A Theory of Model-Based Testing with Labelled Transition Systems

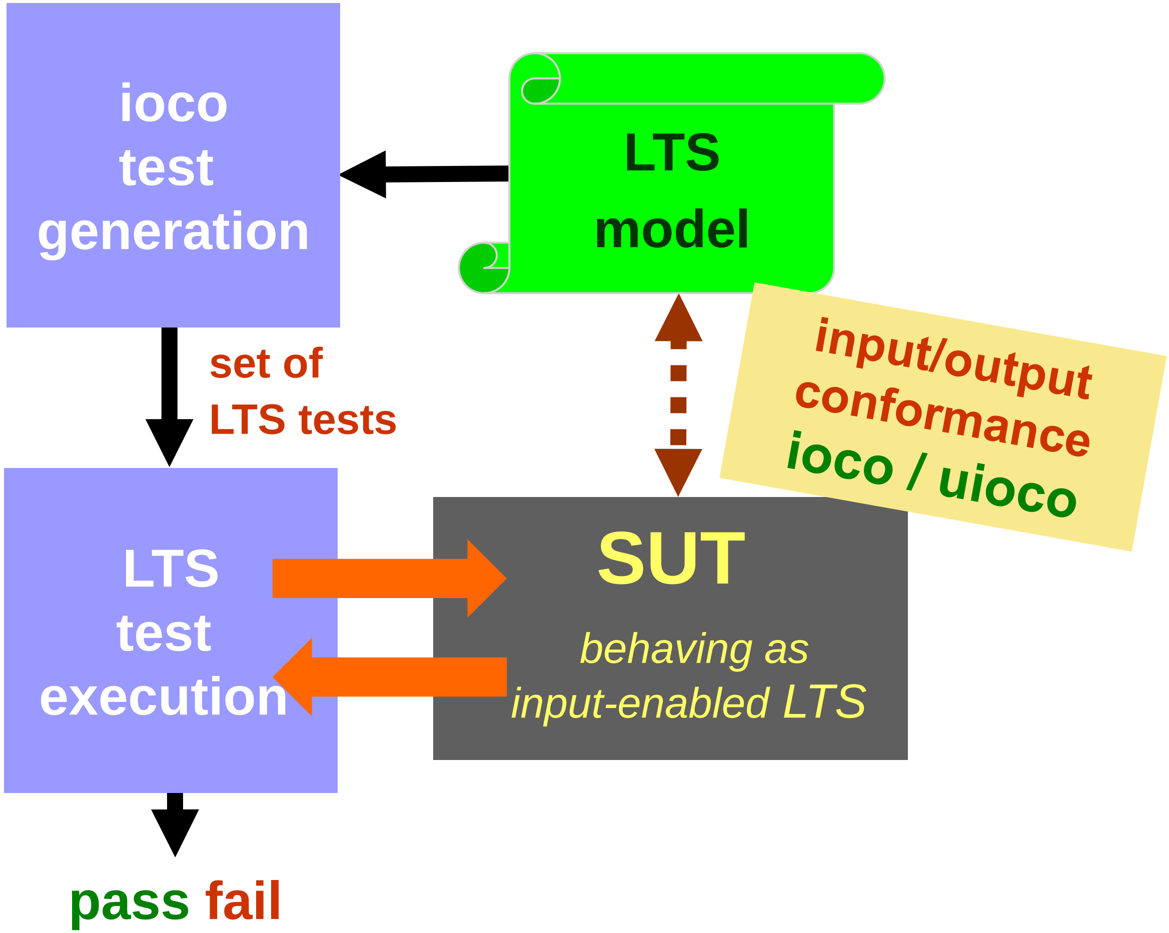
*The **uioco** Theory*

MBT : Labelled Transitions Systems

SUT uioco model

sound ⇓ ⇑ *exhaustive*

SUT passes tests



ioco test generation

LTS model

set of LTS tests

input/output conformance ioco / uioco

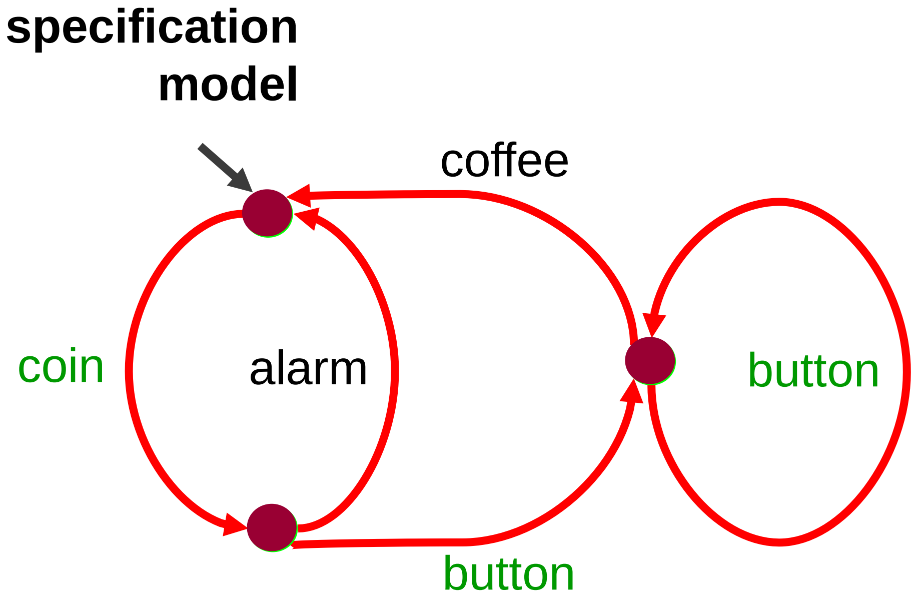
LTS test execution

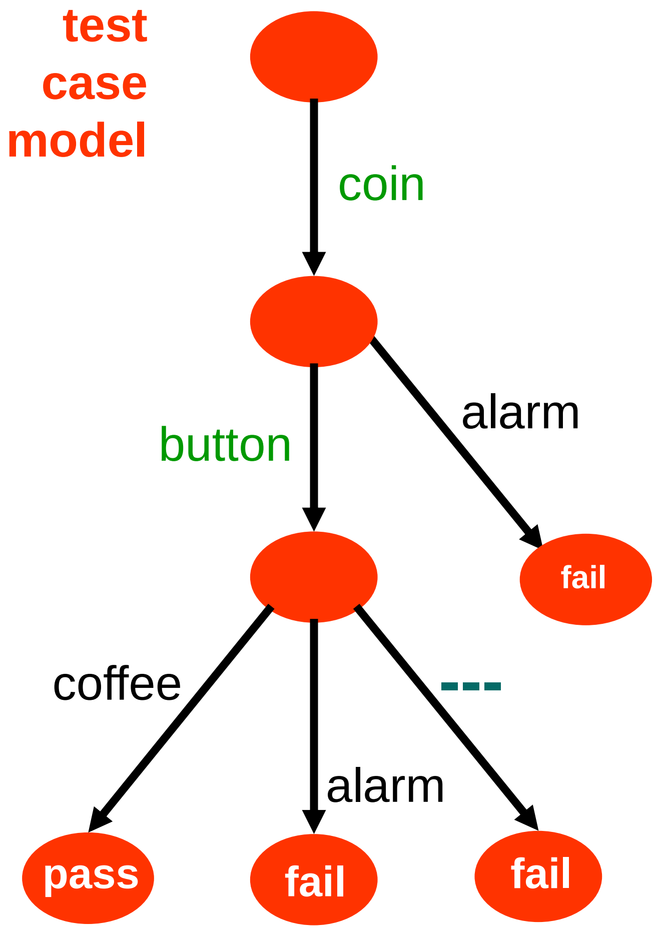
SUT

behaving as input-enabled LTS

pass fail

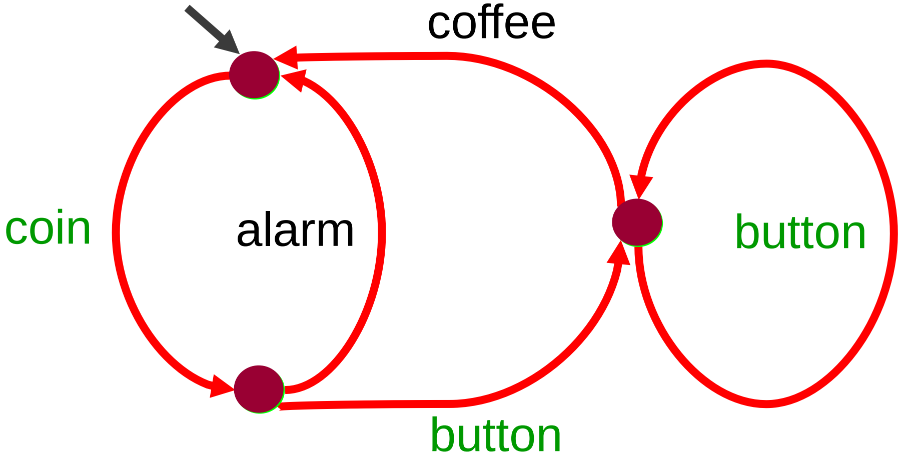
Models: Generation of Test Cases



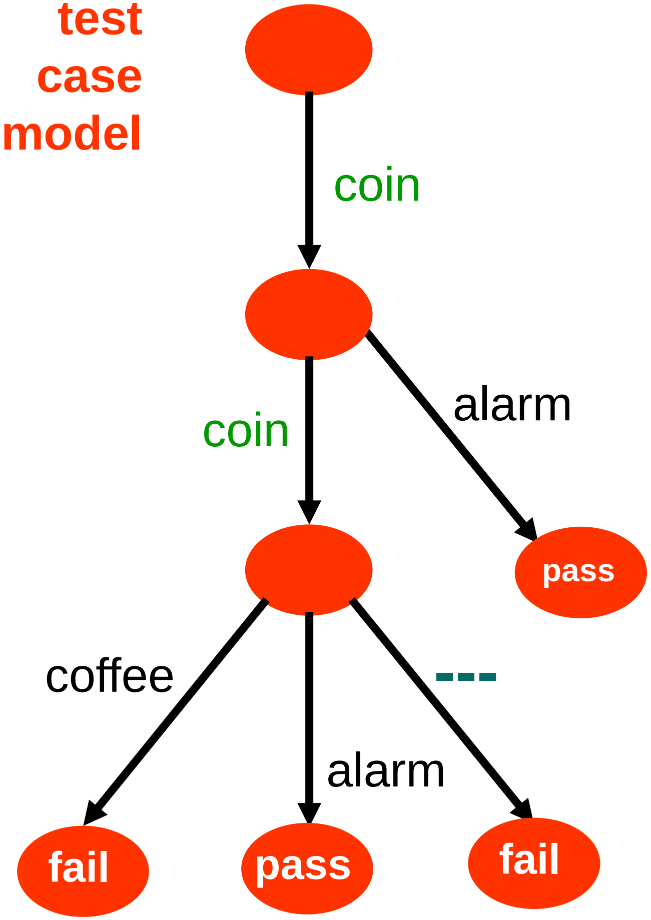


Models: Generation of Test Cases

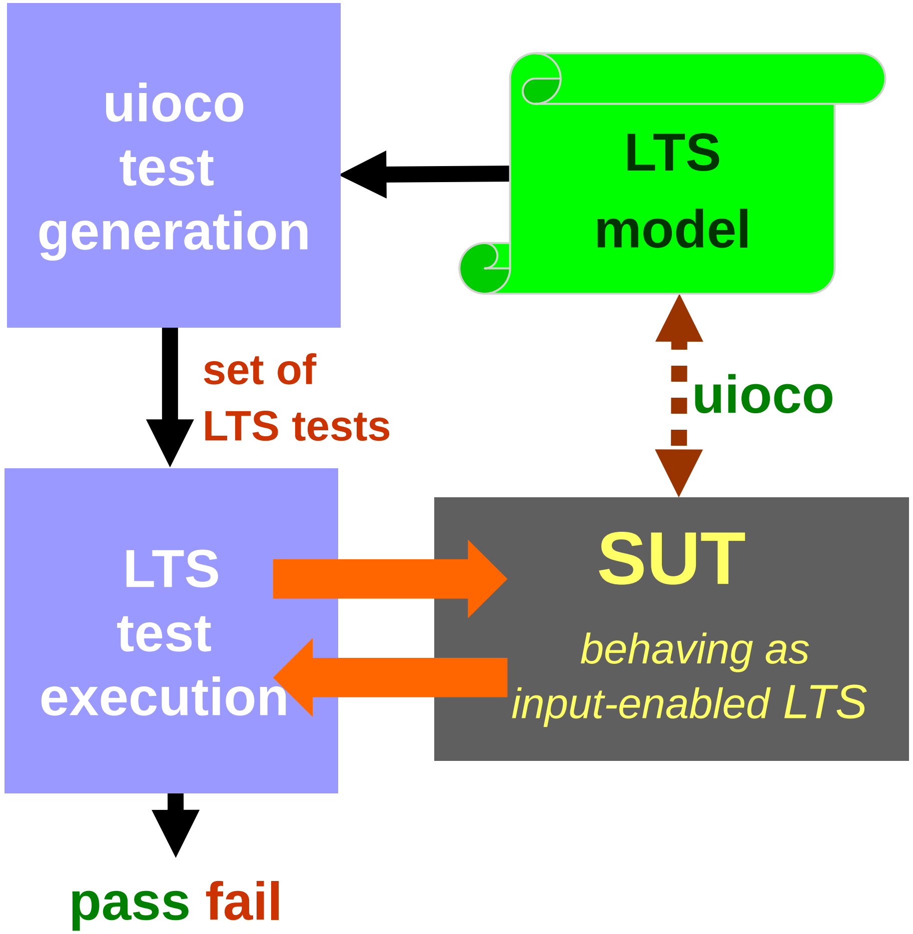
specification model



test case model



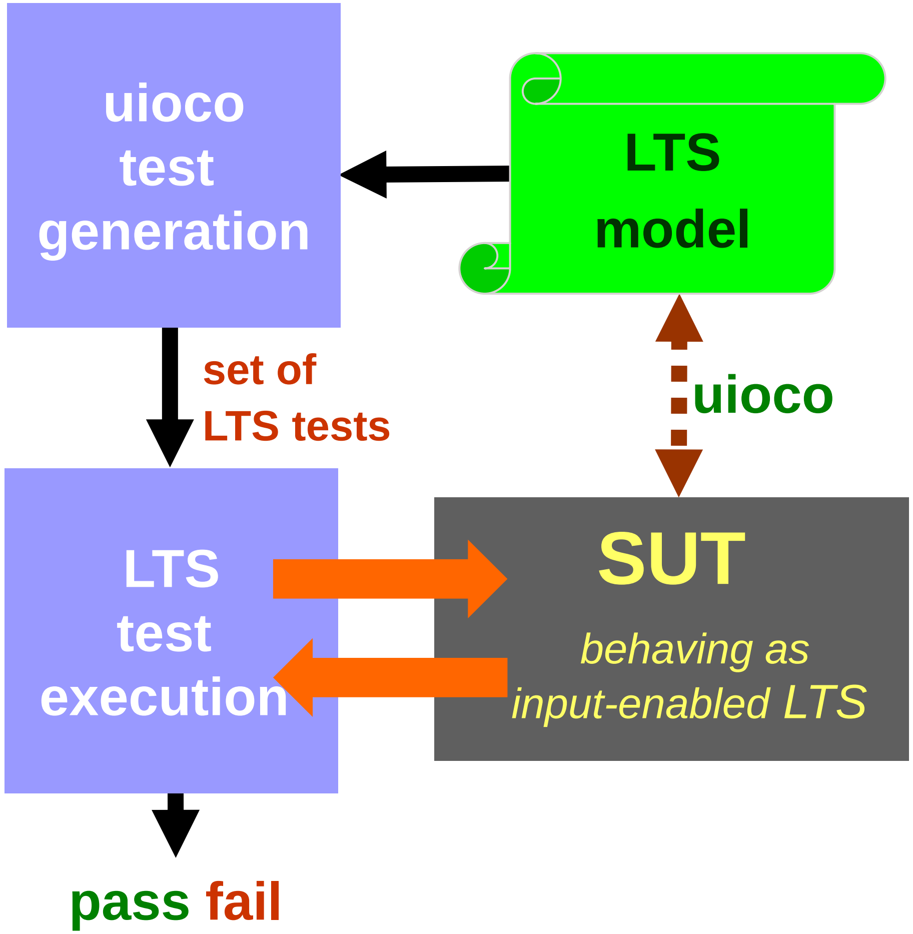
MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ specification model
- ☞ system under test (SUT)
- ☞ SUT model
- ☞ conformance **uioco**
