

## 8. Behavioural equivalences

---

David Pereira   José Proença

RAMDE 2021/2022


Requirements and Model-driven Engineering

CISTER – ISEP

Porto, Portugal

<https://cister-labs.github.io/ramde2122>

## Recall

1. Non-deterministic Finite Automata: 
2. Process algebra:  $P = a.Q$      $Q = b.Q$      $P|Q$
3. Interaction between processes
4. Meaning of PA using NFA

## Still missing

- When is a process  $P$  **equivalent** to a process  $Q$ ?
- When can a process  $P$  be **safely replaced** by a process  $Q$ ?
- When can a sequence of interactions be **safely implemented** as interacting components?

- High-level overview or requirements and associated processes
- Mathematical Preliminaries
  - Basic mathematical notations
  - Set theory
  - Propositional Logic
  - First Order Logic
  - The Z3 automatic theorem prover
- Behavioural modelling
  - Single component
  - Many components
  - Equivalences
    - Language Equivalence
    - (Bi)similarity
    - Realisability
  - Verification

# Behavioural Equivalences – Intuition

Two automata (or LTS) should be **equivalent** if they cannot be distinguished by interacting with them.

## Equality of functional behaviour

is not preserved by **parallel** composition: non **compositional** semantics, cf,

`x:=4; x:=x+1` and `x:=5`

## Graph isomorphism

is too strong (why?)

## EQ1 – Language equivalence

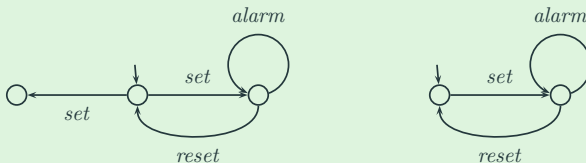
---

# Language equivalence

## Definition

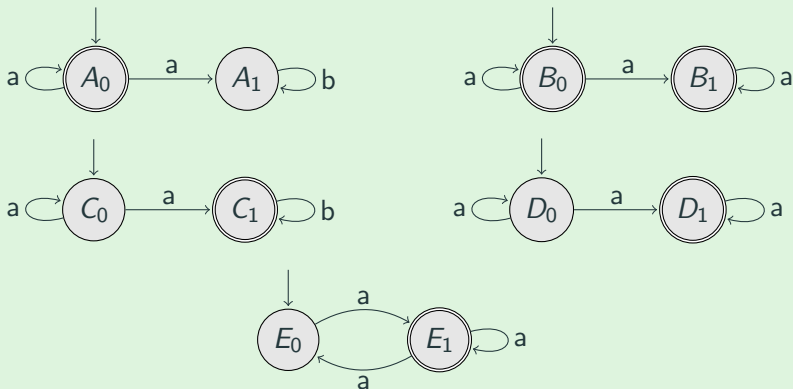
Two automata  $A, B$  are **language equivalent** iff  $L_A = L_B$   
(i.e. if they can perform the same finite sequences of transitions)

## Example



**Language equivalence** applies when one can neither interact with a system, nor distinguish a slow system from one that has come to a stand still.

## Ex. 8.1: Find pairs of automata with the same language



## EQ2 – Similarity

---



the quest for a **behavioural equality**:  
able to identify states that cannot be distinguished by any **realistic** form of observation

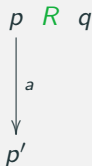
## Simulation

A state  $q$  **simulates** another state  $p$  if  
every transition from  $q$  is corresponded by a transition from  $p$  **and**  
this capacity is kept along the whole life of the system to which state space  $q$  belongs  
to.

