On Composing Communicating Systems

Franco Barbanera[1], Ivan Lanese[2], Emilio Tuosto[3]

[1] University of Catania
[2] University of Bologna/INRIA
[3] GSSI (Gran Sasso Science Institute)

ICE - June 17, 2022

The need of systems composability

- Concurrent/Distributed systems are **not**

  STAND-ALONE ENTITIES

- (expecially nowadays) they are parts of

  JIGSAWS NEVER COMPLETELY TERMINATED

The need of systems composability

- Concurrent/Distributed systems are **not**

  STAND-ALONE ENTITIES

- (expecially nowadays) they are parts of

  JIGSAWS NEVER COMPLETELY TERMINATED

The need of systems composability

- Concurrent/Distributed systems are **not**

STAND-ALONE ENTITIES

- (expecially nowadays) they are parts of

JIGSAWS NEVER COMPLETELY TERMINATED

The need of systems composability

- Concurrent/Distributed systems are **not**

  STAND-ALONE ENTITIES

- (expecially nowadays) they are parts of

  JIGSAWS NEVER COMPLETELY TERMINATED

The need of systems composability

- Concurrent/Distributed systems are **not**

STAND-ALONE ENTITIES

- (expecially nowadays) they are parts of

JIGSAWS NEVER COMPLETELY TERMINATED

The need of systems composability

Composability is useful both

- at design phase (modular design);
- at deployment phase and beyond
  - modular deployment;
  - new functionalities needed;
  - system scalability
  - etc.

The need of systems composability

Composability is useful both

- at design phase (modular design);
- at deployment phase and beyond
  - modular deployment;
  - new functionalities needed;
  - system scalability
  - etc.

The need of systems composability

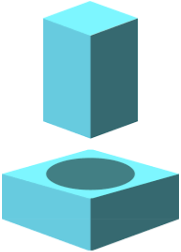
Composability is useful both

- at design phase (modular design);
- at deployment phase and beyond OUR SETTING
  - modular deployment;
  - new functionalities needed;
  - system scalability
  - etc.

Good composition methods

They should be

- FLEXIBLE

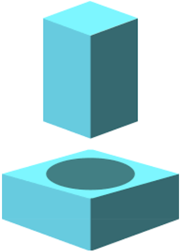


- SAFE

Good composition methods

They should be

- FLEXIBLE

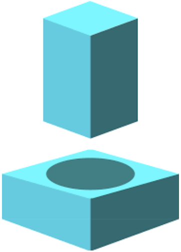


- SAFE

Good composition methods

They should be

- FLEXIBLE



- SAFE

Good composition methods are **safe**

If one starts from something like this....

Good composition methods are **safe**

If one starts from something like this....

Good composition methods are **safe**

If one starts from something like this....

Good composition methods are **safe**

...should not end up with something like that

Good composition methods are **safe**

...should not end up with something like that

Good composition methods are **safe**

Safe composition methods guarantee not to “break” any relevant property of the single systems we compose.

Good composition methods are **flexible**

A **flexible** composition method

- alters as less as possible the single systems
- is “system independent”, that is
  - the composition mechanism is not part of the system
  - it allows to consider any system as potentially open

Good composition methods are **flexible**

A **flexible** composition method

- alters as less as possible the single systems
- is “system independent”, that is
  - the composition mechanism is not part of the system
  - it allows to consider any system as potentially open

Good composition methods are **flexible**

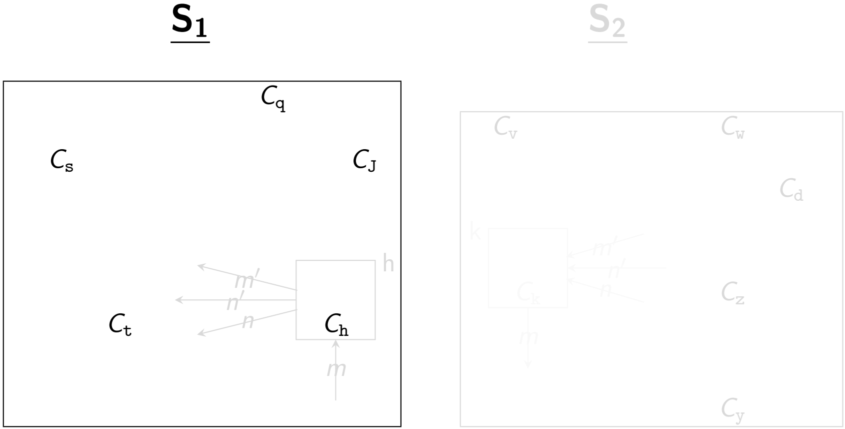
A **flexible** composition method

- alters as less as possible the single systems
- is “system independent”, that is
  - the composition mechanism is not part of the system
  - it allows to consider any system as potentially open

The “participants-as-interfaces” (PaI) approach

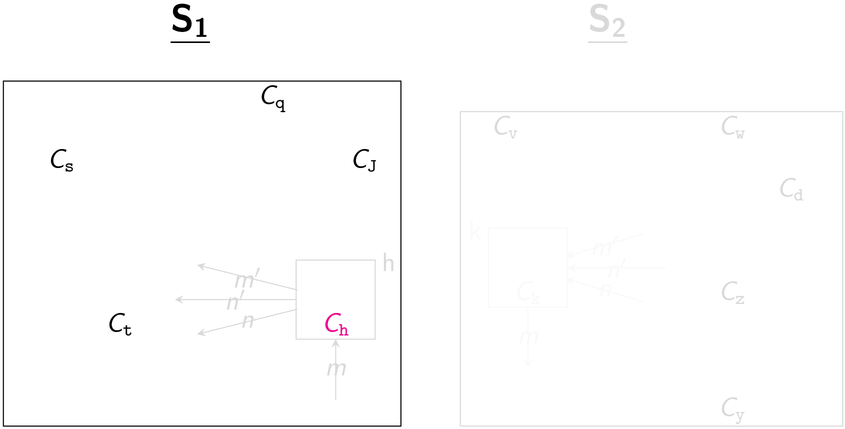
For systems with message-passing interactions