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive

MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ **specification model**
- ☞ system under test (SUT)
- ☞ SUT model
- ☞ conformance **uioco**
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive

Models: Labelled Transition Systems

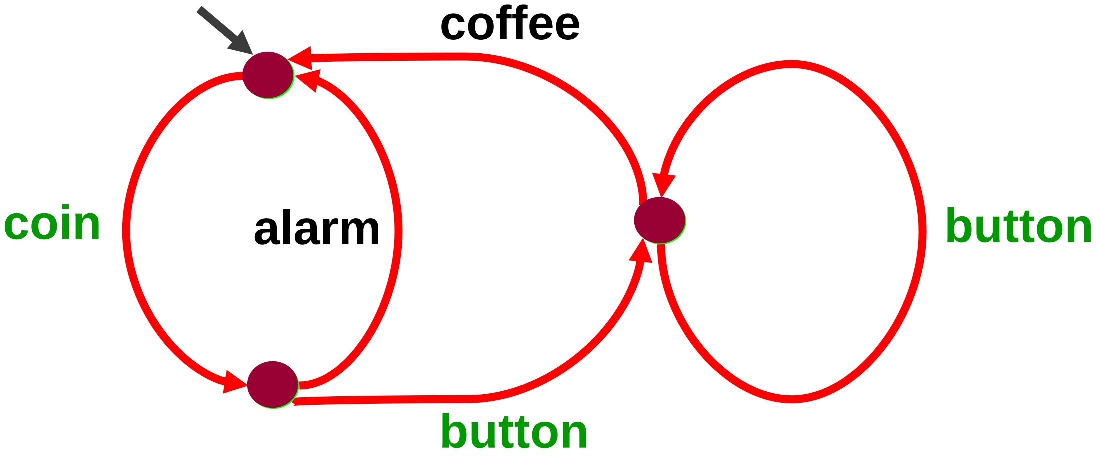
Labelled Transition System: $\langle$ **S, L, T, s_0** $\rangle$

set of states

set of labels

transitions
$T \subseteq S \times (L \cup \{\tau\}) \times S$

initial state
$s_0 \in S$



Models: LTS with Inputs and Outputs

coin, button

from user to machine
initiative with user

? inputs L_I

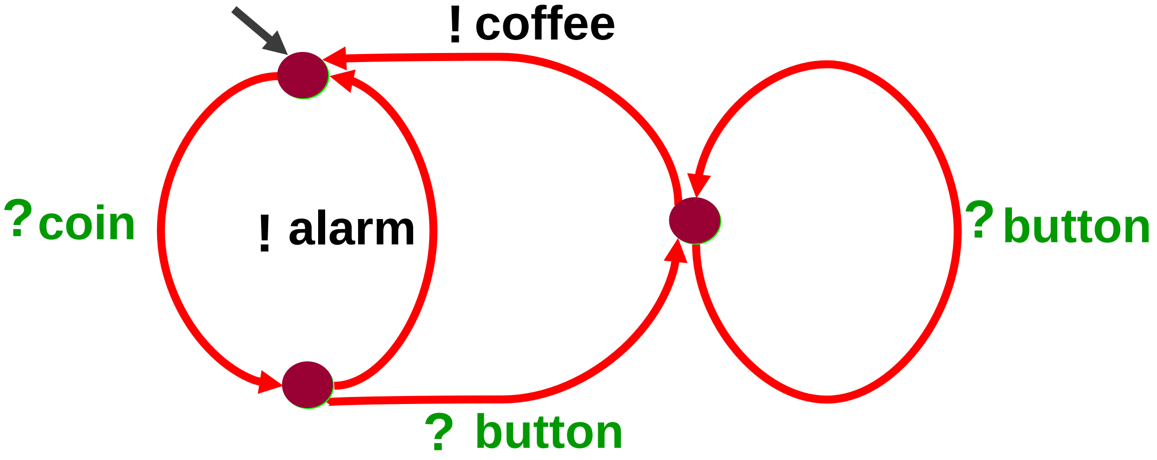
coffee, alarm

from machine to user
initiative with machine

! outputs L_U

$$L_I \cap L_U = \emptyset$$

$$L_I \cup L_U = L$$



Models: Labelled Transition Systems

Labelled Transition System: $\langle$ **S**, **L_I**, **L_U**, **T**, **s_0** $\rangle$