## Definition

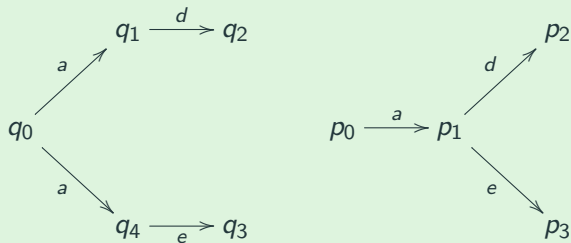
Given  $\langle S_1, N, \longrightarrow_1 \rangle$  and  $\langle S_2, N, \longrightarrow_2 \rangle$  over  $N$  (ignoring initial and final states) a relation  $R \subseteq S_1 \times S_2$  is a **simulation** iff, for all  $\langle p, q \rangle \in R$  and  $a \in N$ ,

$$(1) \quad p \xrightarrow{a}_1 p' \Rightarrow \langle \exists q' : q' \in S_2 : q \xrightarrow{a}_2 q' \wedge \langle p', q' \rangle \in R \rangle$$



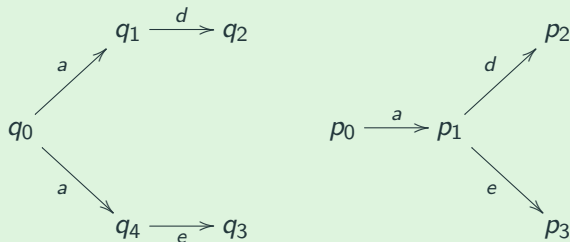
# Example

## Ex. 8.2: Find simulations



# Example

## Ex. 8.2: Find simulations



$$q_0 \lesssim p_0 \quad \text{cf.} \quad \{\langle q_0, p_0 \rangle, \langle q_1, p_1 \rangle, \langle q_4, p_1 \rangle, \dots\}$$

## Definition

$$p \lesssim q \equiv \langle \exists R :: R \text{ is a simulation and } \langle p, q \rangle \in R \rangle$$

We say *p is simulated by q*.

## Lemma

The similarity relation is a preorder  
(ie, reflexive and transitive)

## EQ3 – Bisimilarity

---

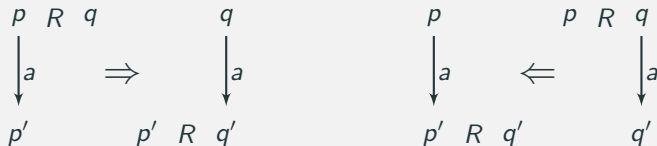
## Definition

Given  $\langle S_1, N, \longrightarrow_1 \rangle$  and  $\langle S_2, N, \longrightarrow_2 \rangle$  over  $N$ , relation  $R \subseteq S_1 \times S_2$  is a **bisimulation** iff both  $R$  and its converse  $R^\circ$  are simulations.

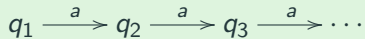
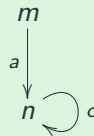
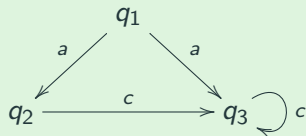
I.e., whenever  $\langle p, q \rangle \in R$  and  $a \in N$ ,

$$(1) \quad p \xrightarrow{a}_1 p' \Rightarrow \langle \exists q' : q' \in S_2 : q \xrightarrow{a}_2 q' \wedge \langle p', q' \rangle \in R \rangle$$

$$(2) \quad q \xrightarrow{a}_2 q' \Rightarrow \langle \exists p' : p' \in S_1 : p \xrightarrow{a}_1 p' \wedge \langle p', q' \rangle \in R \rangle$$



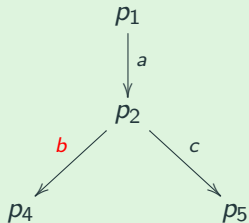
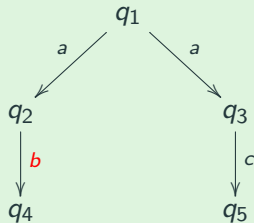
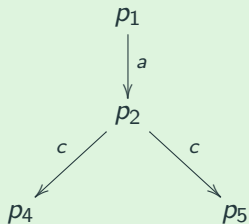
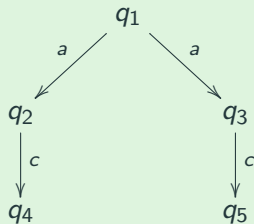
Ex. 8.3: Find bisimulations that include  $q_1$





# Examples

Ex. 8.4: Find bisimulations that include  $q_1$



## Definition

$$p \sim q \equiv \langle \exists R :: R \text{ is a bisimulation and } \langle p, q \rangle \in R \rangle$$

We say *p is bisimilar to q*.