The “participants-as-interfaces” (PaI) approach



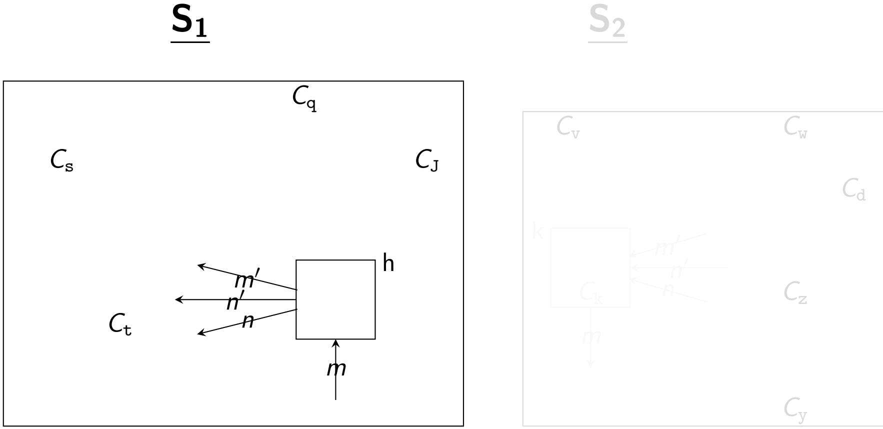
C_h’s behaviour can be looked at as what can be offered by an outer system

The “participants-as-interfaces” (PaI) approach



$\mathrm{C_h}$’s behaviour can be looked at as what can be offered by an outer system

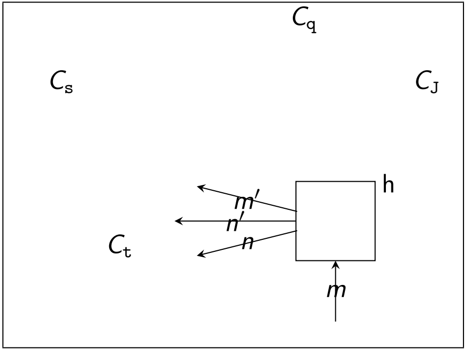
The “participants-as-interfaces” (PaI) approach



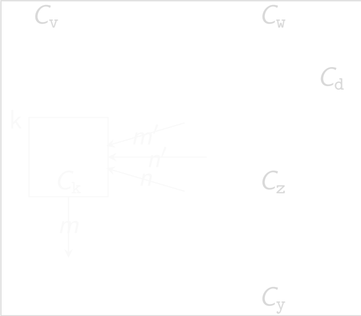
$\mathsf{C_h}$’s behaviour can be looked at as what can be offered by an outer system

The “participants-as-interfaces” (PaI) approach

S₁



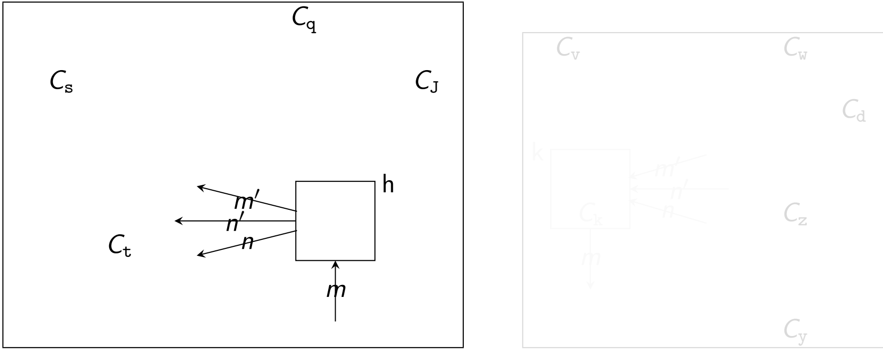
S₂



We abstract here from the way communications are performed and from the logical order of the exchanged messages. C_h's behaviour can be looked at as what can be offered by an outer system.

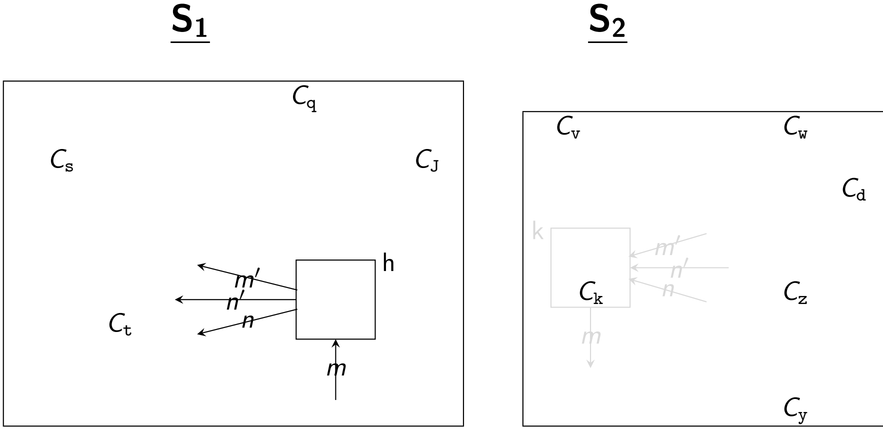
The “participants-as-interfaces” (PaI) approach

S$_1$ S$_2$



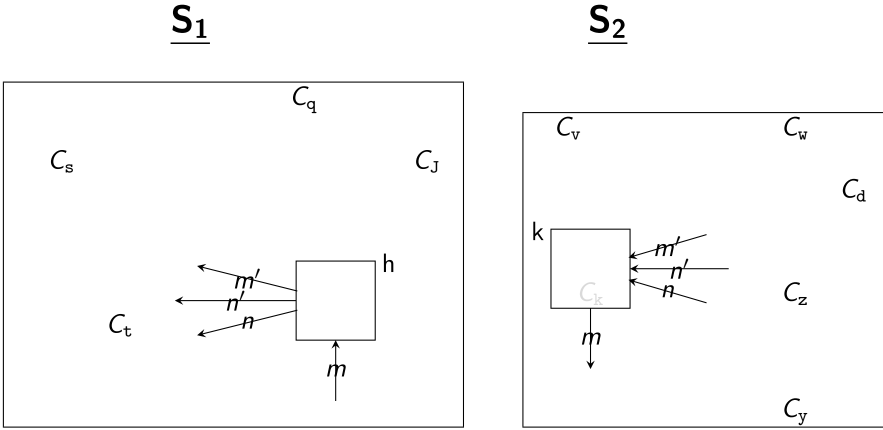
$\mathsf{C_h}$’s behaviour can be looked at as what can be offered by an outer system

The “participants-as-interfaces” (PaI) approach



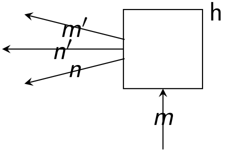
$\mathrm{C_h}$'s behaviour can be looked at as what can be offered by an outer system

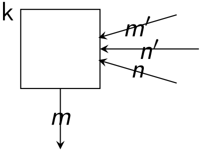
The “participants-as-interfaces” (PaI) approach



