Model Checking I
alias
Reactive Systems Verification

Luca Tesei

MSc in Computer Science, University of Camerino

Topics

- Transition Systems

Material

Reading:

Chapter 2 of the book, pages 19–26.

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.

Overview

Introduction

Modelling parallel systems

- Transition systems ⟵
- Modeling hard- and software systems
- Parallelism and communication

Linear Time Properties

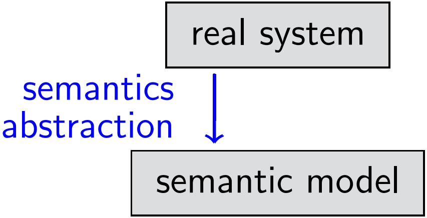
Regular Properties

Linear Temporal Logic

Computation-Tree Logic

Equivalences and Abstraction

Transition systems

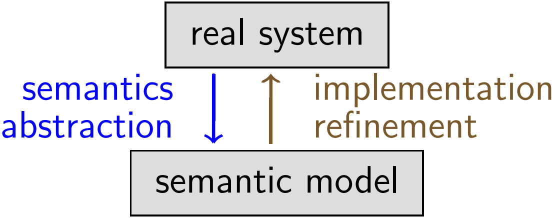


real system

semantics
abstraction

semantic model

Transition systems



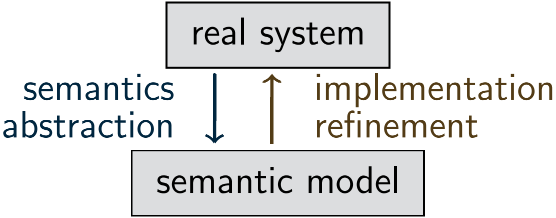
real system

semantics
abstraction

implementation
refinement

semantic model

Transition systems



real system

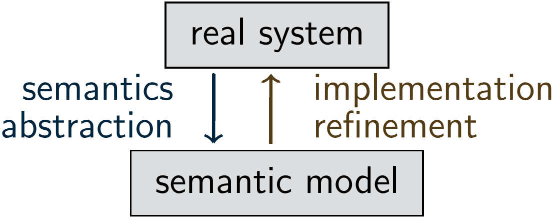
semantics
abstraction

implementation
refinement

semantic model

The semantic model yields a formal representation of:

Transition systems



real system

semantics
abstraction

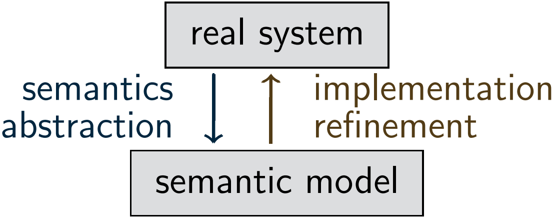
implementation
refinement

semantic model

The semantic model yields a formal representation of:

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

Transition systems



real system

semantics
abstraction

implementation
refinement

semantic model

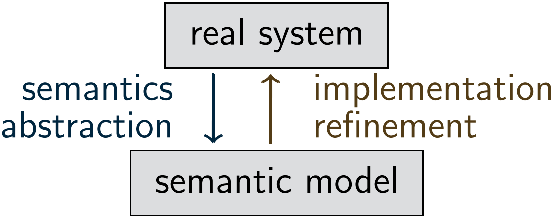
The semantic model yields a formal representation of:

- the states of the system

control component + information on "relevant" data

- the stepwise behaviour
- the initial states

Transition systems $\widehat{=}$ extended digraphs



real system

semantics
abstraction

implementation
refinement

semantic model

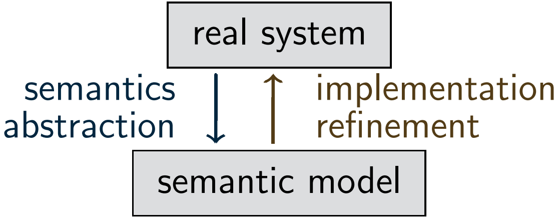
The semantic model yields a formal representation of:

- the states of the system ⟵ **nodes**

control component + information on "relevant" data

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

Transition systems $\widehat{=}$ extended digraphs



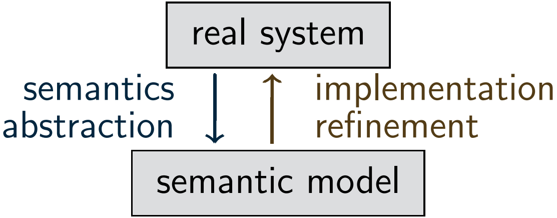
The semantic model yields a formal representation of:

- the states of the system ⟵ **nodes**

control component + information on "relevant" data

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

Transition systems $\widehat{=}$ extended digraphs



real system

semantics
abstraction

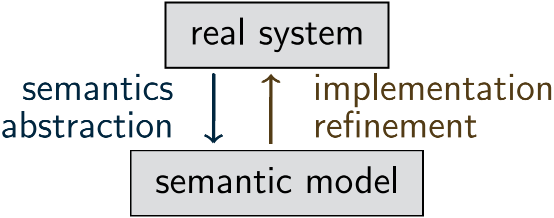
implementation
refinement

semantic model

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

Transition systems $\widehat{=}$ extended digraphs



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)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

Transition system (TS)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

- S is the state space, i.e., set of states,

Transition system (TS)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

- S is the state space, i.e., set of states,
- Act is a set of actions,

Transition system (TS)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

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

Transition system (TS)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

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

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

Transition system (TS)

A transition system is a tuple

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

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

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

- $S_0 \subseteq S$ the set of initial states,

Transition system (TS)

A transition system is a tuple

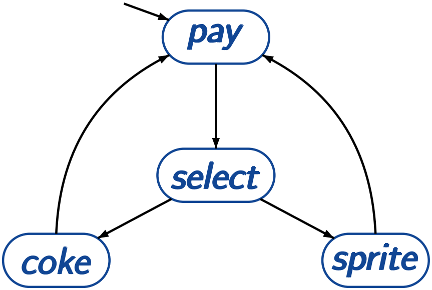
$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

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

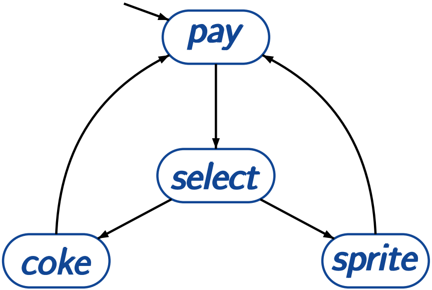
i.e., transitions have the form $s \xrightarrow{\alpha} s'$
where $s, s' \in S$ and $\alpha \in Act$

- $S_0 \subseteq S$ the set of initial states,
- AP a set of atomic propositions,
- $L : S \to 2^{AP}$ the labeling function