states

input labels

output labels

transitions

$T \subseteq S \times (L \cup \{\tau\}) \times S$

initial state

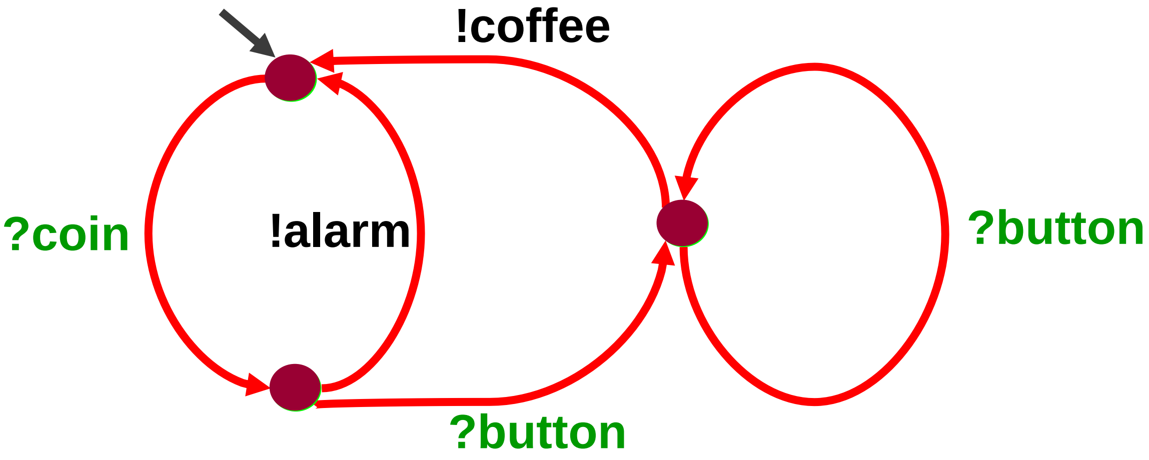
$s_0 \in S$

? = *input*

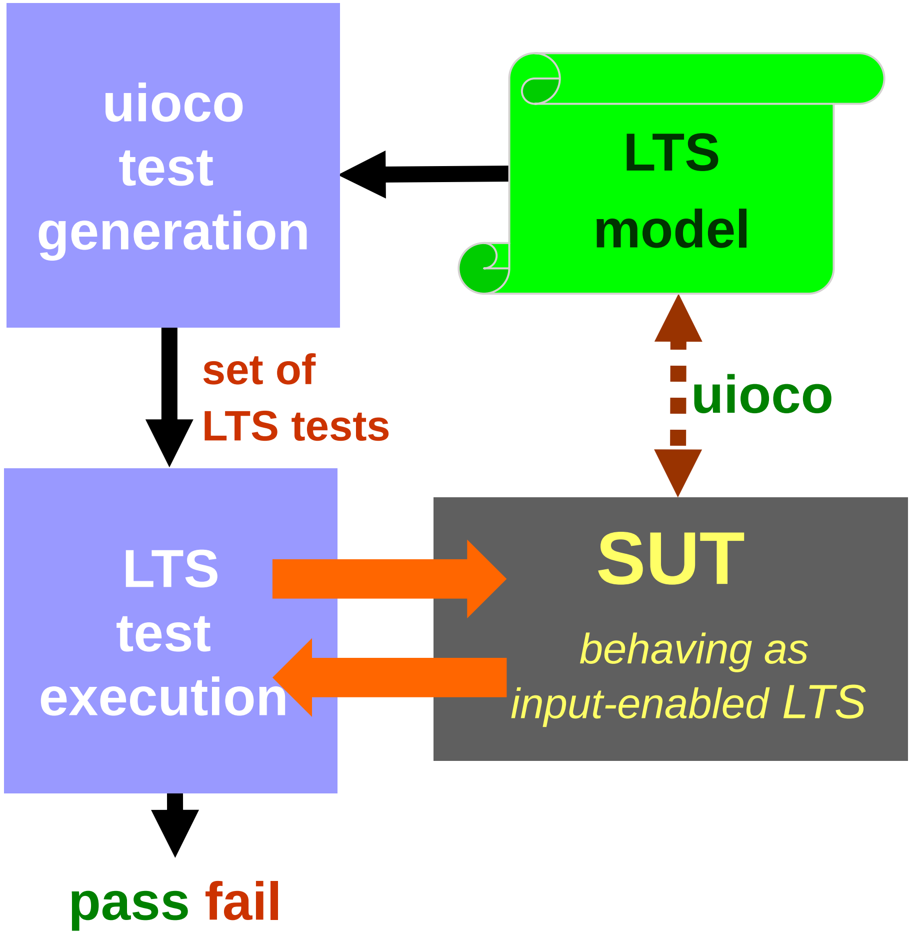
! = *output*

$L = L_I \cup L_U$

$L_I \cap L_U = \emptyset$



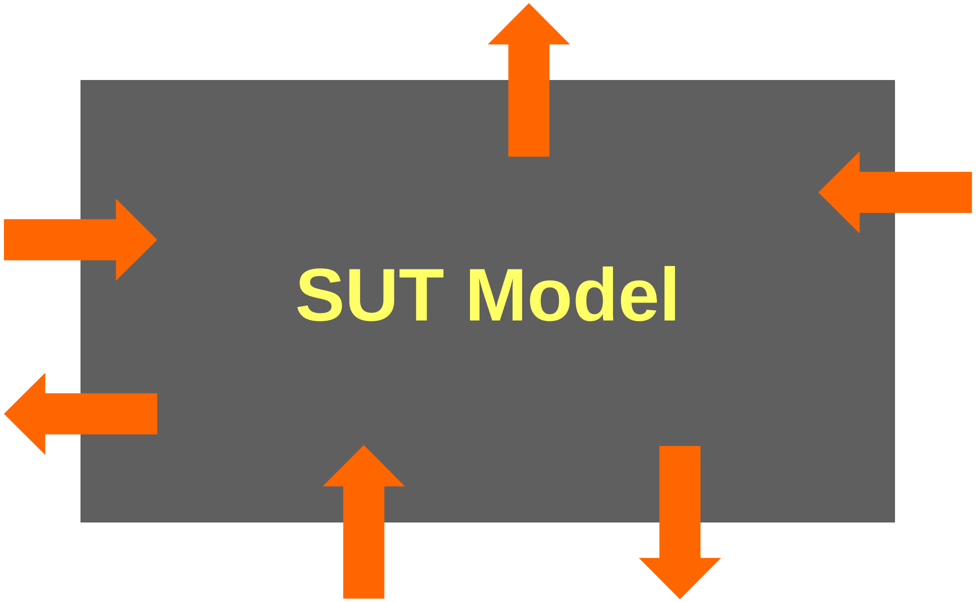
MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ specification model
- ☞ **system under test (SUT)**
- ☞ SUT model
- ☞ conformance **uioco**
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive

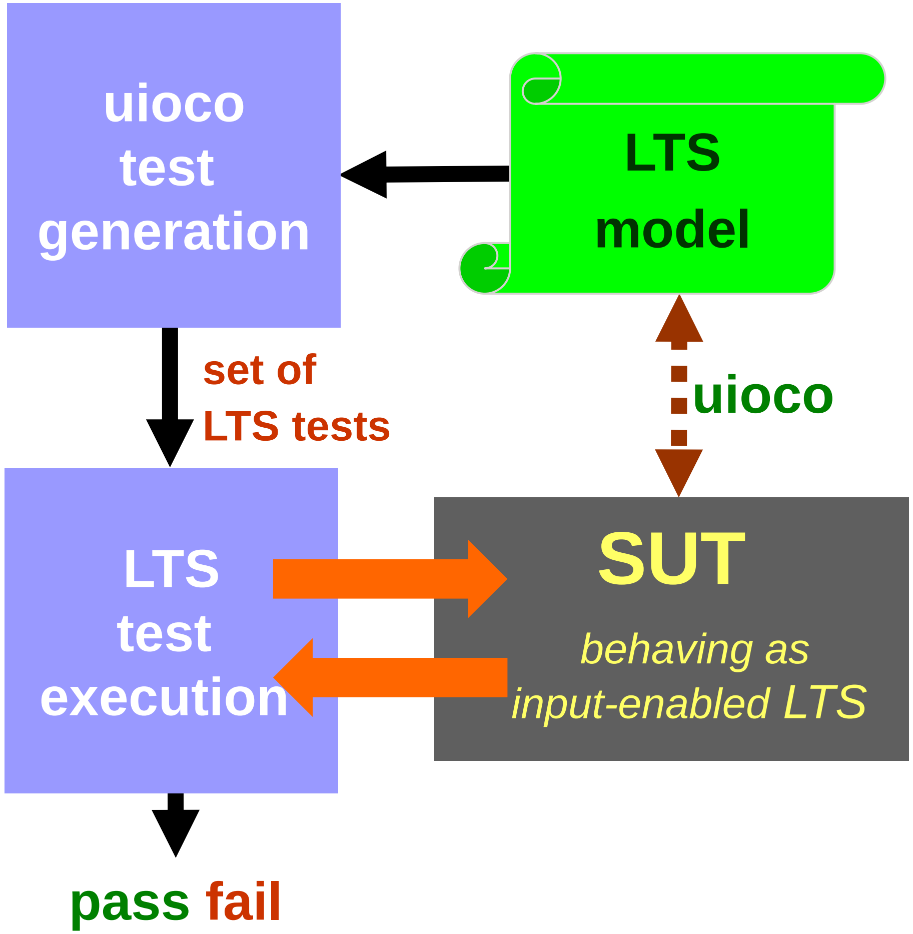
View on SUT



System Under Test

- *discrete, event-driven, reactive, dynamic, data-intensive*
- *black box*
- *inputs on interfaces*
- *outputs on interfaces*

MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ specification model
- ☞ system under test (SUT)
- ☞ **SUT model**
- ☞ conformance **uioco**
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive

Models: Input-Output Transition Systems

In many systems, inputs are always enabled:

input-enabled *transition systems*

= *Input-Output Transition Systems*

$\textit{IOTS}(L_I, L_U) \subseteq \textit{LTS}(L_I \cup L_U)$

input enabled:

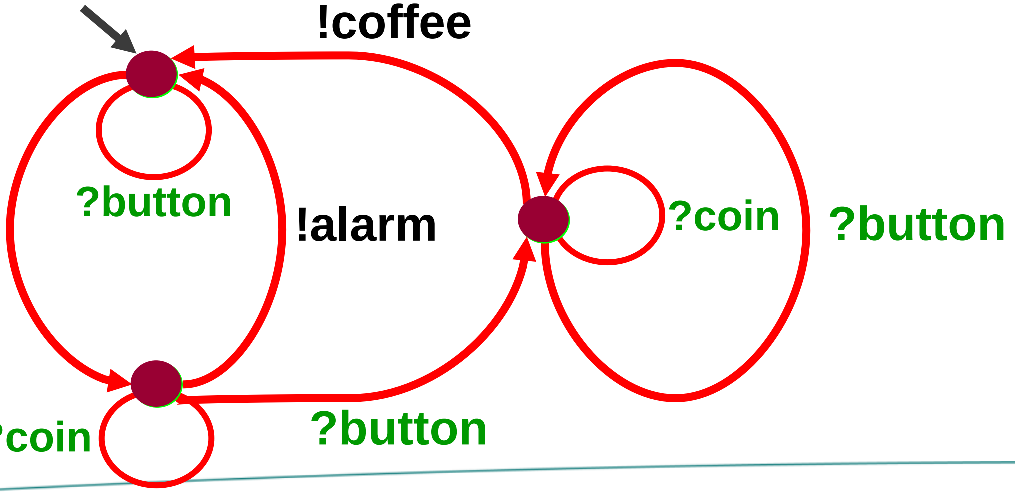
for all states s,

for all inputs $?a \in L_I$:

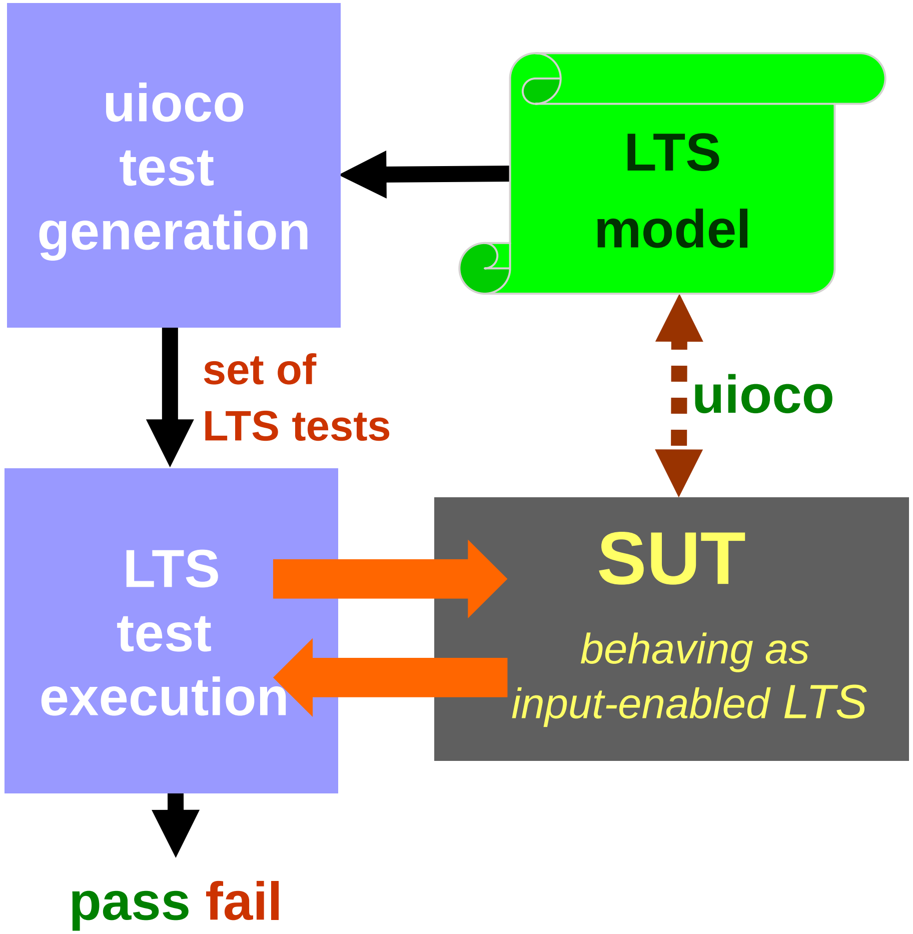
$$s \xRightarrow{?a}$$

Testability Assumption

Model of SUT $\in$ $\textit{IOTS}(L_I, L_U)$



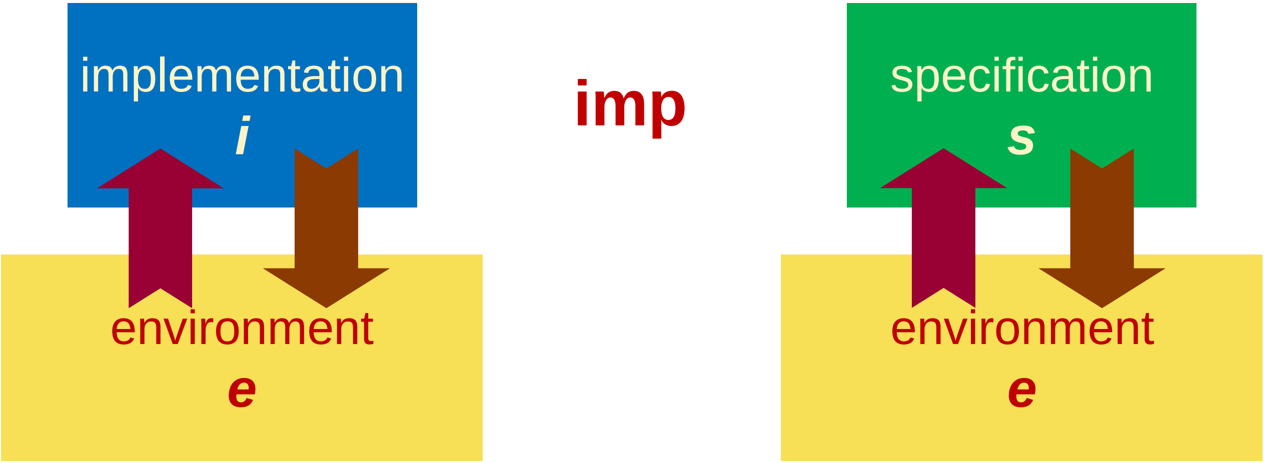
MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ specification model
- ☞ system under test (SUT)
- ☞ SUT model
- ☞ **conformance *uioco***
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive

Implementation Relations for Input-Output Transition Systems



$i \in IOTS(L_I,L_U)$ $s \in LTS(L_I,L_U)$

$\mathbf{imp} \subseteq IOTS(L_I,L_U) \times LTS(L_I,L_U)$

i **imp** ***s***

i *is a conforming implementation of* **s**

Input/Output Conformance : *uioco*

$\mathbf{i}\ \ \mathbf{uioco}\ \ \mathbf{s}\quad =_{\mathrm{def}}\ \ \forall \sigma \in \mathit{Utraces}\,(\mathbf{s}) : \ \mathit{out}\,(\mathbf{i}\ \mathbf{after}\ \sigma) \subseteq \mathit{out}\,(\mathbf{s}\ \mathbf{after}\ \sigma)$

s is a Labelled Transition System $\quad \mathbf{s} \in \mathit{LTS}(L_I, L_U)$

i is (assumed to be) an input-enabled LTS $\quad \mathbf{i} \in \mathit{IOTS}(L_I, L_U)$

$\mathbf{uioco} \subseteq \mathit{IOTS}\,(L_I, L_U)\ \mathbf{x}\ \mathit{LTS}\,(L_I, L_U)$

Alternative: **ioco**:

(*see Lecture Notes*)

$\mathbf{i}\ \ \mathbf{ioco}\ \ \mathbf{s}\quad =_{\mathrm{def}}\ \ \forall \sigma \in \mathit{Straces}\,(\mathbf{s}) : \ \mathit{out}\,(\mathbf{i}\ \mathbf{after}\ \sigma) \subseteq \mathit{out}\,(\mathbf{s}\ \mathbf{after}\ \sigma)$

Input/Output Conformance : *uioco*

$$i \;\; \text{uioco} \;\; s \quad =_{def} \;\; \forall \sigma \in \textit{Utraces}\,(s) : \;\; \textit{out}\,(i \text{ after } \sigma) \;\subseteq\; \textit{out}\,(s \text{ after } \sigma)$$

s is a Labelled Transition System

i is (assumed to be) an input-enabled LTS

$$p \xrightarrow{\delta} p \quad \Leftrightarrow \quad \forall \;\; !x \;\in\; L_U \cup \{\tau\} \,.\;\; p \overset{!x}{\not\longrightarrow} \quad \Leftrightarrow \quad p \text{ refuses } L_U$$

$$\textit{Straces}\,(s) \quad = \quad \{\; \sigma \;\in (\, L \cup \{\delta\}\,)^* \mid s \overset{\sigma}{\Longrightarrow} \;\}$$

$$\textit{Utraces}\,(s) \quad = \quad \{\, \sigma \in \textit{Straces}\,(s) \;\; |$$

$$\forall\; \sigma_1 \,?b\;\, \sigma_2 = \sigma : \;\; \textit{not}\,(\, s \;\text{ after } \sigma_1 \text{ refuses } \{?b\}\,) \;\; \}$$

$$\textit{out}\,(\,P\,) \quad = \{\;\; !x \in L_U \mid \exists\, p{\in}P : p \xrightarrow{!x} \} \;\cup\; \{\, \delta \mid \exists\, p{\in}P : p \xrightarrow{\delta} p \,\}$$

Input/Output Conformance : *uioco*

i uioco s $=_{def}$ $\forall\, \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after σ)** $\subseteq$ ***out*** **(s after σ)**

s is a Labelled Transition System

i is (assumed to be) an input-enabled LTS

Intuition:

i **uioco**-conforms to **s**, iff

- if **i** produces output **x** after *U*-trace σ,

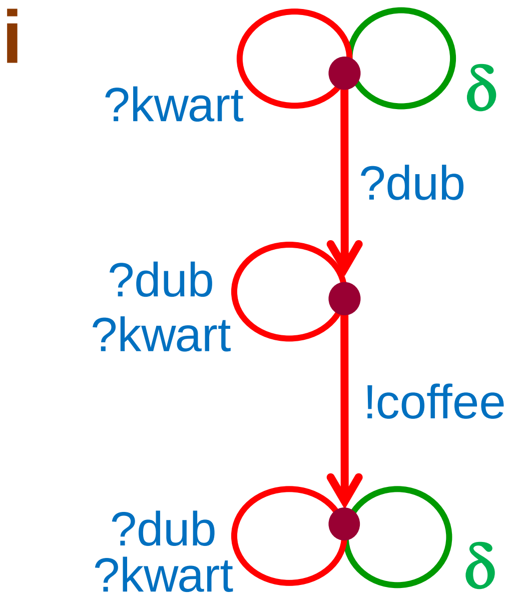
  then **s** can produce **x** after σ

- if **i** cannot produce any output after *U*-trace σ,

  then **s** cannot produce any output after σ (called *quiescence* δ)