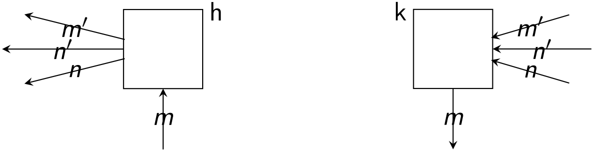
C_{h}’s behaviour can be looked at as what can be offered by an outer system

The "components-as-interfaces" (PaI) approach



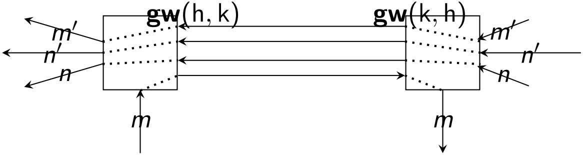


The "components-as-interfaces" (PaI) approach

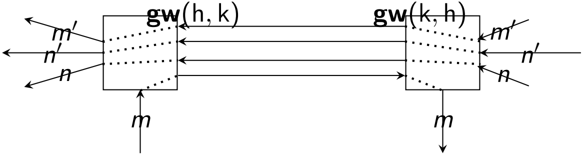


COMPATIBLE: an h's input is a k's output, and vice versa

The “components-as-interfaces” (PaI) approach



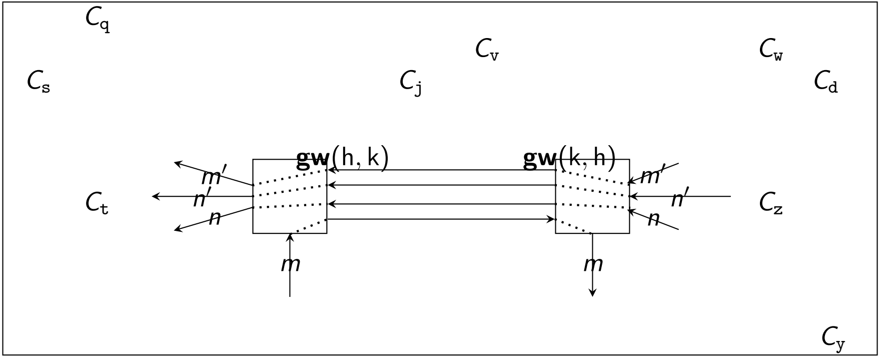
The “components-as-interfaces” (PaI) approach



Composition via gateways (forwarders)

The "components-as-interfaces" (PaI) approach

$$\underline{\mathbf{S_1}}\ {}^{\mathrm{J}\leftrightarrow\mathrm{K}}\ \underline{\mathbf{S_2}}$$



The “components-as-interfaces” (PaI) approach

- FLEXIBLE ✓
- SAFE ?

The “components-as-interfaces” (PaI) approach

- FLEXIBLE ✓
- SAFE ?

The "components-as-interfaces" (PaI) approach

- FLEXIBLE ✓
- SAFE ?

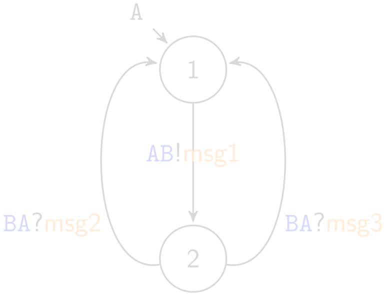
The "components-as-interfaces" (PaI) approach

- FLEXIBLE ✓
- SAFE ?

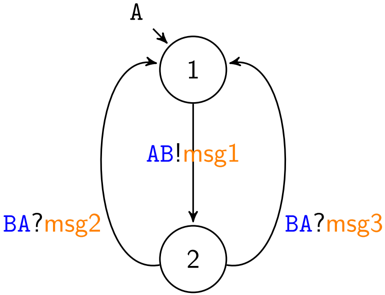
The "components-as-interfaces" (PaI) approach

- FLEXIBLE ✓
- SAFE ?

Investigating the PaI approach: **which formalism for participants' behaviours?**



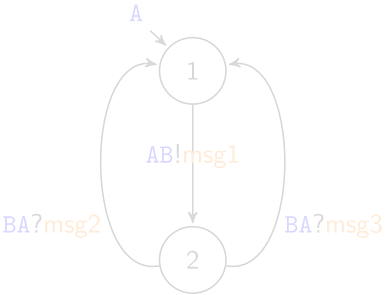
Investigating the PaI approach: **which formalism for participants' behaviours?**



Communicating Finite State Machines (CFSMs)

An **automata-based** formalism for the description and the analysis of distributed systems. [BRAND AND ZAFIROPULO, 1983]

A machine $M_{\mathtt{A}}$

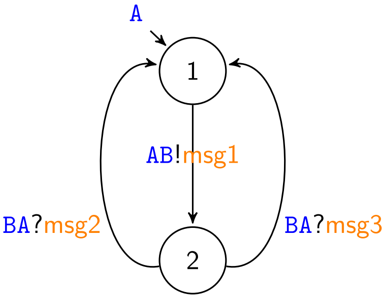


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
- and so on....

Communicating Finite State Machines (CFSMs)

An **automata-based** formalism for the description and the analysis of distributed systems. [Brand and Zafiropulo, 1983]

A machine $M_{\mathtt{A}}$

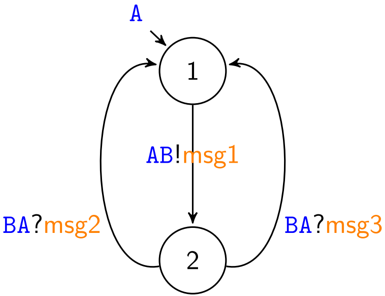


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
- and so on....

Communicating Finite State Machines (CFSMs)

An **automata-based** formalism for the description and the analysis of distributed systems. [Brand and Zafiropulo, 1983]

A machine $M_{\mathtt{A}}$

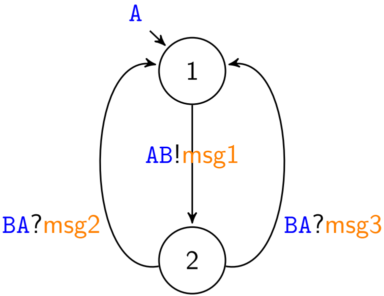


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
- and so on....

Communicating Finite State Machines (CFSMs)

An **automata-based** formalism for the description and the analysis of distributed systems. [Brand and Zafiropulo, 1983]

A machine $M_{\mathtt{A}}$

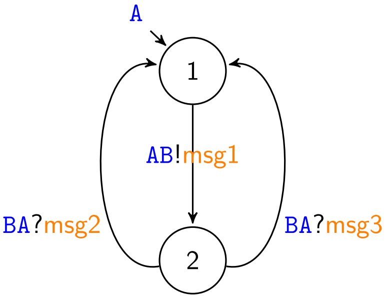


- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
- and so on....