Transition system for beverage machine



Transition system for beverage machine

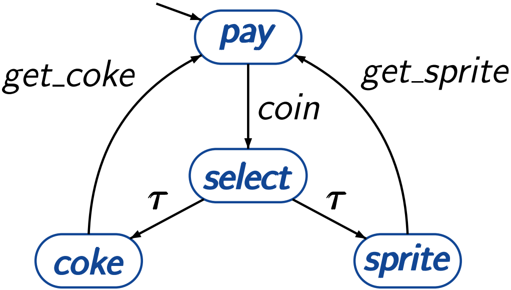


state space $S = \{pay, select, coke, sprite\}$

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

Transition system for beverage machine

TS1.4-2

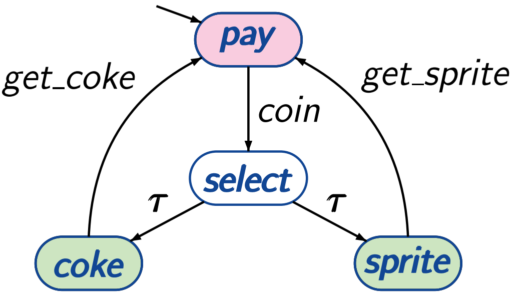


actions:
coin
τ
get_sprite
get_coke

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

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

Transition system for beverage machine



actions:
- *coin*
- τ
- *get_sprite*
- *get_coke*

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

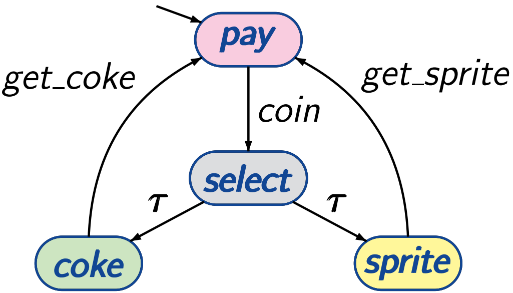
set of initial states: $S_0 = \{pay\}$

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

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

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

Transition system for beverage machine



actions:
coin
τ
get_sprite
get_coke

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

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

set of atomic propositions: $AP = S$

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

"Behaviour" of transition systems

TS1.4-3

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
```

"Behaviour" of transition systems

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"

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

“Behaviour” of transition systems

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”

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

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

Possible meanings of nondeterminism in TS

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

Transition system for parallel actions

parallel execution of independent actions

parallel execution of dependent actions

Transition system for parallel actions

parallel execution of independent actions