Implementation Relation uioco

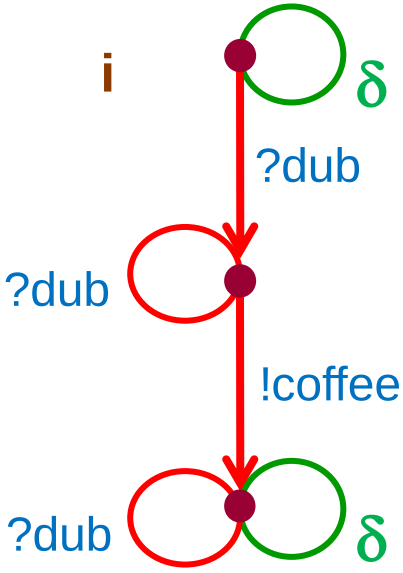
$$i \ \mathbf{uioco} \ s \ =_{def} \ \forall \sigma \in \mathit{Utraces}(s) : \ \mathit{out}(i \text{ after } \sigma) \subseteq \mathit{out}(s \text{ after } \sigma)$$



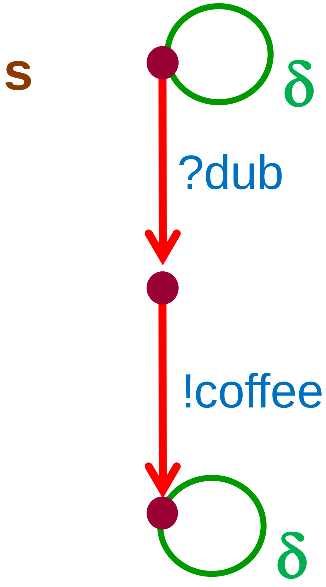
out (i after ε)	=	{ δ }
out (i after ?dub)	=	{ !coffee }
out (i after ?dub ?dub)	=	{ !coffee }
out (i after ?dub !coffee)	=	{ δ }
out (i after ?kwart)	=	{ δ }
out (i after !coffee)	=	∅
out (i after ?dub !tea)	=	∅
out (i after δ)	=	{ δ }
out (i after δ δ ?dub)	=	{ !coffee }

Implementation Relation uioco

i uioco s $=_{def}$ $\forall \sigma \in$ *Utraces* (s) : *out* (i after σ) $\subseteq$ *out* (s after σ)



uioco



out (i after ε) = { δ }

out (i after ?dub) = { !coffee }

out (i after ?dub !coffee) = { δ }

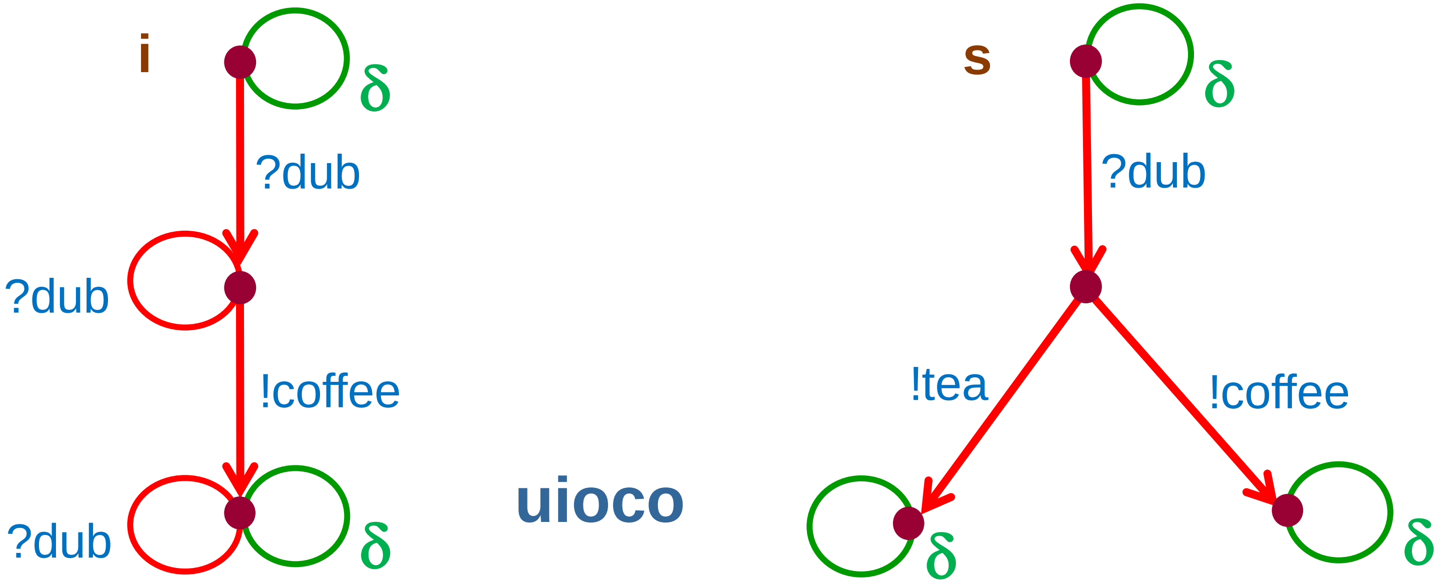
out (s after ε) = { δ }

out (s after ?dub) = { !coffee }

out (s after ?dub !coffee) = { δ }

Implementation Relation uioco

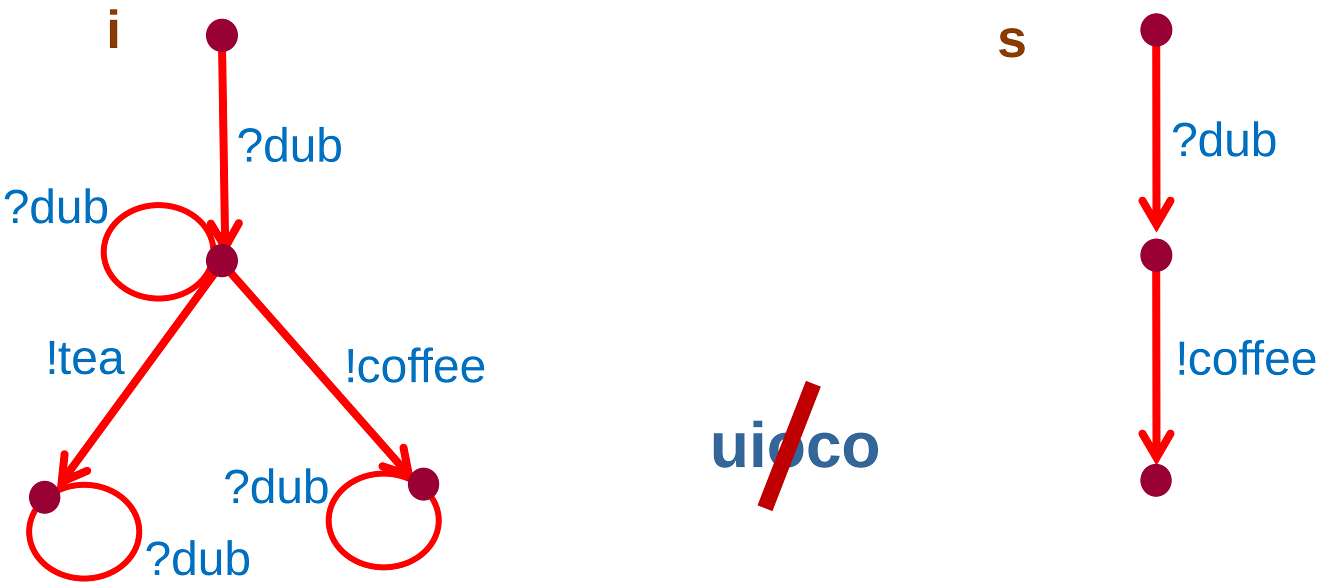
i **uioco** s $=_{def}$ $\forall\ \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after σ)** $\subseteq$ ***out*** **(s after σ)**



out (**i** **after** ?dub) = { !coffee } $\subseteq$ ***out*** (**s** **after** ?dub) = { !coffee. !tea }

Implementation Relation uioco

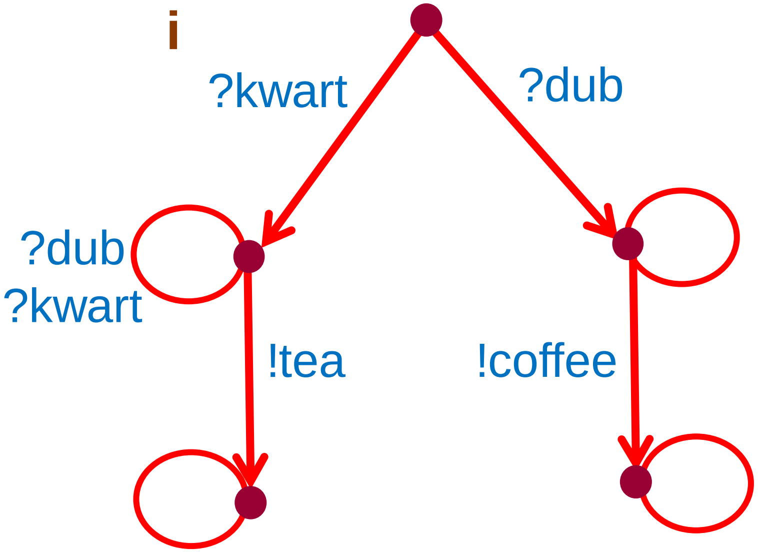
i uioco s $=_{def}$ $\forall\ \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after σ)** $\subseteq$ ***out*** **(s after σ)**