Communicating Finite State Machines (CFSMs)

An **automata-based** formalism for the description and the analysis of distributed systems. [Brand and Zafiropulo, 1983]

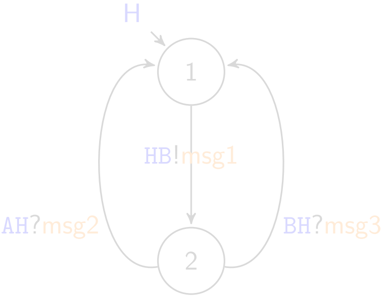
A machine $M_{\mathtt{A}}$



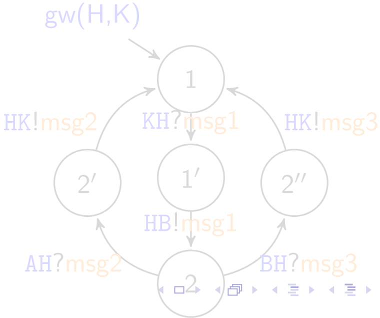
- $M_{\mathtt{A}}$ can send msg1 to machine $M_{\mathtt{B}}$;
- Then, either msg2 or msg3 can be received from $M_{\mathtt{B}}$;
- and so on....

The PaI approach for systems of CFSMs

An “interface” participant like

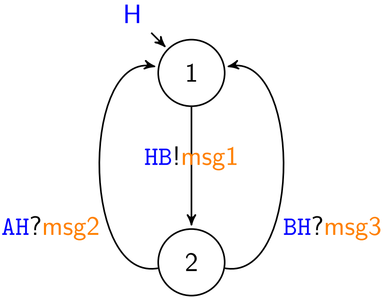


Is replaced by

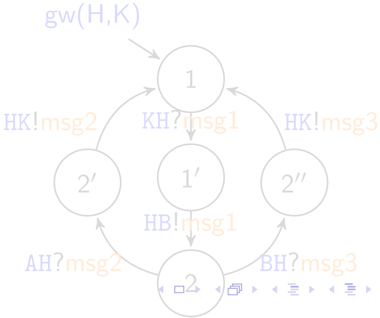


The PaI approach for systems of CFSMs

An “interface” participant like

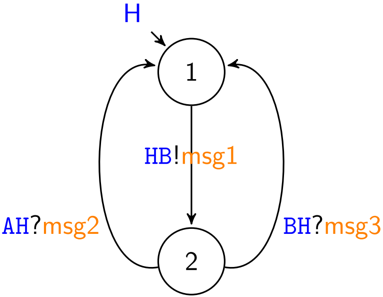


Is replaced by

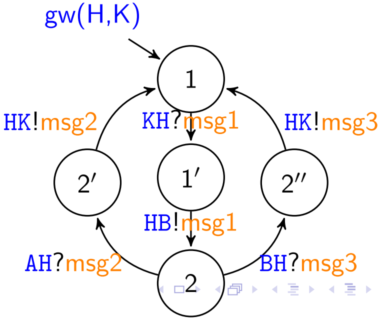


The PaI approach for systems of CFSMs

An "interface" participant like



Is replaced by



In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

In the investigation of PaI for systems of CFSMs

Which underlying interaction model?

- **asynchronous**; *through the directed buffered FIFO channels (also other possibilities)*
- **synchronous**; *also for this, there is not just one formalization*

PaI Composition of Systems of **asynchronous** CFSMs

Barbanera, de'Liguoro, Hennicker

> Connecting open systems of communicating finite state machines (JLAMP)

Several communication properties preserved by composition:

- deadlock freedom
- orphan message freedom
- unspecified reception
- progress

Required conditions on interfaces, besides **compatibility** (essentially bisimulation)

- **!(?)-determinism**: the message does uniquely determine the receiver(sender)
- **no-mixed-state**: from each state, either input or output actions, not both.

PaI Composition of Systems of **asynchronous** CFSMs

Barbanera, de'Liguoro, Hennicker

> Connecting open systems of communicating finite state machines (JLAMP)

Several communication properties preserved by composition:

- deadlock freedom
- orphan message freedom
- unspecified reception
- progress

Required conditions on interfaces, besides **compatibility** (essentially bisimulation)

- **!(?)-determinism**: the message does uniquely determine the receiver(sender)
- **no-mixed-state**: from each state, either input or output actions, not both.

PaI Composition of Systems of **asynchronous** CFSMs

Barbanera, de'Liguoro, Hennicker

> Connecting open systems of communicating finite state machines (JLAMP)

Several communication properties preserved by composition:

- deadlock freedom
- orphan message freedom
- unspecified reception
- progress

Required conditions on interfaces, besides **compatibility** (essentially bisimulation)

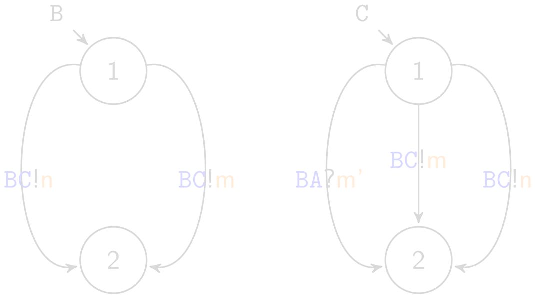
- **!(?)-determinism**: the message does uniquely determine the receiver(sender)
- **no-mixed-state**: from each state, either input or output actions, not both.

And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?
The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is “external” (“agreed upon”).



In a sense, in CCS style

$$+\mathtt{BC!n} \quad | \quad \mathtt{BC?m'} + +\mathtt{BC?n}$$

$$\xrightarrow{\tau} \quad \mathbf{0}$$

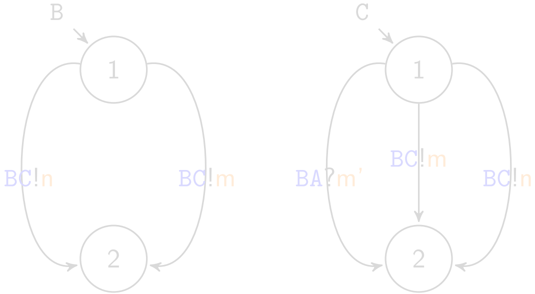
And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?

The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is "external" ("agreed upon").