## Lemma

Two processes  $P$  and  $Q$  are bisimilar if there is a bisimulation that includes  $\langle P, Q \rangle$ .

## Warning

$\left[ p \lesssim q \text{ and } q \lesssim p \right]$  does **not** imply  $\left[ p \sim q \right]$

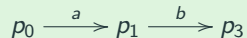
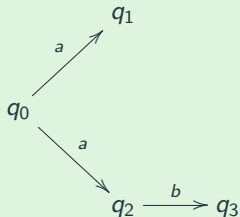
# Properties

## Warning

$[p \lesssim q \text{ and } q \lesssim p]$  does **not** imply  $[p \sim q]$

## Example

$q_0 \lesssim p_0, p_0 \lesssim q_0$  but  $p_0 \not\sim q_0$



## Similarity as the greatest simulation

$$\lesssim \triangleq \bigcup \{S \mid S \text{ is a simulation}\}$$

## Bisimilarity as the greatest bisimulation

$$\sim \triangleq \bigcup \{S \mid S \text{ is a bisimulation}\}$$

## Ex. 8.5: P,Q Bisimilar?

$$P = a.P_1$$

$$P_1 = b.P + c.P$$

$$Q = a.Q_1$$

$$Q_1 = b.Q_2 + c.Q$$

$$Q_2 = a.Q_3$$

$$Q_3 = b.Q + c.Q_2$$

## Ex. 8.6: P,Q Bisimilar?

$$P = a.(b.0 + 0)$$

$$Q = a.b.0$$

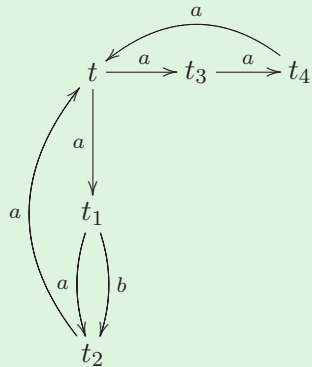
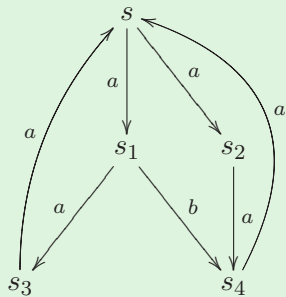
## Ex. 8.7: P,Q Bisimilar?

$$P = a.(b.0 + c.0)$$

$$Q = a.b.0 + a.c.0$$

Draw their LTS. If bisimilar, find the bisimulation.

## Ex. 8.8: Find a bisimulation



## Weak bisimilarity

---



## Considering $\tau$ -transitions

### Weak transition

$$\begin{aligned} p \xRightarrow{\alpha} q & \text{ iff } p (\xrightarrow{\tau})^* q_1 \xrightarrow{a} q_2 (\xrightarrow{\tau})^* q \\ p \xRightarrow{\tau} q & \text{ iff } p (\xrightarrow{\tau})^* q \end{aligned}$$

where  $\alpha \neq \tau$  and  $(\xrightarrow{\tau})^*$  is the reflexive and transitive closure of  $\xrightarrow{\tau}$ .

