

# Transition Systems

Luca Tesei

Reactive Systems Verification

MSc in Computer Science

University of Camerino

## Topics

- Transition Systems

## Material

Reading:

Chapter 2 of the book, Section 2.1.

More:

The slides in the following pages are taken from the material of the course “Introduction to Model Checking” held by Prof. Dr. Ir. Joost-Pieter Katoen at Aachen University.

Introduction

## Modelling parallel systems

Transition systems



Modeling hard- and software systems

Parallelism and communication

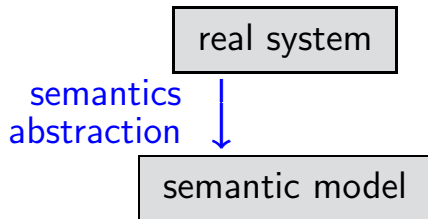
Linear Time Properties

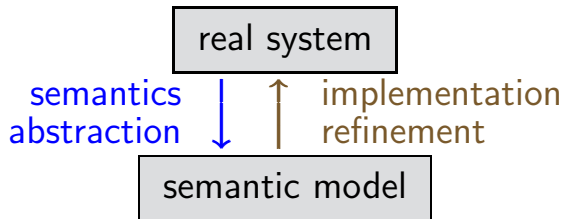
Regular Properties

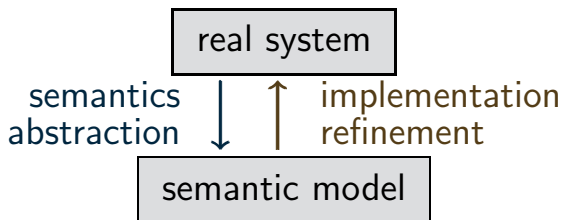
Linear Temporal Logic

Computation-Tree Logic

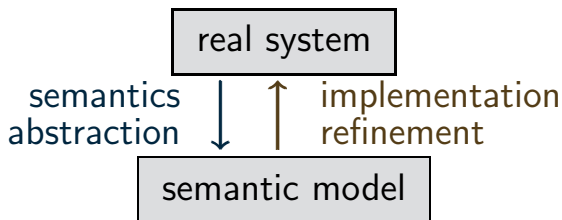
Equivalences and Abstraction





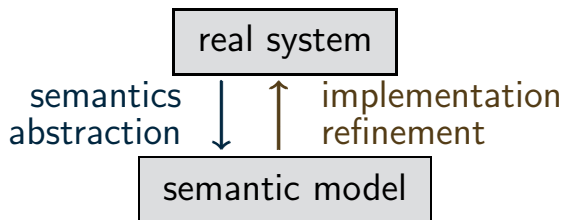


The semantic model yields a formal representation of:



The semantic model yields a formal representation of:

- the **states** of the system
- the **stepwise behaviour**
- the **initial states**



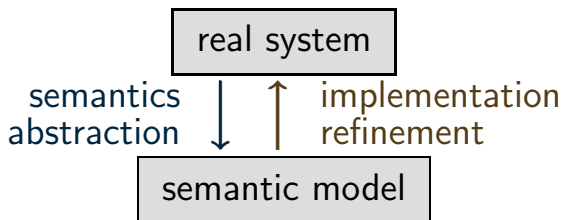
The semantic model yields a formal representation of:

- the **states** of the system



control component + information on “relevant” data

- the **stepwise behaviour**
- the **initial states**



The semantic model yields a formal representation of:

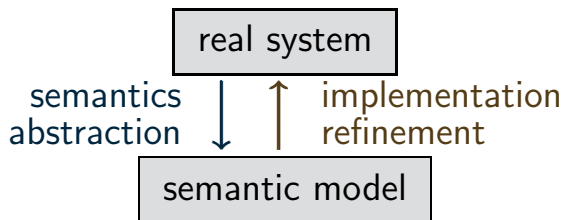
- the **states** of the system ← **nodes**



control component + information on “relevant” data

- the **stepwise behaviour** ← **edges**
- the **initial states**





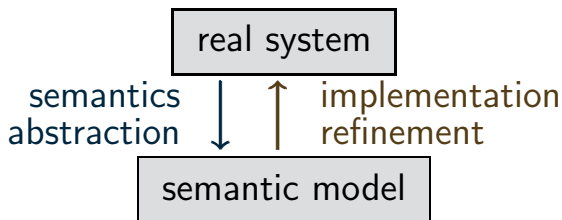
The semantic model yields a formal representation of:

- the **states** of the system ← **nodes**



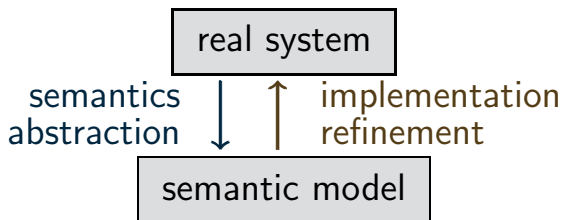
control component + information on “relevant” data

- the **stepwise behaviour** ← **transitions**
- the **initial states**



The semantic model yields a formal representation of:

- the **states** of the system ← **nodes**
- the **stepwise behaviour** ← **transitions**
- the **initial states**
- **additional information** on  
communication  
state properties



The semantic model yields a formal representation of:

- the **states** of the system ← **nodes**
- the **stepwise behaviour** ← **transitions**
- the **initial states**
- **additional information** on
  - communication ← **actions**
  - state properties ← **atomic proposition**

# Transition system (TS)

TS1.4-TS-DEF

A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \text{Act}, \longrightarrow, \mathcal{S}_0, AP, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,

A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,
- $\mathit{Act}$  is a set of **actions**,

A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,
- $\mathit{Act}$  is a set of **actions**,
- $\longrightarrow \subseteq \mathcal{S} \times \mathit{Act} \times \mathcal{S}$  is the transition relation,

A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,
- $\mathit{Act}$  is a set of **actions**,
- $\longrightarrow \subseteq \mathcal{S} \times \mathit{Act} \times \mathcal{S}$  is the transition relation,

i.e., transitions have the form  $s \xrightarrow{\alpha} s'$   
where  $s, s' \in \mathcal{S}$  and  $\alpha \in \mathit{Act}$



A transition system is a tuple

$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,
- $\mathit{Act}$  is a set of **actions**,
- $\longrightarrow \subseteq \mathcal{S} \times \mathit{Act} \times \mathcal{S}$  is the transition relation,

i.e., transitions have the form  $s \xrightarrow{\alpha} s'$   
where  $s, s' \in \mathcal{S}$  and  $\alpha \in \mathit{Act}$

- $\mathcal{S}_0 \subseteq \mathcal{S}$  the set of **initial states**,

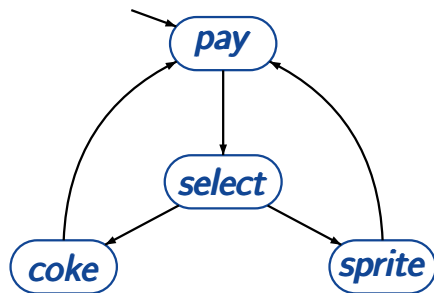
A transition system is a tuple

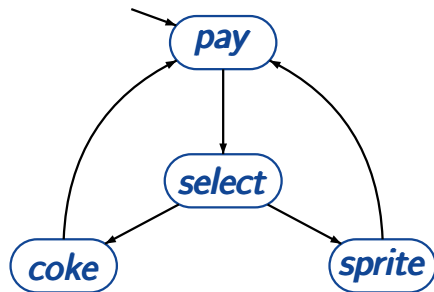
$$\mathcal{T} = (\mathcal{S}, \mathit{Act}, \longrightarrow, \mathcal{S}_0, \mathit{AP}, L)$$

- $\mathcal{S}$  is the state space, i.e., set of **states**,
- $\mathit{Act}$  is a set of **actions**,
- $\longrightarrow \subseteq \mathcal{S} \times \mathit{Act} \times \mathcal{S}$  is the transition relation,

i.e., transitions have the form  $s \xrightarrow{\alpha} s'$   
where  $s, s' \in \mathcal{S}$  and  $\alpha \in \mathit{Act}$