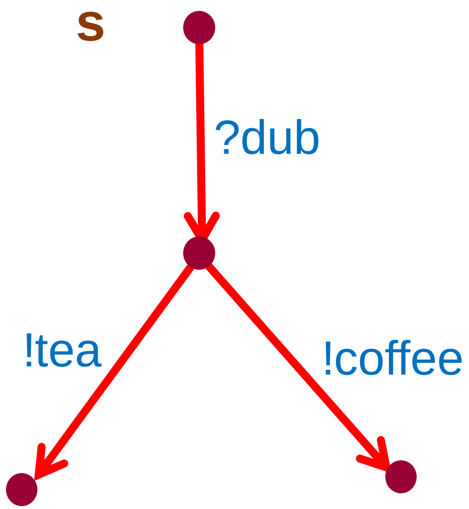
out (**i after** ?dub) = { !coffee, !tea } $\not\subseteq$ ***out*** (**s after** ?dub) = { !coffee }

Implementation Relation uioco

i uioco s $=_{def}$ $\forall \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after σ)** $\subseteq$ ***out*** **(s after σ)**



uioco



out (**i after** ?dub) = { !coffee }

out (**i after** ?kwart) = { !tea }

$\not\subseteq$

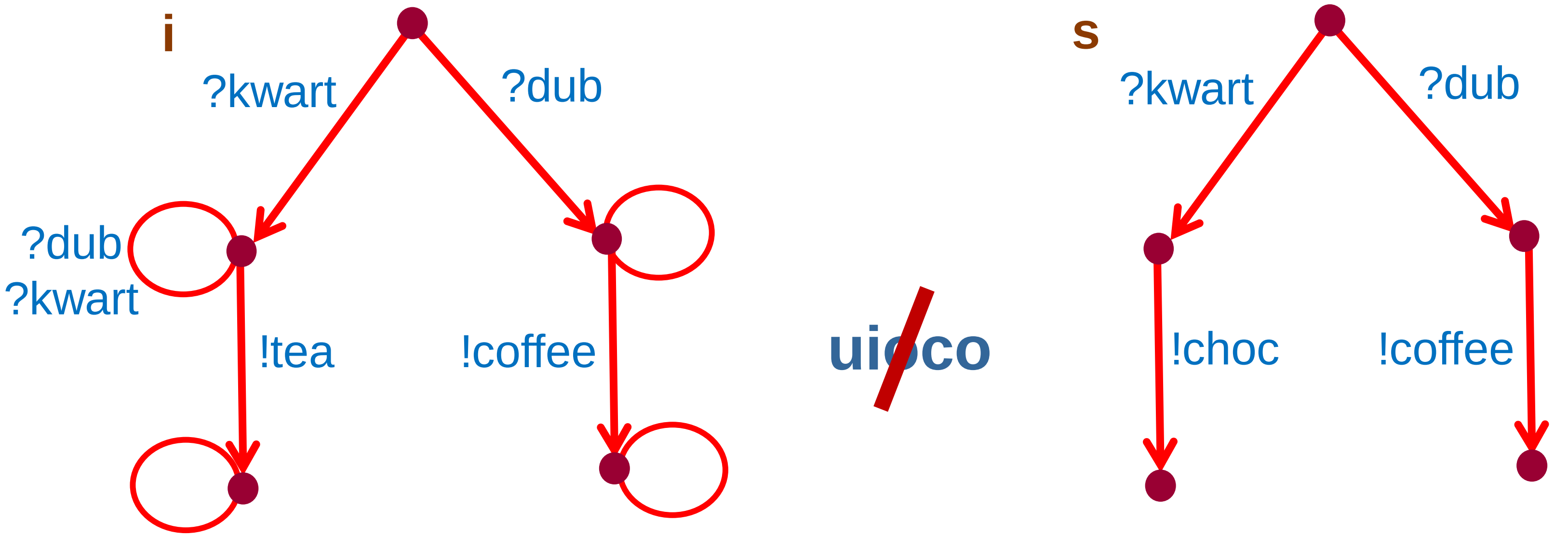
out (**s after** ?dub) = { !coffee, !tea }

out (**s after** ?kwart) = $\emptyset$

but ?kwart $\notin$ ***Utraces*** **(s)**

Implementation Relation uioco

i uioco s $=_{def}$ $\forall\, \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after σ)** $\subseteq$ ***out*** **(s after σ)**

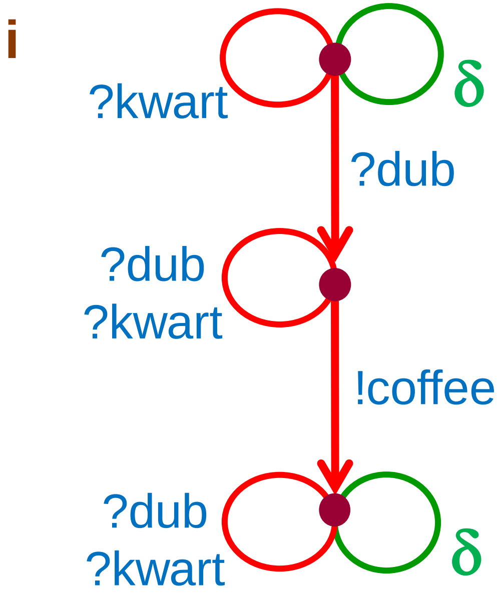


out (**i after** ?dub) = { !coffee } ***out*** (**s after** ?dub) = { !coffee }

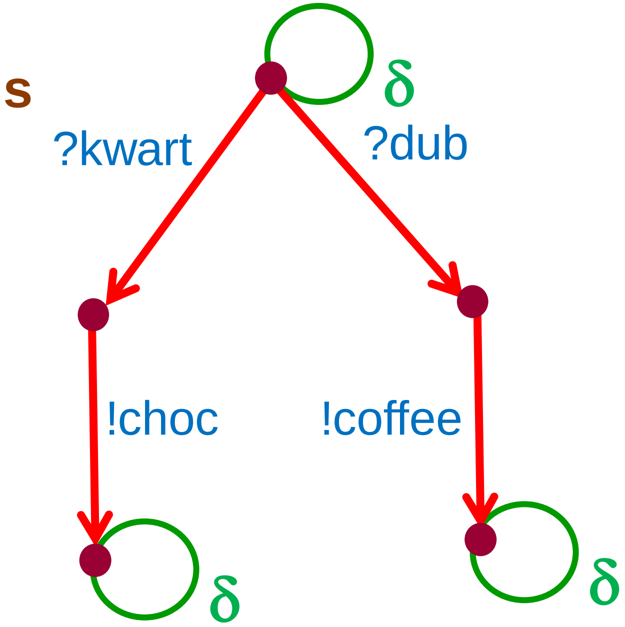
out (**i after** ?kwart) = { !tea } $\not\subseteq$ ***out*** (**s after** ?kwart) = { !choc }

Implementation Relation uioco

i **uioco** s $=_{\text{def}}$ $\forall\, \sigma \in \textit{Utraces}\,(s) : \; \textit{out}\,(i \text{ after } \sigma) \subseteq \textit{out}\,(s \text{ after } \sigma)$



$\cancel{\textbf{uioco}}$



out (i after ?dub) = { !coffee }

out (i after ?kwart) = { δ }

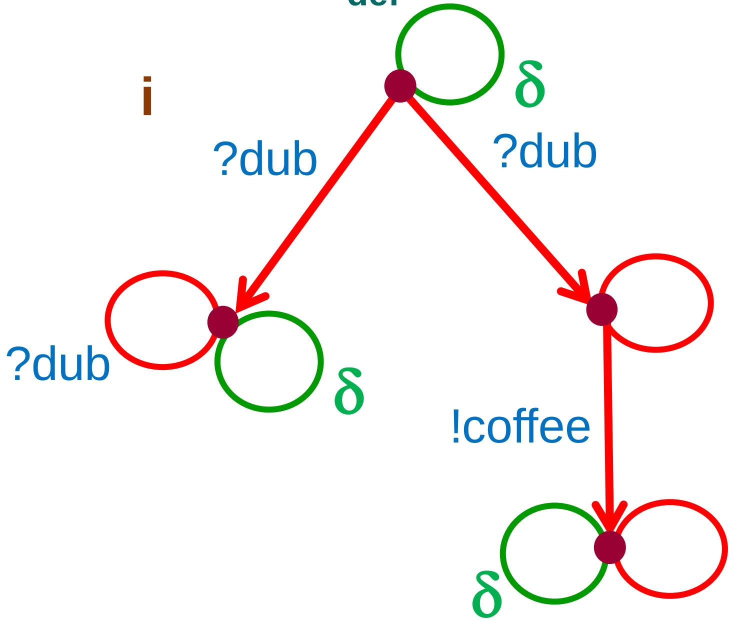
$\not\subseteq$

out (s after ?dub) = { !coffee }

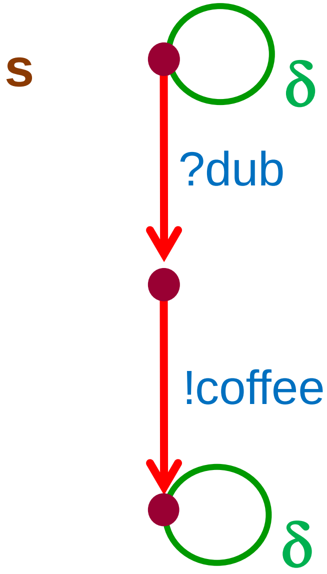
out (s after ?kwart) = { !choc }

Implementation Relation uioco

i **uioco** s $=_{def}$ $\forall\, \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after** σ**)** $\subseteq$ ***out*** **(s after** σ**)**