### Weak bisimulation (vs. strong)

Given  $\langle S_1, N, \xrightarrow{\phantom{a}}_1 \rangle$  and  $\langle S_2, N, \xrightarrow{\phantom{a}}_2 \rangle$  over  $N$ , relation  $R \subseteq S_1 \times S_2$  is a **bisimulation** iff for all  $\langle p, q \rangle \in R$  and  $a \in N \cup \{\tau\}$ ,

$$(1) \quad p \xrightarrow{a}_1 p' \Rightarrow \langle \exists q' : q' \in S_2 : q \xRightarrow{a}_2 q' \wedge \langle p', q' \rangle \in R \rangle$$

$$(2) \quad q \xrightarrow{a}_2 q' \Rightarrow \langle \exists p' : p' \in S_1 : p \xRightarrow{a}_1 p' \wedge \langle p', q' \rangle \in R \rangle$$

## Considering $\tau$ -transitions

### Branching bisimulation

Given  $\langle S_1, N, \longrightarrow_1 \rangle$  and  $\langle S_2, N, \longrightarrow_2 \rangle$  over  $N$ , relation  $R \subseteq S_1 \times S_2$  is a **bisimulation** iff for all  $\langle p, q \rangle \in R$  and  $a \in N \cup \{\tau\}$ ,

(1) if  $p \xrightarrow{a}_1 p'$  then either

(1.1)  $a = \tau$  and  $\langle p', q \rangle \in R$  or

(1.2)  $\langle \exists q', q'' \in S_2 :: q (\xrightarrow{\tau}_2)^* q' \xrightarrow{a}_2 q'' \wedge \langle p, q' \rangle \in R \wedge \langle p', q'' \rangle \in R \rangle$

(2) if  $q \xrightarrow{a}_2 q'$  then either

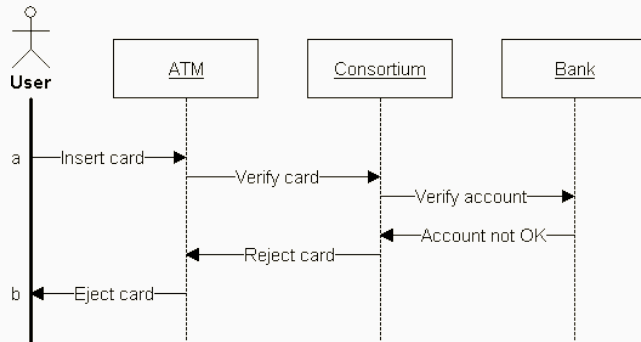
(2.1)  $a = \tau$  and  $\langle p', q' \rangle \in R$  or

(2.2)  $\langle \exists p', p'' \in S_1 :: p (\xrightarrow{\tau}_1)^* p' \xrightarrow{a}_1 p'' \wedge \langle p', q \rangle \in R \wedge \langle p'', q' \rangle \in R \rangle$

# Realisability of Sequence Diagrams

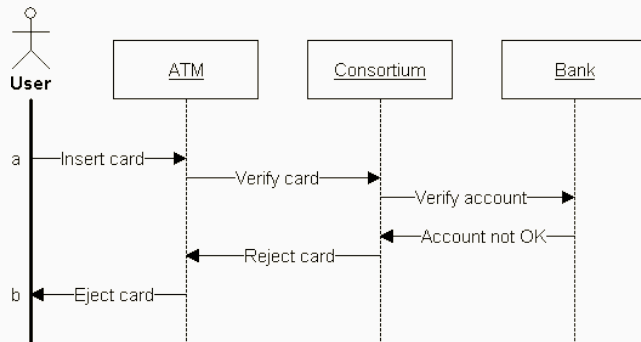
---

## Recall: Sequence Diagrams as Interactive Processes



- **Objects** as **Processes**  
(e.g., processes  $U$ ,  $A$ ,  $C$ ,  $B$ )
- **Send** actions (e.g., *insertCard*)
- **Receive** actions (e.g.,  $\overline{\text{insertCard}}$ )
- Unique action for each object pair
- Do not write  $(\dots + \mathbf{0})$

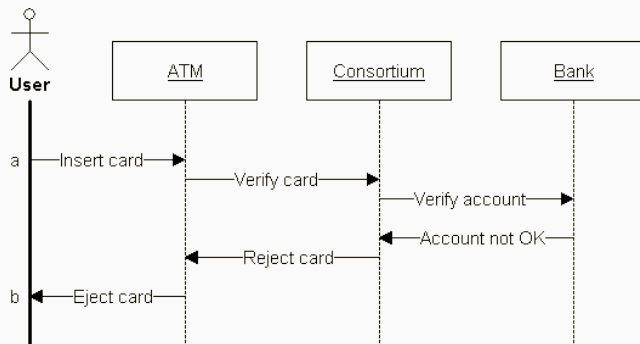
## Recall: Language of Sequence Diagrams, Informally