- $\mathcal{S}_0 \subseteq \mathcal{S}$  the set of **initial states**,
- $\mathit{AP}$  a set of **atomic propositions**,
- $L : \mathcal{S} \rightarrow 2^{\mathit{AP}}$  the **labeling function**



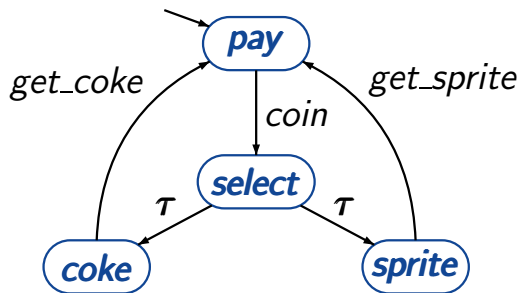


state space  $S = \{pay, select, coke, sprite\}$

set of initial states:  $S_0 = \{pay\}$

# Transition system for beverage machine

TS1.4-2



actions:

*coin*

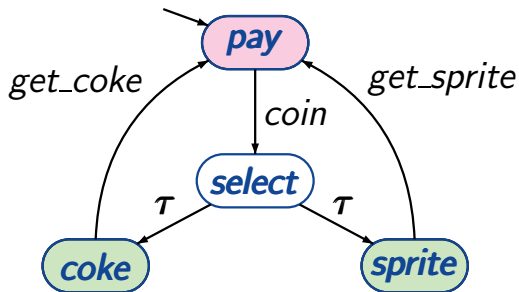
$\tau$

*get\_sprite*

*get\_coke*

state space  $S = \{pay, select, coke, sprite\}$

set of initial states:  $S_0 = \{pay\}$



actions:

*coin*

$\tau$

*get\_sprite*

*get\_coke*

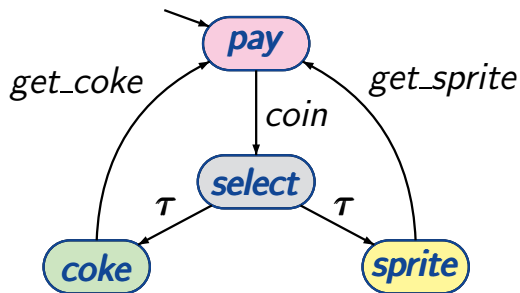
state space  $S = \{\textit{pay}, \textit{select}, \textit{coke}, \textit{sprite}\}$

set of initial states:  $S_0 = \{\textit{pay}\}$

set of atomic propositions:  $AP = \{\textit{pay}, \textit{drink}\}$

labeling function:  $L(\textit{coke}) = L(\textit{sprite}) = \{\textit{drink}\}$

$L(\textit{pay}) = \{\textit{pay}\}, L(\textit{select}) = \emptyset$



actions:

*coin*

$\tau$

*get\_sprite*

*get\_coke*

state space  $S = \{\textit{pay}, \textit{select}, \textit{coke}, \textit{sprite}\}$

set of initial states:  $S_0 = \{\textit{pay}\}$

set of atomic propositions:  $AP = S$

labeling function:  $L(s) = \{s\}$  for each state  $s$

possible behaviours of a TS result from:

```
select nondeterministically an initial state  $s \in S_0$ 
WHILE  $s$  is non-terminal DO
    select nondeterministically a transition  $s \xrightarrow{\alpha} s'$ 
    execute the action  $\alpha$  and put  $s := s'$ 
OD
```



possible behaviours of a TS result from:

```
select nondeterministically an initial state  $s \in S_0$ 
WHILE  $s$  is non-terminal DO
    select nondeterministically a transition  $s \xrightarrow{\alpha} s'$ 
    execute the action  $\alpha$  and put  $s := s'$ 
OD
```

