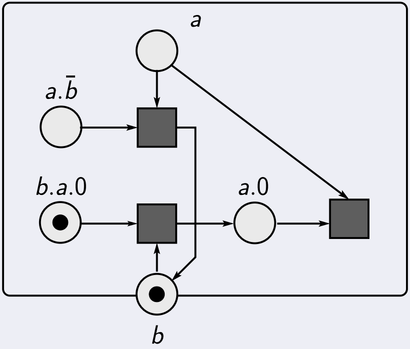
$(\nu a)(\bar{a} \mid a.\bar{b} \mid b.a.0)$

$\downarrow$

$(\nu a)(\bar{b} \mid b.a.0)$

$\downarrow$

$(\nu a)(a.0)$

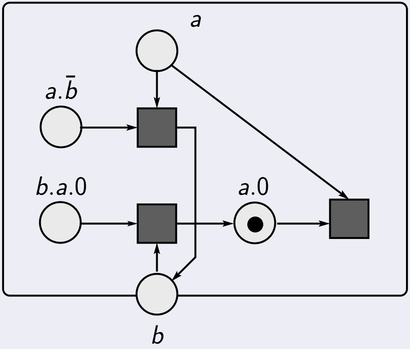


In general . . .

Any bounded asynchr. CCS process P encoded as an open net $[\![P]\!]$

Any Q such that $P \rightarrow^* Q$ corresponds to a marking $\mathbf{m}(Q)$ of $[\![P]\!]$

$$(\nu a)(\bar{a} \mid a.\bar{b} \mid b.a.0)$$
$$\downarrow$$
$$(\nu a)(\bar{b} \mid b.a.0)$$
$$\downarrow$$
$$(\nu a)(a.0)$$



Properties of the encoding

Preservation and reflection of the operational semantics

For any bounded process P

$$P \rightarrow Q \qquad \text{iff} \qquad \mathbf{m}(P) \rightarrow \mathbf{m}(Q) \text{ in the open net } [\![P]\!]$$

Properties of the encoding

Preservation and reflection of the operational semantics

For any bounded process P

$$P \rightarrow Q \qquad \text{iff} \qquad \mathbf{m}(P) \rightarrow \mathbf{m}(Q) \text{ in the open net } [\![P]\!]$$

Preservation and reflection of (strong/weak) bisimilarity

For any two bounded processes

$$P \sim Q \qquad \text{iff} \qquad [\![P]\!] \sim [\![Q]\!]$$

Undecidability of bisimilarity

Undecidability of bisimilarity for bounded asynchronous CCS

- 2-register machines:
  - two integer registers r, s
  - program instructions: increment a register, jump on zero
- encoding 2-register machines as bounded aCCS processes
  - registers are represented as channels and their content as messages on such channels
  - zero testing can be only "weakly" simulated
- for any given machine we can construct two processes P and P' such that $P \sim P'$ iff machine halts

$\longrightarrow$ bisimilarity on bounded asynchronous CCS is undecidable

Undecidability of bisimilarity

As a consequence of the properties of the encoding . . .

Corollary

Bisimilarity is undecidable for open Petri nets

Note

Outside the known undecidability results for PNs as we only observe interactions with the environment (all "traditional nets" are weakly bisimilar)

Convergence/reachability is decidable

Convergence in process calculi

A process P is called *convergent* if there is Q such that $P \Rightarrow Q \not\rightarrow$

Reachability and presence of deadlocks is decidable for (open) nets

$\downarrow$

Corollary

Convergence is decidable for bounded asyncronous CCS

Convergence/reachability is decidable

Convergence in process calculi

A process P is called *convergent* if there is Q such that $P \Rightarrow Q \not\rightarrow$

Reachability and presence of deadlocks is decidable for (open) nets
$\downarrow$

Corollary

Convergence is decidable for bounded asyncronous CCS

More generally ...

For P, Q bounded processes, the problem

$$P|R \Rightarrow Q \text{ for some } R = \bar{a}_1 \mid \ldots \mid \bar{a}_n \text{ is decidable}$$

Conclusions

Tight relation between asynchronous CCS and open Petri nets, exploited for a technology transfer in expressiveness

Generalisation to full CCS and pi-calculus

Infiniteness of channels and variable topology. Open dynamic nets? Open GTSs?

Concurrent semantics

- well-understood for open Petri nets
- few studies for asynchronous calculi

Step equivalences

Weak concurrent equivalences coincide with non-concurrent ones: intriguing connection between concurrency and asynchrony