**This example has only 1 word and its prefixes**

$$L_{sd} = \text{insertCard} \cdot \text{verifyCard} \cdot \text{verifyAccount} \cdot \text{accountNotOK} \cdot \\ \text{rejectedCard} \cdot \text{ejectCard}$$

## Recall: Sequence Diagrams as Interactive Processes



We can specify a SD as interactive processes

$$\begin{aligned} \text{Sys} &= (U|A|C|E) \backslash \dots \\ U &= \text{insertCard}.\overline{\text{ejectCard}}.0 \\ A &= \dots \\ C &= \dots \\ E &= \dots \end{aligned}$$

# Sequence Diagrams covered by Interactive Processes

- **Sequence diagrams** depict **scenarios**  
(possible sequence of actions)
- **Processes** abstract **implementations**  
(simplified view of concrete implementations)

## Processes can do more

E.g., an ATM that also *accepts* cards can (and should) still support the *rejection* scenario.

# Observing the interactions

We want to **observe** interactions in such processes

## Modified CCS semantics

$$\begin{array}{c} \text{(com1)} \\ \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \\ \text{(com2)} \\ \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'} \\ \text{(com3)} \\ \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\bar{a}} Q'}{P|Q \xrightarrow{\tau_a} P'|Q'} \end{array}$$

$\alpha \in N \cup \bar{N} \cup \{\tau_a \mid a \in N\}$  is an action



Recall  $Sys$  from Slide 24 and its diagram  $sd$ .

$$L_{sd} = \{iC \cdot vC \cdot cA \cdot aN \cdot rC \cdot eC\}$$

$$L_{Sys} = \{\tau_{iC} \cdot \tau_{vC} \cdot \tau_{cA} \cdot \tau_{aN} \cdot \tau_{rC} \cdot \tau_{eC}\}$$

## Language inclusion

$P$  includes  $sd$

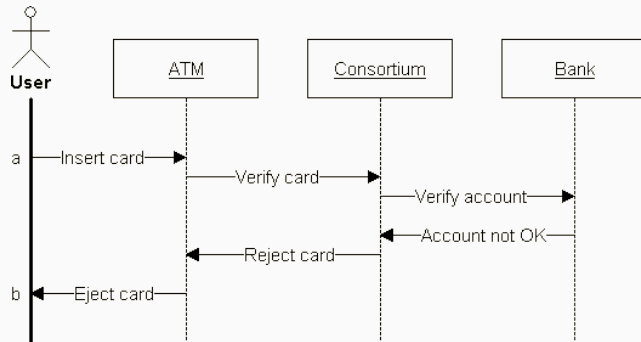
iff

$$L_{sd} \subseteq L_{P^\dagger}$$

$P^\dagger$  modifies  $P$ 's LTS by:

filtering actions of  $sd$  and replacing  $\tau_a$  by  $a$

## Are words enough?



**Ex. 8.9:** Let  $sd$  be the diagram above and recall Slide 24

Does  $Sys$  still includes  $sd$  if  $U$  is instead defined as below?

1.  $U = insertCard.\overline{ejectCard}.0 + insertCard.0$
2.  $U = (insertCard.\overline{ejectCard}.0) + goAway.0$

# Is language coverage enough?

## Implementations can have:

- extra undesirable behaviour
- less behaviour

## Alternative: change the inclusion/equivalence

Let  $SD = \{sd_1, sd_2, \dots\}$  be a set of sequence diagrams.