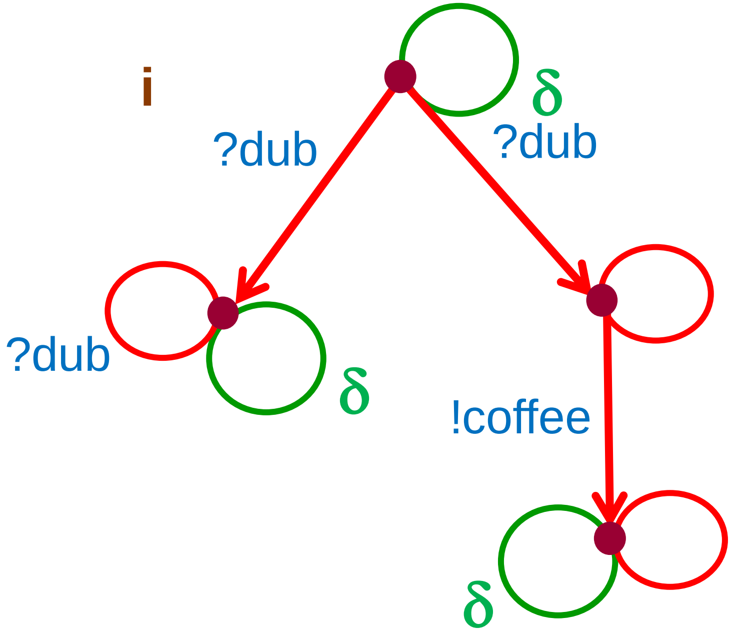
$\not$uioco



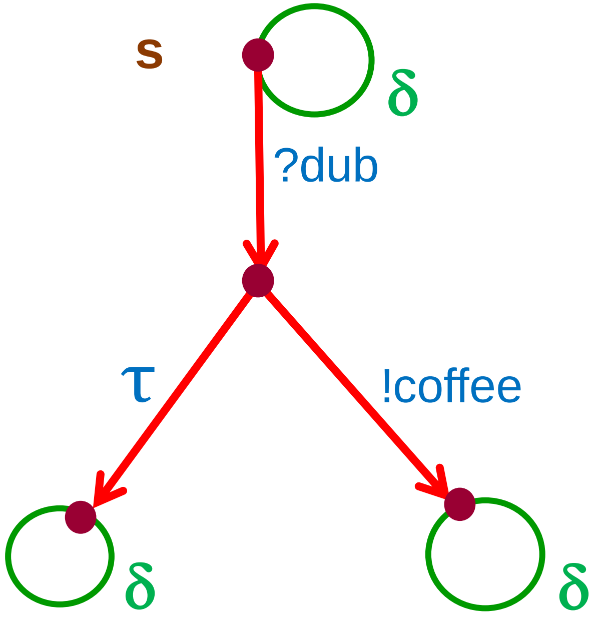
out (**i after** ?dub) = { !coffee, δ } $\not\subseteq$ ***out*** (**s after** ?dub) = { !coffee }

Implementation Relation uioco

$$\mathbf{i}\ \ \mathbf{uioco}\ \ \mathbf{s}\ \ =_{\mathrm{def}}\ \ \forall\, \sigma \in \textbf{\textit{Utraces}}\,(\mathbf{s}) :\ \ \textbf{\textit{out}}\,(\mathbf{i}\ \mathbf{after}\ \sigma)\ \subseteq\ \textbf{\textit{out}}\,(\mathbf{s}\ \mathbf{after}\ \sigma)$$



uioco

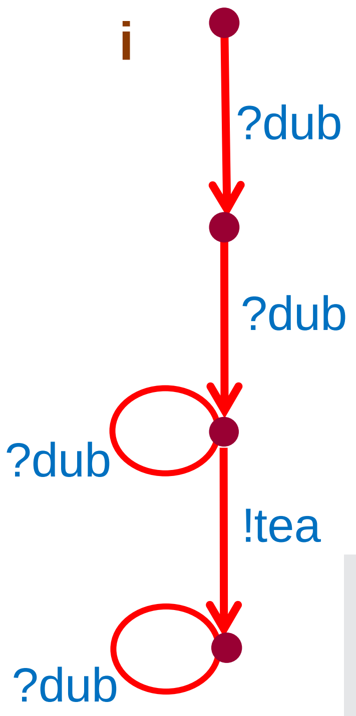


out (**i** **after** ?dub) = { !coffee, δ }

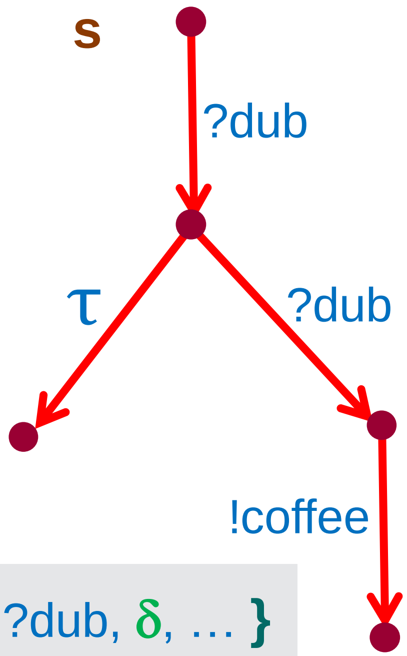
out (**s** **after** ?dub) = { !coffee, δ }

Implementation Relation uioco

i uioco s $=_{def}$ $\forall\, \sigma \in$ ***Utraces*** **(s) :** ***out*** **(i after** σ**)** $\subseteq$ ***out*** **(s after** σ**)**



uioco



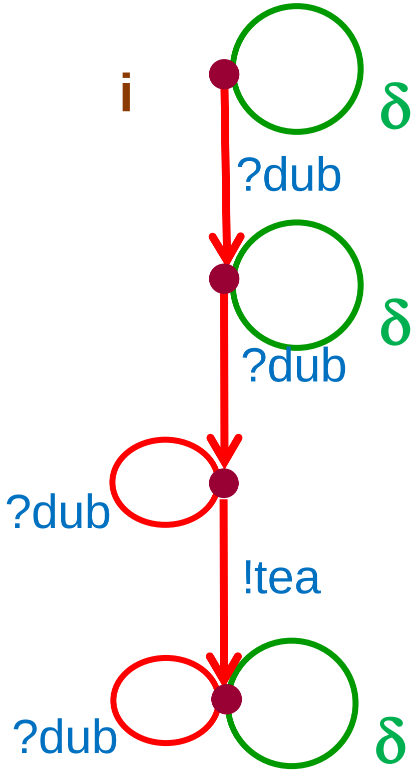
?dub ?dub $\notin$ ***Utraces*** **(s)** = **{ ε,** ?dub, δ, … **}**

because **s after** ?dub **refuses {** ?dub **}**

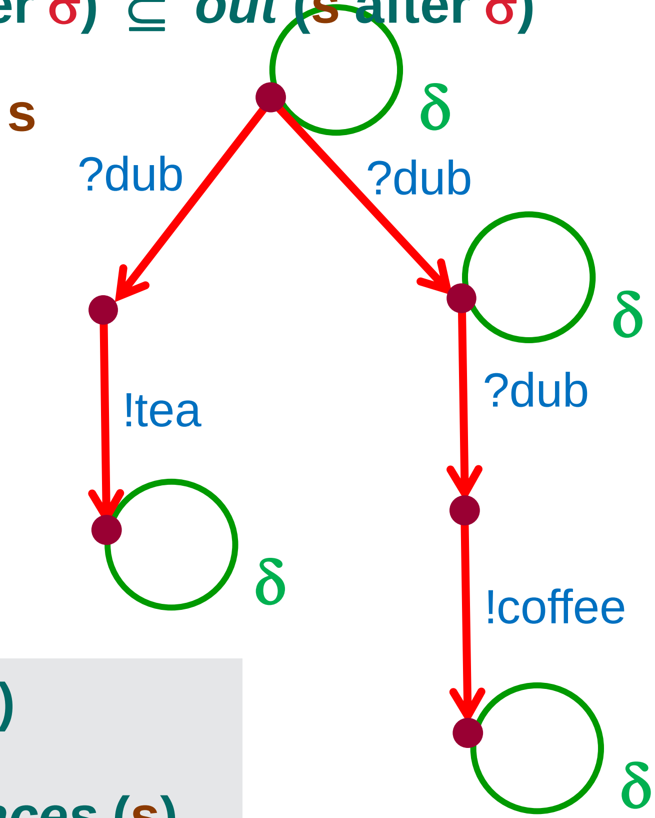
out (**i after** ?dub ?dub) = { !tea } $\not\subseteq$ ***out*** (**s after** ?dub ?dub) = { !coffee }

Implementation Relation uioco

$i \ \text{uioco} \ s \ =_{def} \ \forall \sigma \in \textit{Utraces}(s) : \ \textit{out}(i \text{ after } \sigma) \subseteq \textit{out}(s \text{ after } \sigma)$



$\not\text{uioco}$



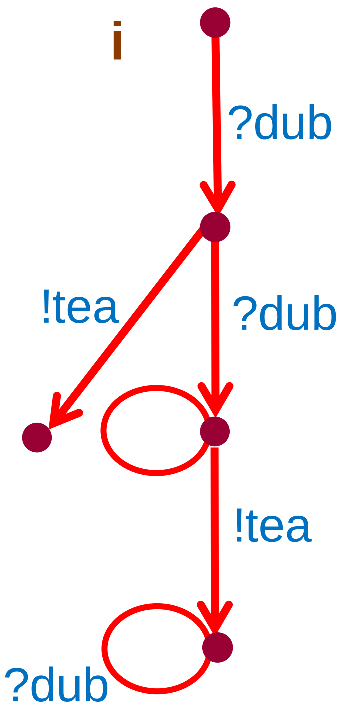
?dub ?dub $\notin$ ***Utraces*** **(s)**

but ?dub δ ?dub $\in$ ***Utraces*** **(s)**

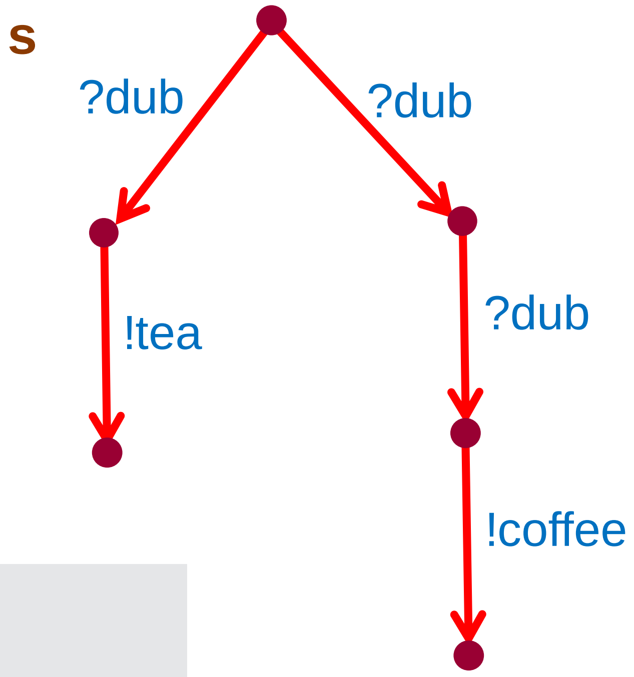
$\textit{out}(i \text{ after } ?dub\ \delta\ ?dub) = \{ !tea \} \not\subseteq \textit{out}(s \text{ after } ?dub\ \delta\ ?dub) = \{ !coffee \}$