In a sense, in CCS style

$$+\mathtt{BC!n} \quad | \quad \mathtt{BC?m'} + +\mathtt{BC?n}$$

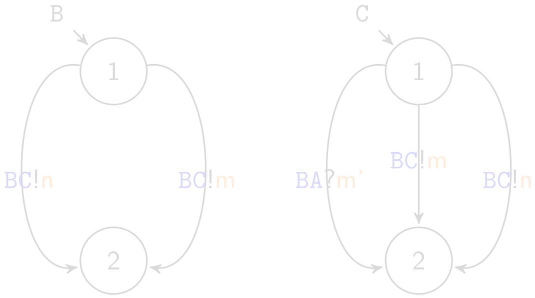
$$\xrightarrow{\tau} \quad \mathbf{0}$$

And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?
The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is “external” (“agreed upon”).



In a sense, in CCS style

$$+\mathtt{BC!n} \quad | \quad \mathtt{BC?m}' + +\mathtt{BC?n}$$

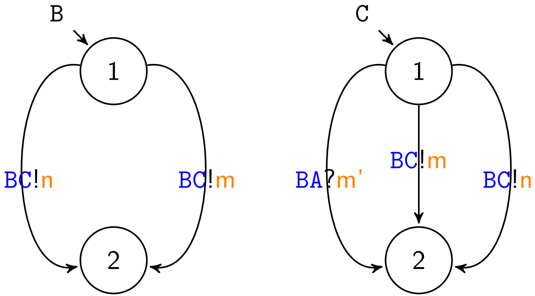
$$\xrightarrow{\tau} \quad \mathbf{0}$$

And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?
The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is “external” (“agreed upon”).



In a sense, in CCS style

$$\mathtt{BC!m} + \mathtt{BC!n} \quad | \quad \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n}$$

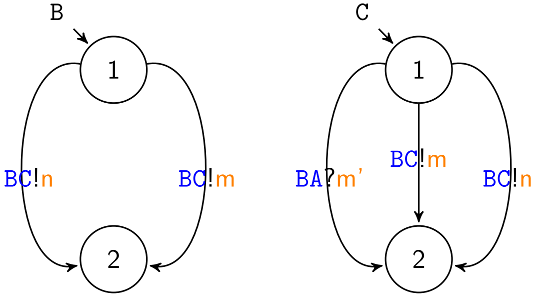
$\xrightarrow{\tau} \mathbf{0}$

And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?
The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is “external” (“agreed upon”).



In a sense, in CCS style

$$\underline{\mathtt{BC!m}} + \mathtt{BC!n} \quad | \quad \mathtt{BC?m'} + \underline{\mathtt{BC?m}} + \mathtt{BC?n}$$

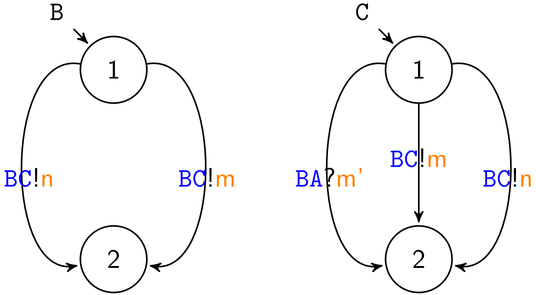
$$\xrightarrow{\tau} \quad \mathbf{0}$$

And for systems of **symmetric synchronous** CFSMs?

What is a synchronous communication (in the CFSM model)?
The **symmetric** approach:

> *sender and receiver play the same role in an interaction.*

Any choice is “external” (“agreed upon”).



In a sense, in CCS style

$$\underline{\mathtt{BC!m}} + \mathtt{BC!n} \quad \mid \quad \mathtt{BC?m'} + \underline{\mathtt{BC?m}} + \mathtt{BC?n}$$

$$\xrightarrow{\tau} \quad \mathbf{0}$$

Composing systems of **symmetric synchronous** CFSMs

Barbanera, Lanese, Tuosto

> COMPOSING COMMUNICATING SYSTEMS, SYNCHRONOUSLY. ISOLA (1) 2020

where

- Compatibility = Bisimulation (forgetting senders and receivers, and exchanging ‘!’ and ‘‘?’ on one side);
- **!?-determinism** and **no-mixed-state** still needed.

NOT enough!

Composing systems of **symmetric synchronous** CFSMs

Barbanera, Lanese, Tuosto

Composing Communicating Systems, Synchronously. ISoLA (1) 2020

where

- Compatibility = Bisimulation (forgetting senders and receivers, and exchanging '!' and ''?' on one side);
- **!?-determinism** and **no-mixed-state** still needed.

NOT enough!

Composing systems of **symmetric synchronous** CFSMs

Barbanera, Lanese, Tuosto

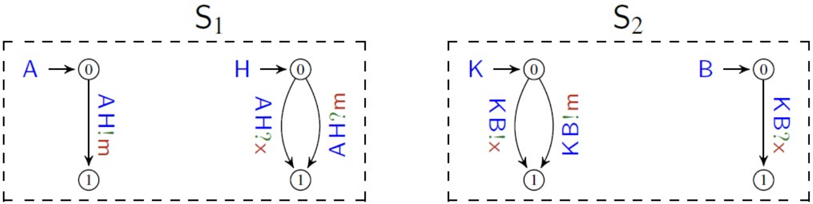
Composing Communicating Systems, Synchronously.
ISoLA (1) 2020

where

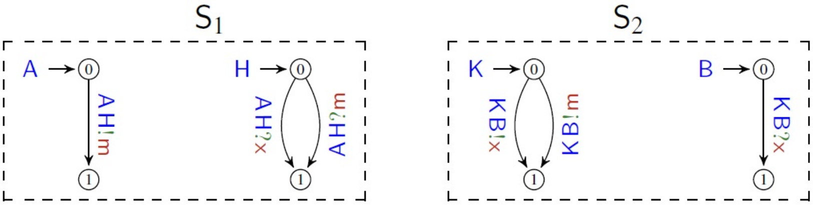
- Compatibility = Bisimulation (forgetting senders and receivers, and exchanging '!' and ''?' on one side);
- **!?-determinism** and **no-mixed-state** still needed.

NOT enough!