*executions*: maximal “transition sequences”

$$s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \dots \text{ with } s_0 \in S_0$$

possible behaviours of a TS result from:

select **nondeterministically** an initial state  $s \in S_0$   
WHILE  $s$  is non-terminal DO  
    select **nondeterministically** a transition  $s \xrightarrow{\alpha} s'$   
    execute the **action**  $\alpha$  and put  $s := s'$   
OD

*executions*: maximal “transition sequences”

$$s_0 \xrightarrow{\alpha_1} s_1 \xrightarrow{\alpha_2} s_2 \xrightarrow{\alpha_3} \dots \text{ with } s_0 \in S_0$$

*reachable fragment*:

**Reach**( $\mathcal{T}$ ) = set of all states that are **reachable** from an initial state through some execution

# Possible meanings of nondeterminism in TS

TS1.4-3A

- (true) concurrency modeled by interleaving
- competition of parallel dependent actions
- implementational freedom, underspecification
- incomplete information on system environment

parallel execution of independent actions

parallel execution of dependent actions

parallel execution of independent actions

e.g.  $\underbrace{x := x+1}_{\text{action } \alpha} \parallel \underbrace{y := y-3}_{\text{action } \beta}$      $\alpha, \beta$  independent

parallel execution of dependent actions

parallel execution of independent actions

e.g.  $\underbrace{x := x+1}_{\text{action } \alpha} \parallel \parallel \underbrace{y := y-3}_{\text{action } \beta}$      $\alpha, \beta$  independent

parallel execution of dependent actions

e.g.  $\underbrace{x := x+1}_{\text{action } \alpha} \parallel \parallel \underbrace{y := 2*x}_{\text{action } \beta}$      $\alpha, \beta$  dependent

# Transition system for parallel actions

TS1.4-4

parallel execution of independent actions ← interleaving

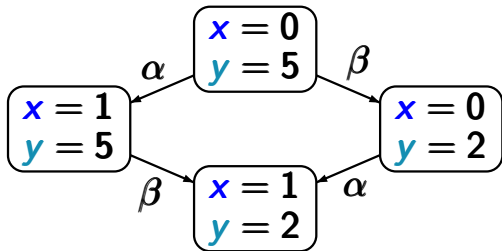
e.g.  $\underbrace{x := x+1}_{\text{action } \alpha} \parallel \underbrace{y := y-3}_{\text{action } \beta}$   $\alpha, \beta$  independent

parallel execution of dependent actions ← competition

e.g.  $\underbrace{x := x+1}_{\text{action } \alpha} \parallel \underbrace{y := 2*x}_{\text{action } \beta}$   $\alpha, \beta$  dependent

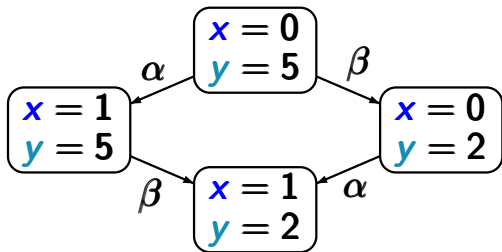


parallel execution of independent actions ← interleaving



$x := x + 1$  |||  $y := y - 3$   
action  $\alpha$       action  $\beta$

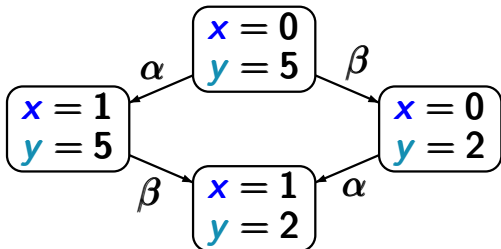
parallel execution of independent actions ← interleaving