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

parallel execution of dependent actions

Transition system for parallel actions

parallel execution of independent actions

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

parallel execution of dependent actions

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

Transition system for parallel actions

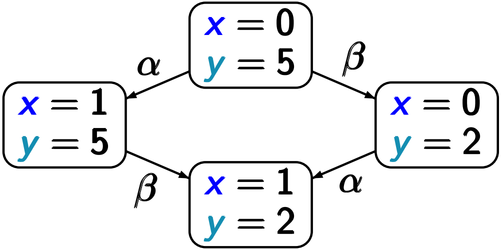
parallel execution of independent actions ← $\boxed{\text{interleaving}}$

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

parallel execution of dependent actions ← $\boxed{\text{competition}}$

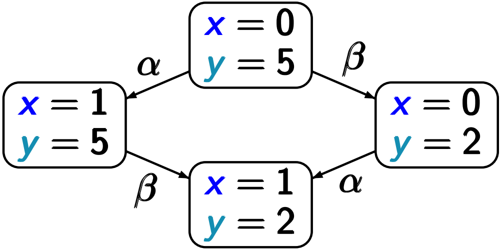
e.g. $\underbrace{x := x+1}_{\text{action } \alpha} \; ||| \; \underbrace{y := 2*x}_{\text{action } \beta}$ α, β dependent

parallel execution of independent actions ← interleaving



$$\underbrace{x := x+1}_{\text{action } \alpha} \;|||\; \underbrace{y := y-3}_{\text{action } \beta}$$

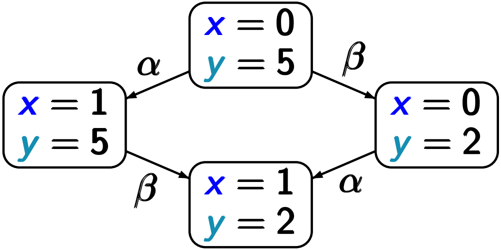
parallel execution of independent actions ← interleaving



$$\underbrace{x := x+1}_{\text{action } \alpha} \;|||\; \underbrace{y := y-3}_{\text{action } \beta}$$

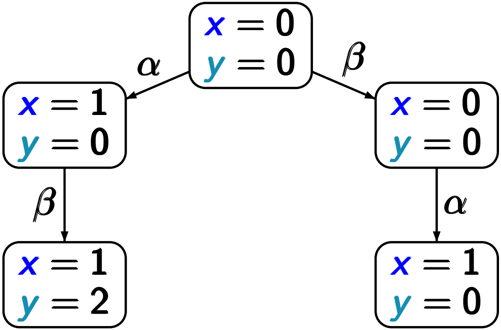
parallel execution of dependent actions ← competition

parallel execution of independent actions ← interleaving



$$\underbrace{x := x+1}_{\text{action } \alpha} \;|||\; \underbrace{y := y-3}_{\text{action } \beta}$$

parallel execution of dependent actions ← competition



$$\underbrace{x := x+1}_{\text{action } \alpha} \;|||\; \underbrace{y := 2*x}_{\text{action } \beta}$$

Possible meanings of nondeterminism in TS

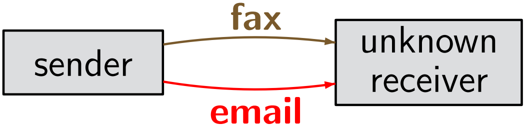
- (true) concurrency modeled by interleaving
- competition of parallel dependent actions
- implementational freedom, underspecification
- incomplete information on system environment

Implementation freedom

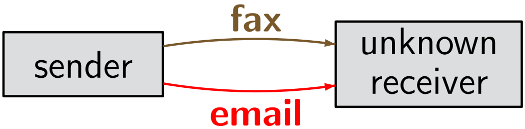