Composing systems of **symmetric synchronous** CFSMs

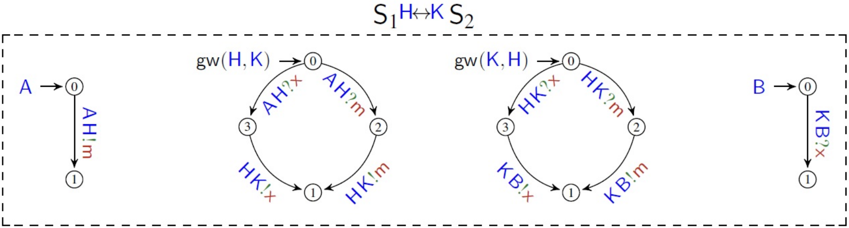


Composing systems of **symmetric synchronous** CFSMs



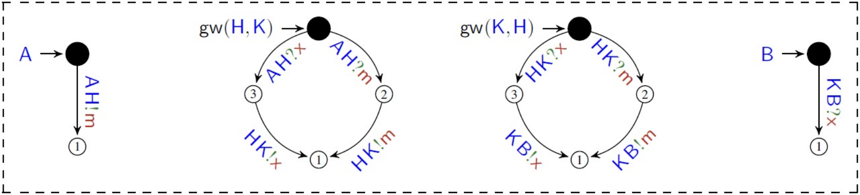
Both deadlock-free

Composing systems of **symmetric synchronous** CFSMs



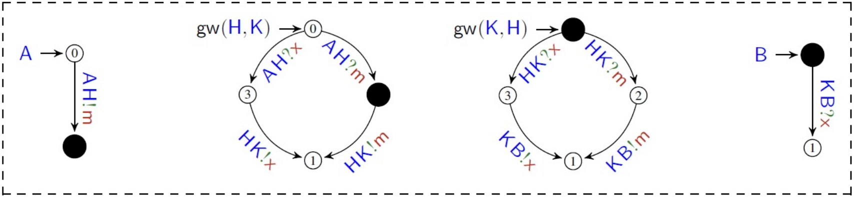
Composing systems of **symmetric synchronous** CFSMs

$S_1 {}^{H \leftrightarrow K} S_2$



Composing systems of **symmetric synchronous** CFSMs

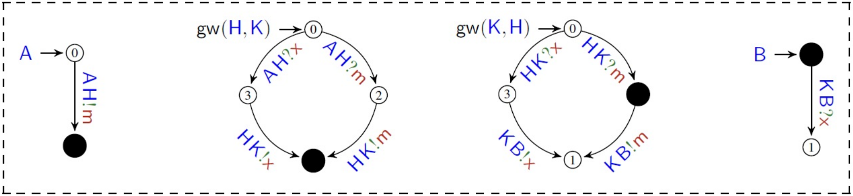
$S_1{}^{H \leftrightarrow K} S_2$



A→H: m

Composing systems of **symmetric synchronous** CFSMs

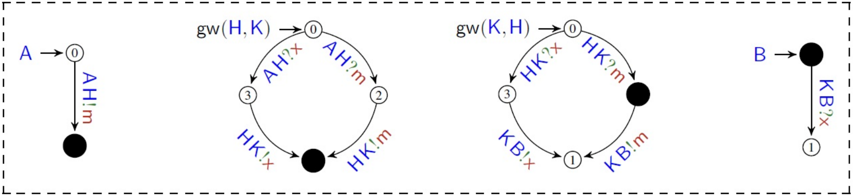
$S_1 \, {}^{H \leftrightarrow K} \, S_2$



$\mathtt{gw(H,K)} \rightarrow \mathtt{gw(K,H)}$: m

Composing systems of **symmetric synchronous** CFSMs

$S_1 {}^{H \leftrightarrow K} S_2$



Deadlock!

Composing systems of **symmetric synchronous** CFSMs

Definition

A CFSM A is

1. is **sequential** if each state has **at most** one outgoing transition.
2. is **!-live** if, for any reachable configuration s: any output action A can perform occurs in a continuation of the system. Formally

$$s(\mathsf{A}) \xrightarrow{\mathsf{A\,B!m}} \quad \textit{implies } s \to^* s' \xrightarrow{\mathsf{A \to B:\ m}} \quad \textit{for some } s'$$

Theorem

Deadlock-freedom preservation by composition when interfaces (and hence gateways) are **also** *either sequential or !-live.*

Composing systems of **symmetric synchronous** CFSMs

Definition

A CFSM A is

1. is **sequential** if each state has **at most** one outgoing transition.
2. is **!-live** if, for any reachable configuration s: any output action A can perform occurs in a continuation of the system. Formally

$$s(\mathsf{A}) \xrightarrow{\mathsf{A\,B!m}} \quad \textit{implies } s \to^* s' \xrightarrow{\mathsf{A \rightarrow B:\ m}} \quad \textit{for some } s'$$

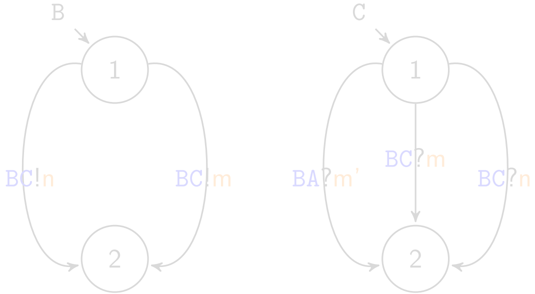
Theorem

Deadlock-freedom preservation by composition when interfaces (and hence gateways) are **also** *either sequential or !-live.*

Asymmetric synchronous interactions

Sender and receiver play different roles in choice resolution while still relying on "handshakes"

In particular: Choices of outputs are "internal" ("sender chooses").

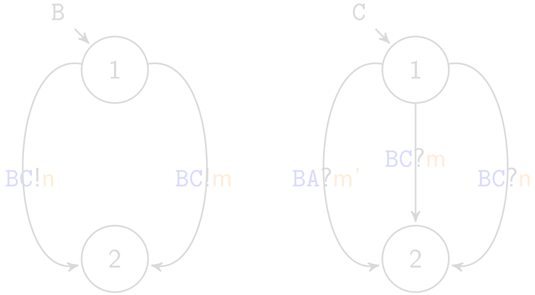


In a sense, interpreted as

Asymmetric synchronous interactions

> *Sender and receiver play different roles in choice resolution while still relying on "handshakes"*

In particular: Choices of outputs are "internal" ("sender chooses").

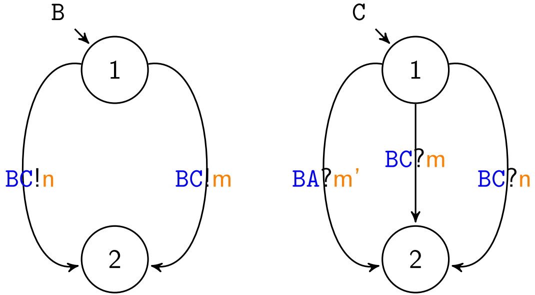


In a sense, interpreted as

Asymmetric synchronous interactions

Sender and receiver play different roles in choice resolution while still relying on "handshakes"

In particular: Choices of outputs are "internal" ("sender chooses").