$x := x + 1$  |||  $y := y - 3$   
action  $\alpha$       action  $\beta$

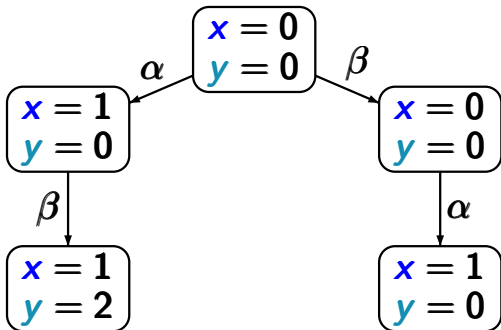
parallel execution of dependent actions ← competition

parallel execution of independent actions ← interleaving



$x := x + 1$  |||  $y := y - 3$   
action  $\alpha$       action  $\beta$

parallel execution of dependent actions ← competition



$x := x + 1$  |||  $y := 2 * x$   
action  $\alpha$       action  $\beta$

- (true) concurrency modeled by interleaving
- competition of parallel dependent actions
- implementational freedom, underspecification
- incomplete information on system environment

... modelled by nondeterminism

# Implementation freedom

TS1.4-5

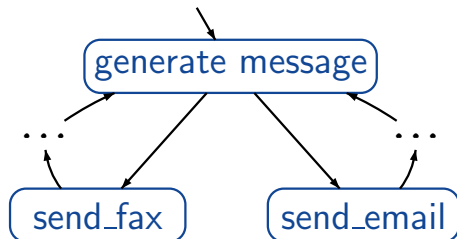


# Implementation freedom

TS1.4-5

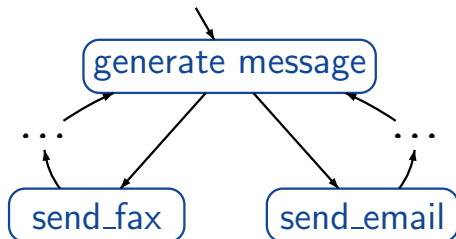


realization by a TS:



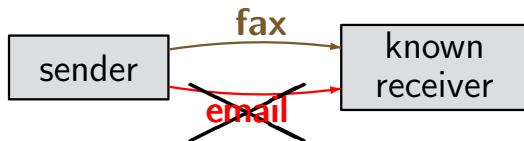


realization by a TS:



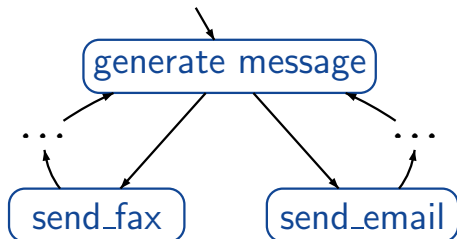
at a future refinement step the **nondeterminism** is replaced with **one** of the alternatives





without  
email access

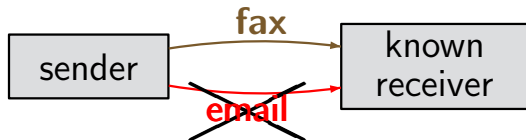
realization by a TS:



at a future refinement step the **nondeterminism**  
is replaced with **one** of the alternatives

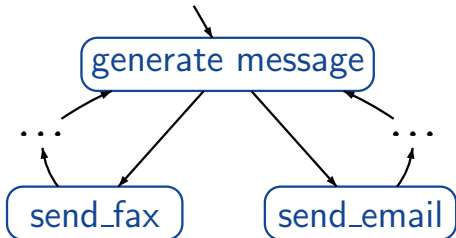
# Implementation freedom

TS1.4-5

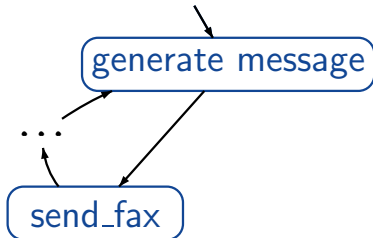


without  
email access

realization by a TS:

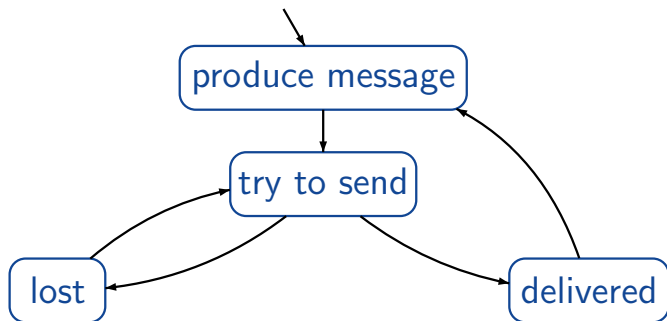


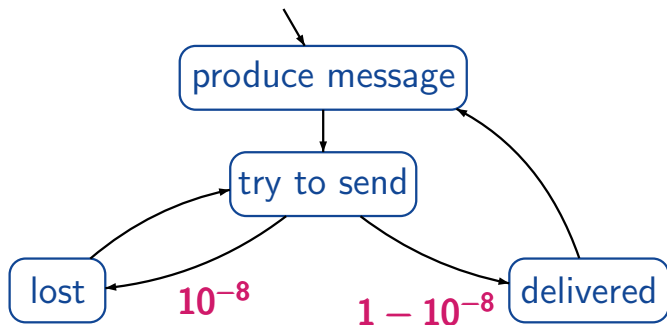
refined TS:



at a future refinement step the **nondeterminism**  
is replaced with **one** of the alternatives







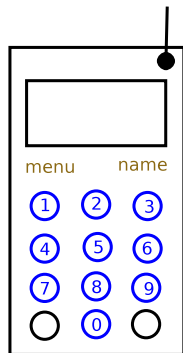
at a future refinement step the **nondeterminism** is replaced with **probabilism**

- (true) concurrency modeled by interleaving
- competition of parallel dependent actions
- implementational freedom, underspecification
- incomplete information on system environment

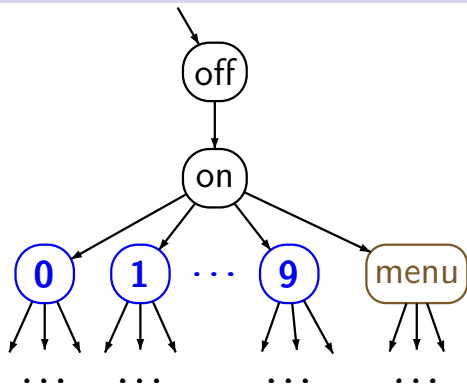
- (true) concurrency modeled by interleaving
- competition of parallel dependent actions
- implementational freedom, underspecification
- incomplete information on system environment, e.g., interfaces with other programs, human users, sensors

# Incomplete information on the environment

TS1.4-7



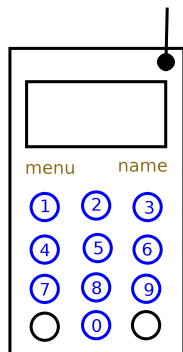
mobile phone



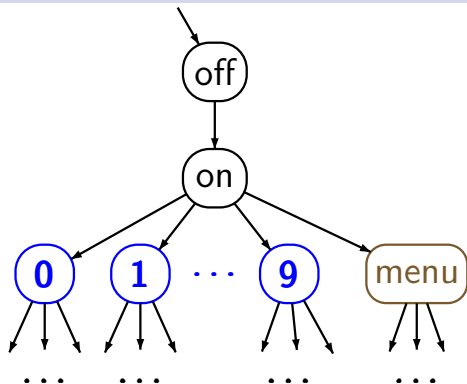


# Incomplete information on the environment

TS1.4-7



mobile phone



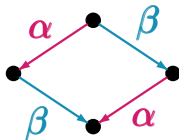
resolution of the **nondeterministic choices**  
by a **human user**

# Possible meanings of nondeterminism in TS

TS1.4-8

*concurrency (interleaving)*

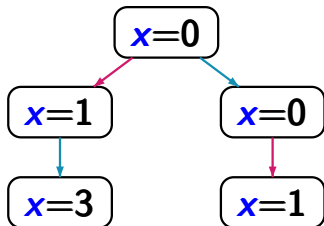
$\alpha \parallel \beta$  is represented by



*competitions*

to be resolved by a scheduler

e.g.  $x := x + 1 \parallel x := 3x$



*underspecification, implementational freedom*

*incomplete information* on system environment, e.g.,  
interfaces with other programs, human users, sensors