... modelled by nondeterminism

Implementation freedom

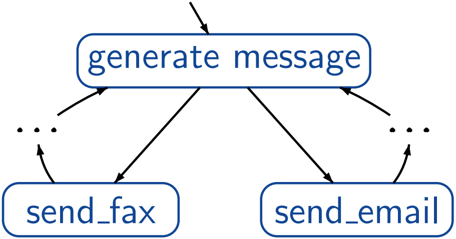
TS1.4-5



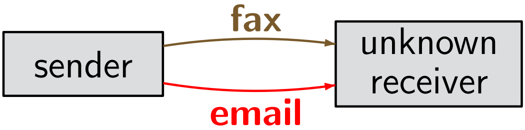
Implementation freedom



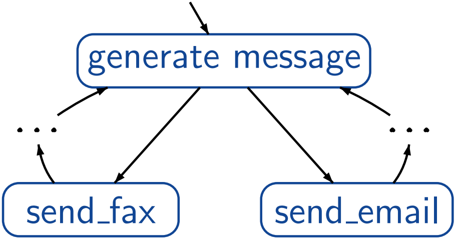
realization by a TS:



Implementation freedom

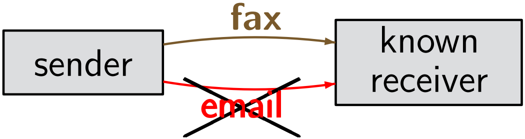


realization by a TS:



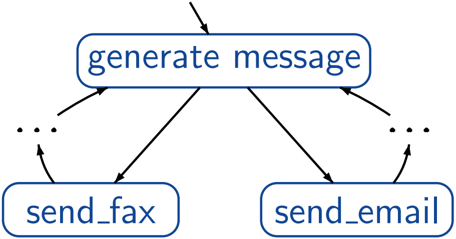
at a future refinement step the nondeterminism is replaced with one of the alternatives

Implementation freedom



without
email access

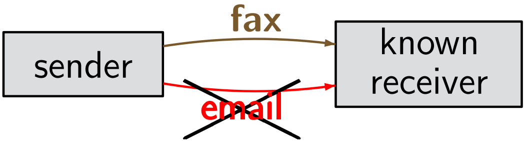
realization by a TS:



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

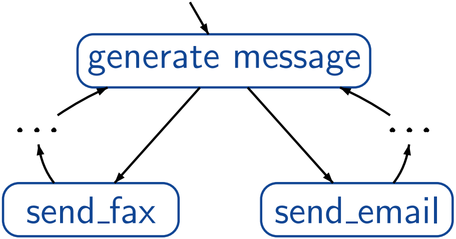
Implementation freedom

TS1.4-5

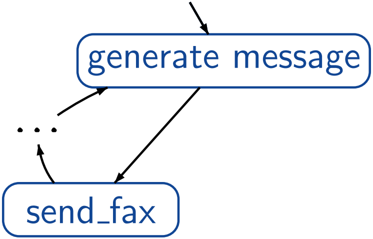


without email access

realization by a TS:



refined TS:



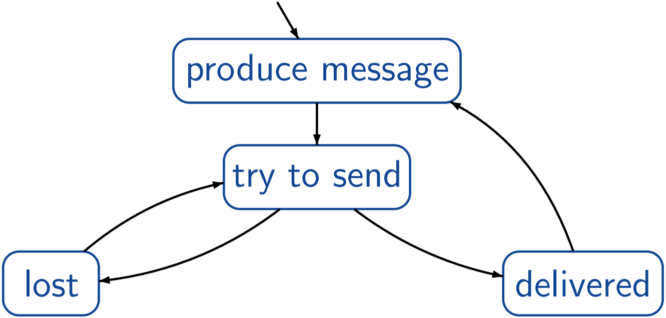
at a future refinement step the nondeterminism is replaced with one of the alternatives

Underspecification

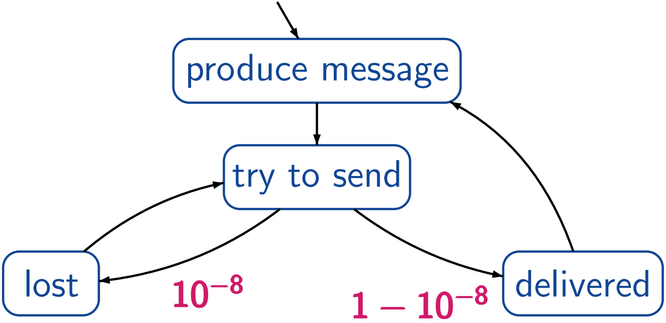
TS1.4-6

Underspecification

TS1.4-6



Underspecification



at a future refinement step the nondeterminism is replaced with probabilism

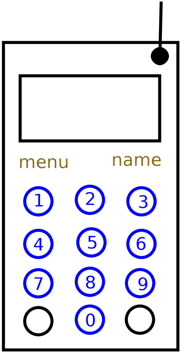
Possible meanings of nondeterminism in TS

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

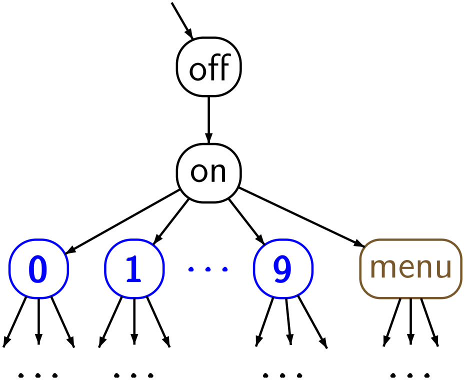
Possible meanings of nondeterminism in TS

- (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

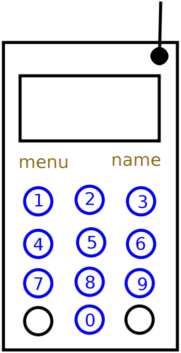
Incomplete information on the environment



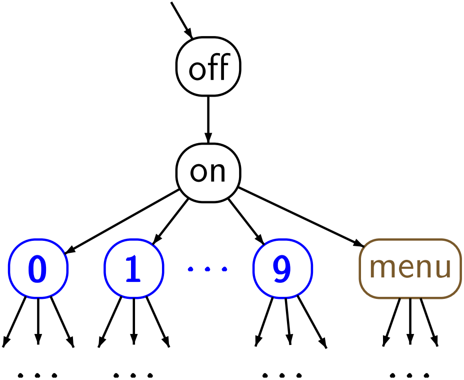
mobile phone



Incomplete information on the environment



mobile phone

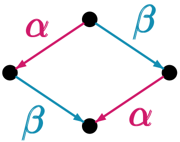


resolution of the nondeterministic choices
by a human user

Possible meanings of nondeterminism in TS

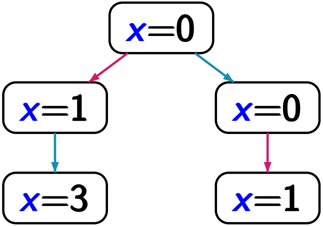
concurrency (interleaving)

$\alpha \,|||\, \beta$ 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