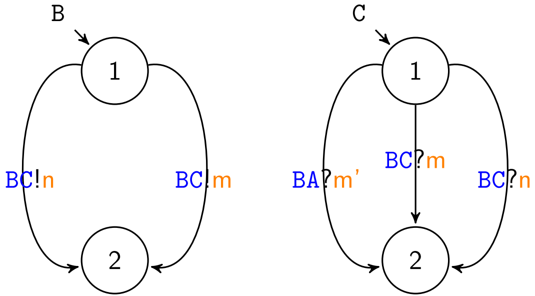
In a sense, interpreted as

$$\mathtt{BC!m} \oplus \mathtt{BC!n} \mid \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n}$$

Asymmetric synchronous interactions

Sender and receiver play different roles in choice resolution while still relying on "handshakes"

In particular: Choices of outputs are "internal" ("sender chooses").



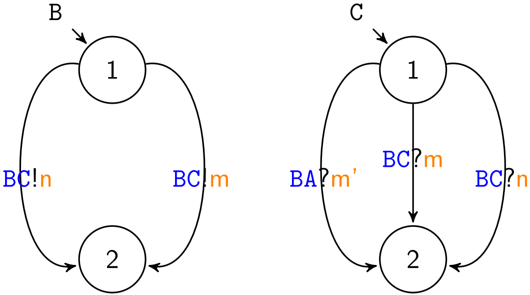
In a sense, interpreted as

$\underline{\mathtt{BC!m}} \oplus \mathtt{BC!n} \mid \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n}$

Asymmetric synchronous interactions

Sender and receiver play different roles in choice resolution while still relying on "handshakes"

In particular: Choices of outputs are "internal" ("sender chooses").



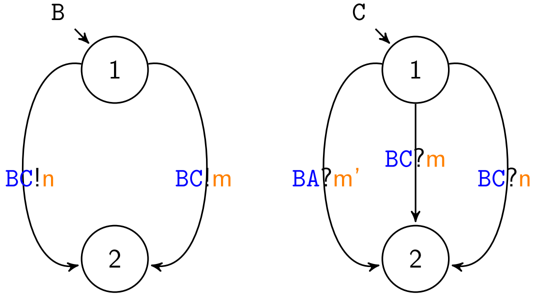
In a sense, interpreted as

$$\begin{array}{lll} & \mathtt{BC!m} \oplus \mathtt{BC!n} & \mid \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n} \\ \xrightarrow{\oplus} & \mathtt{BC!m} & \mid \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n} \end{array}$$

Asymmetric synchronous interactions

> *Sender and receiver play different roles in choice resolution while still relying on "handshakes"*

In particular: Choices of outputs are "internal" ("sender chooses").



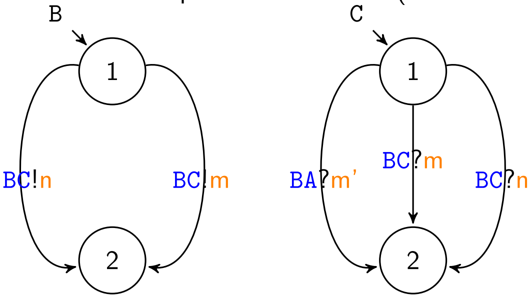
In a sense, interpreted as

$$
\begin{array}{rlcl}
 & \mathtt{BC!m} \oplus \mathtt{BC!n} & \mid & \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n} \\
\xrightarrow{\oplus} & \mathtt{BC!m} & \mid & \mathtt{BC?m'} + \underline{\mathtt{BC?m}} + \mathtt{BC?n}
\end{array}
$$

Asymmetric synchronous interactions

Sender and receiver play different roles in choice resolution while still relying on "handshakes"

In particular: Choices of outputs are "internal" ("sender chooses").



In a sense, interpreted as

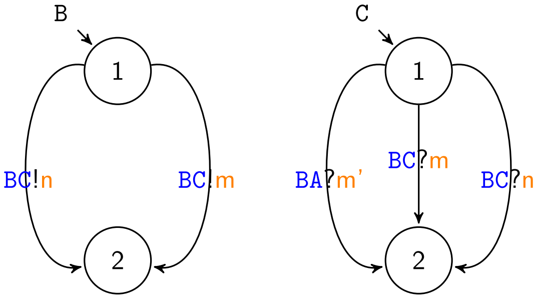
$$\begin{array}{rlcl} & \mathtt{BC!m} \oplus \mathtt{BC!n} & | & \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n} \\ \xrightarrow{\oplus} & \mathtt{BC!m} & | & \mathtt{BC?m'} + \mathtt{BC?m} + \mathtt{BC?n} \\ \xrightarrow{\tau} & \mathbf{0} & & \end{array}$$

Formalising asymmetric synchronous interactions for CFSMs

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.

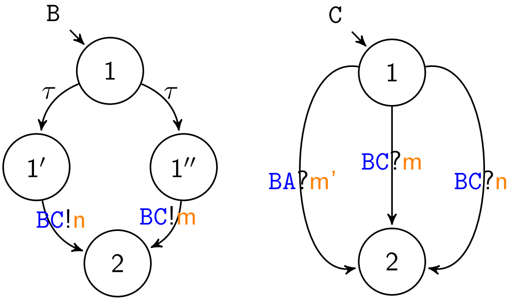
Formalising asymmetric synchronous interactions for CFSMs

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



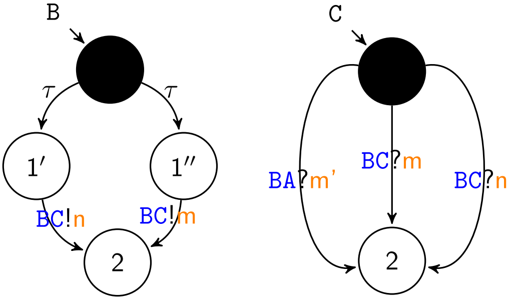
Formalising asymmetric synchronous interactions for CFSMs