Language inclusion:  $L_{SD} \subseteq L_{P^\dagger}$

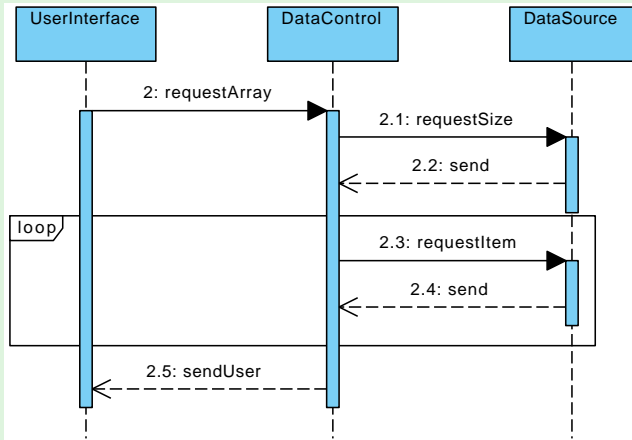
Language equivalence:  $L_{SD} = L_{P^\dagger}$

Similarity:  $NFA(SD) \lesssim P^\dagger$

Bisimilarity:  $NFA(SD) \sim P^\dagger$

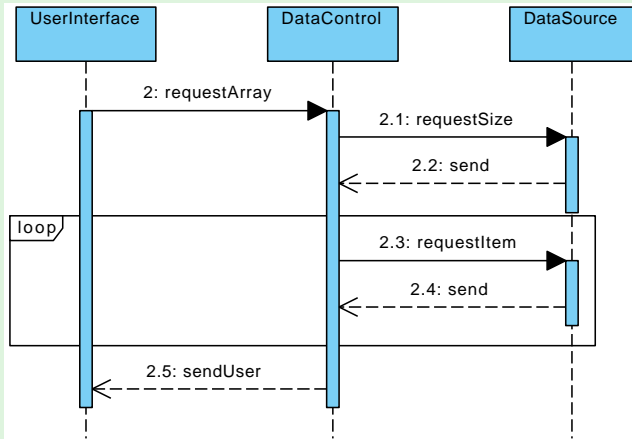
# Exercise

## Ex. 8.10: Draw an NFA that captures the following diagram



# Exercise

## Ex. 8.11: Write a process for each object of the diagram



# Realisability

**Question:** after encoding SD into processes:

Can we recover the behaviour of the original sequence diagram  
by composing  
the encoded processes?

## Realisability

A set SD of sequence diagrams is **realisable**

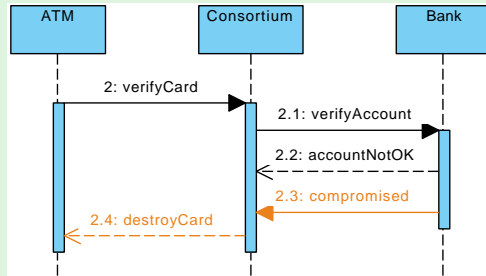
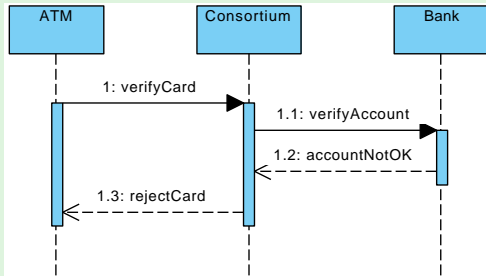
iff

$$\text{NFA}(SD) \sim \text{Comp}(\text{Proc}(SD))^{\dagger}$$

$\text{Proc}(SD)$  returns the set of encoded processes for each  $sd \in SD$

$\text{Comp}(P_1, P_2, \dots) = (P_1 | P_2 | \dots) \setminus \{\text{actions of } SD\}$

## Ex. 8.12: Are the diagrams below realisable?



1. draw  $NFA(SD)$
2. calculate  $Proc(SD)$   
Hint:  $B = \overline{vA}.(aN.0 + aN.c.0)$
3. draw  $Comp(\cdot)$
4. search for a bisimulation

**Ex. 8.13: Verify if the diagram in Slide 31 is realisable.**

# Exercise

## Ex. 8.14: Verify if the diagram is realisable.