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



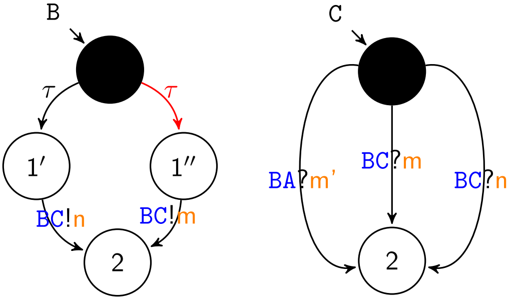
Asymmetric synchronous interactions

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



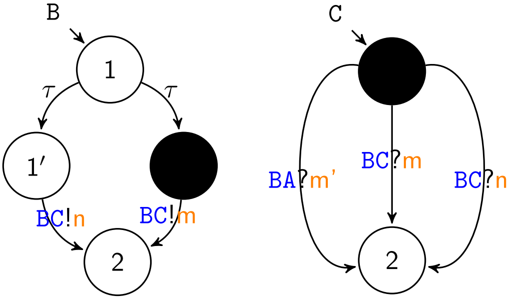
Asymmetric synchronous interactions

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



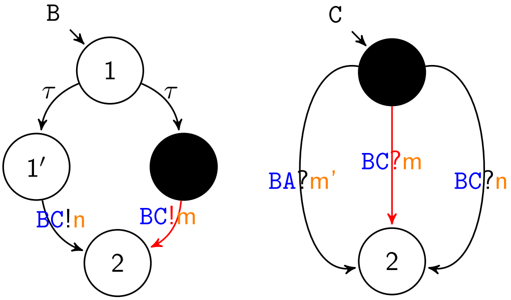
Asymmetric synchronous interactions

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



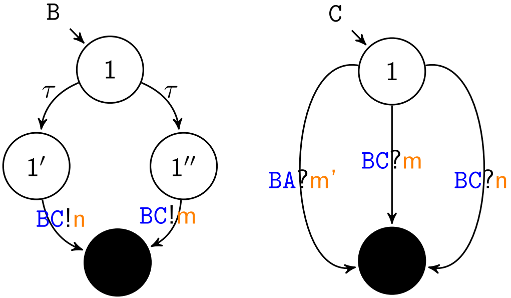
Asymmetric synchronous interactions

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



Asymmetric synchronous interactions

We can use the symmetric model of synchronous interactions prefixing any output with silent actions.



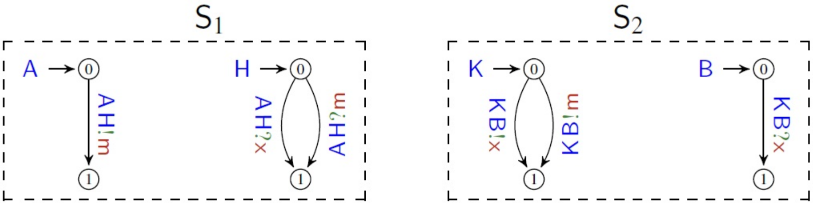
PaI with **Asymmetric synchronous** interactions

Barbanera, Lanese, Tuosto

ICE 2022

Composition with **Asymmetric synchronous** interactions

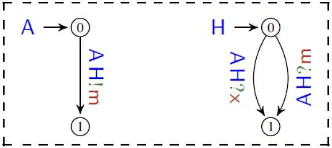
Counterexample for symmetric synchronous interactions does not apply



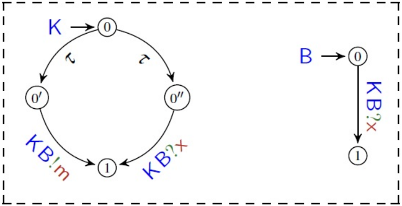
Composition with **Asymmetric synchronous** interactions

Counterexample for symmetric synchronous interactions does not apply

S_1

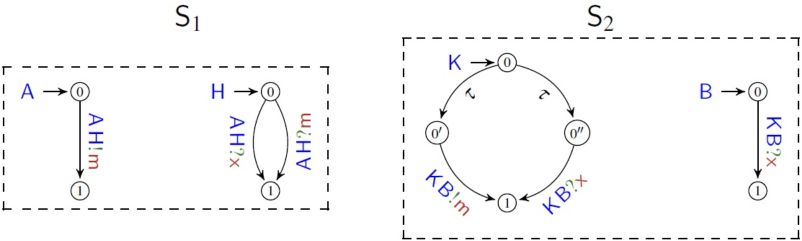


S_2



Composition with **Asymmetric synchronous** interactions

Counterexample for symmetric synchronous interactions does not apply



S_2 is not deadlock-free

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom (***sequentiality required!***).*

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds...

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom (**sequentiality required!**).*

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds...

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom* **(sequentiality required!**).

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds...

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom (***sequentiality required!***).*

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds...

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom (***sequentiality required!***).*

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds...

Composition with **Asymmetric synchronous** interactions

Definition

Deadlock-freedom: when the system cannot proceed, no participant is willing to proceed;
Lock-freedom: when a participant is willing to proceed, the system can allow that in some of its continuations;
Strong lock-freedom: when a participant is willing to proceed, the system allows that in any of its continuations.

Theorem

For !?-deterministic, no mixed states and compatible interfaces, composition preserves

- *deadlock-freedom (in a sense it implies !-liveness);*
- *strong lock-freedom;*
- *lock-freedom (***sequentiality required!***).*

Proof Essentially, a deadlock/lock/strong-lock in the composed system "corresponds" to a deadlock/lock/strong-lock in one of the two systems we started with. Unfortunately cannot be shown trivially as it sounds....

Loosening Compatibility=Bisimilarity

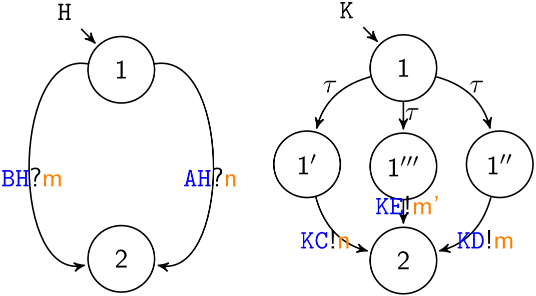
For the previous result also the interface-participants below can be deemed compatible.

Loosening Compatibility=Bisimilarity

For the previous result also the interface-participants below can be deemed compatible.

Loosening Compatibility=Bisimilarity

For the previous result also the interface-participants below can be deemed compatible.



Some pieces of the mosaic still missing

- lock/strong-lock freedom still to be investigated for symmetric synchronous interactions
- Loose compatibility for asynchronous and symmetric synchronous interactions. (Almost immediate, we guess). Can it be made looser?
- Composition using multiple interfaces

Some pieces of the mosaic still missing

- lock/strong-lock freedom still to be investigated for symmetric synchronous interactions
- Loose compatibility for asynchronous and symmetric synchronous interactions. (Almost immediate, we guess). Can it be made looser?
- Composition using multiple interfaces

Some pieces of the mosaic still missing

- lock/strong-lock freedom still to be investigated for symmetric synchronous interactions
- Loose compatibility for asynchronous and symmetric synchronous interactions. (Almost immediate, we guess). Can it be made looser?
- Composition using multiple interfaces

Some pieces of the mosaic still missing

- lock/strong-lock freedom still to be investigated for symmetric synchronous interactions
- Loose compatibility for asynchronous and symmetric synchronous interactions. (Almost immediate, we guess). Can it be made looser?
- Composition using multiple interfaces

The need of systems composability

Composability is useful both

- at design phase (modular design);
  Application of PaI for Multi-Party Session Types

- at deployment phase and beyond OUR SETTING

Thank you for your attention.