Implementation Relation uioco

$\mathbf{i} \;\; \mathbf{uioco} \;\; \mathbf{s} \quad =_{def} \; \forall \, \sigma \in \mathit{\mathbf{Utraces}}\,(\mathbf{s}) : \; \mathit{\mathbf{out}}\,(\mathbf{i} \text{ after } \sigma) \subseteq \mathit{\mathbf{out}}\,(\mathbf{s} \text{ after } \sigma)$



uioco



?dub ?dub $\notin$ ***Utraces*** (**s**)

but ?dub δ ?dub $\in$ ***Utraces*** (**s**)

out (**i** **after** ?dub δ ?dub) = $\varnothing$ $\subseteq$ ***out*** (**s** **after** ?dub δ ?dub) = { !coffee }

Input/Output Conformance : *uioco*

$$i \;\; \textbf{uioco} \;\; s \quad =_{def} \quad \forall\, \sigma \in \textit{Utraces}\,(s) : \;\; \textit{out}\,(i \text{ after } \sigma) \;\subseteq\; \textit{out}\,(s \text{ after } \sigma)$$

s is a Labelled Transition System

i is (assumed to be) an input-enabled LTS

$$p \xrightarrow{\delta} p \quad \Leftrightarrow \quad \forall \;\; !x \;\in\; L_U \cup \{\tau\} .\;\; p \overset{!x}{\not\longrightarrow} \quad \Leftrightarrow \quad p \text{ refuses } L_U$$

$$\textit{Straces}\,(s) \quad = \quad \{ \;\sigma \;\in (\, L \cup \{\delta\}\,)^* \mid s \overset{\sigma}{\Longrightarrow} \;\}$$

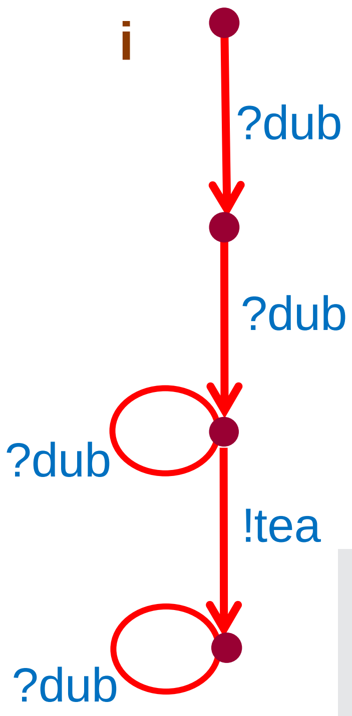
$$\textit{Utraces}\,(s) \quad = \quad \{\, \sigma \in \textit{Straces}\,(s) \;\; |$$

$$\forall \; \sigma_1 \; ?b \;\; \sigma_2 \; = \; \sigma : \;\; \textit{not}\,(\, s \;\; \text{after } \sigma_1 \text{ refuses } \; \{?b\}\,) \;\; \}$$

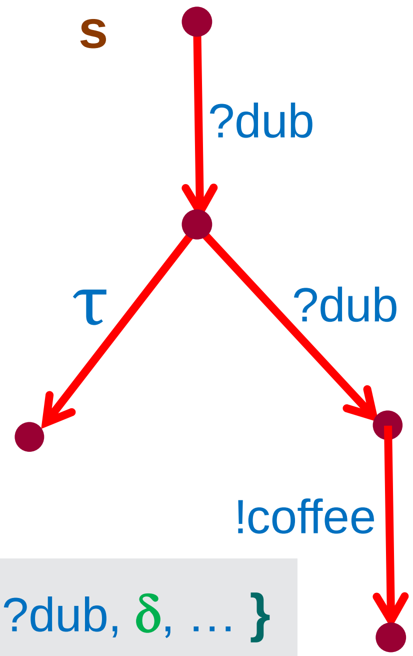
$$\textit{out}\,(\,P\,) \quad = \{ \;\; !x \in L_U \mid \exists\, p{\in}P : p \xrightarrow{!x} \} \;\cup\; \{\, \delta \mid \exists\, p{\in}P : p \xrightarrow{\delta} p \,\}$$

Implementation Relation uioco

$i \ \mathbf{uioco} \ s \ =_{def} \ \forall \sigma \in \textit{Utraces}(s) : \ \textit{out}(i \text{ after } \sigma) \subseteq \textit{out}(s \text{ after } \sigma)$



uioco



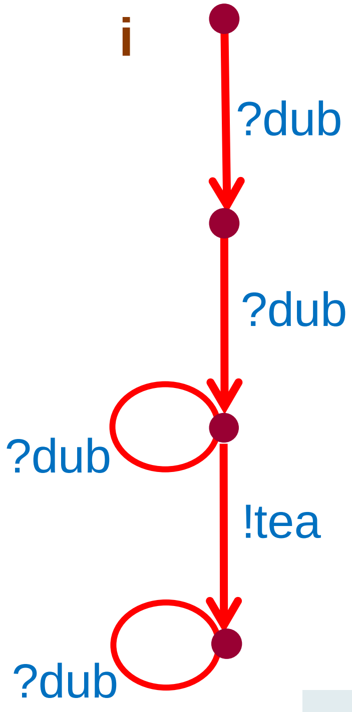
?dub ?dub $\notin$ ***Utraces*** **(s)** = { **ε,** ?dub, δ, … }

because **s after** ?dub **refuses {** ?dub **}**

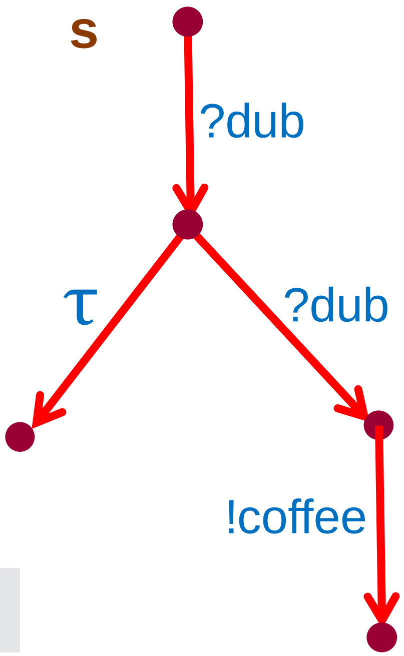
out (**i after** ?dub ?dub) = { !tea } $\not\subseteq$ ***out*** (**s after** ?dub ?dub) = { !coffee }

Alternative Implementation Relation ioco

$\mathbf{i}\ \ \textbf{ioco}\ \ \mathbf{s}\quad =_{\text{def}}\ \ \forall\, \sigma \in \textbf{\textit{Straces}}\,(\mathbf{s}) :\ \ \textbf{\textit{out}}\,(\mathbf{i}\ \textbf{after}\ \sigma)\ \subseteq\ \textbf{\textit{out}}\,(\mathbf{s}\ \textbf{after}\ \sigma)$



$\cancel{\textbf{ioco}}$



?dub ?dub $\in$ ***Straces*** **(s)**

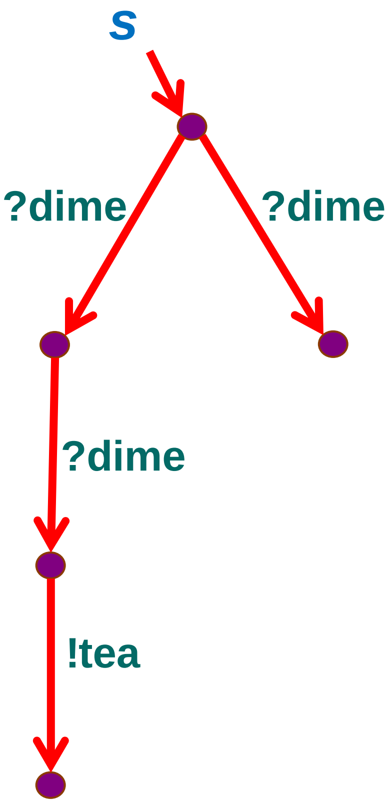
Straces (**s**) $=$ { $\sigma \in (\, L \cup \{\delta\}\,)^*$ | s $^{\sigma}$ }

out (**i** **after** ?dub ?dub) = { !tea } $\not\subseteq$ ***out*** (**s** **after** ?dub ?dub) = { !coffee }

Input/Output Conformance : (*u*)*ioco*

i uioco s $=_{def}$ $\forall\, \sigma \in$ *Utraces* (s) : *out* (i after σ) $\subseteq$ *out* (s after σ)

i ioco s $=_{def}$ $\forall\, \sigma \in$ *Straces* (s) : *out* (i after σ) $\subseteq$ *out* (s after σ)



Straces (s) $= \{\ \sigma \in (\ L \cup \{\delta\}\ )^* \mid s \overset{\sigma}{\Longrightarrow} \}$

Utraces (s) $= \{\ \sigma \in$ *Straces* (s) $\mid$

$\forall\, \sigma_1\ ?b\ \sigma_2 = \sigma$: *not* (s after σ_1 refuses $\{?b\}$) }

?dime ?dime $\in$ *Straces* (s)

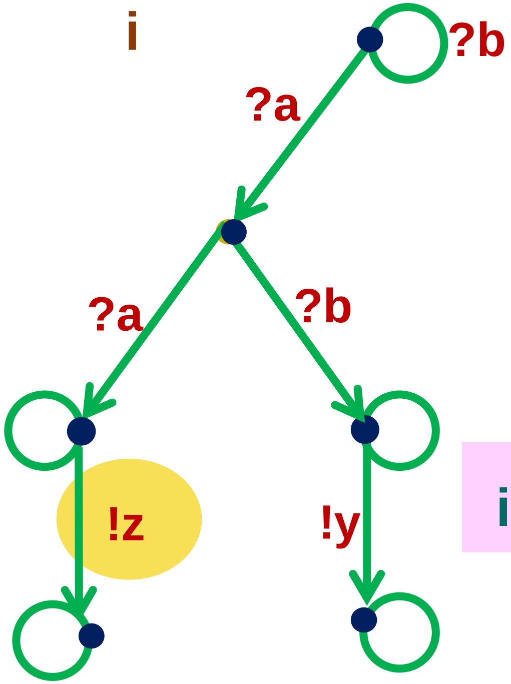
?dime ?dime $\notin$ *Utraces* (s)

ioco $\subset$ uioco

Input/Output Conformance : (*u*)*ioco*

i uioco s $=_{def}$ $\forall \sigma \in$ ***Utraces*** (s) : ***out*** (i after σ) $\subseteq$ ***out*** (s after σ)

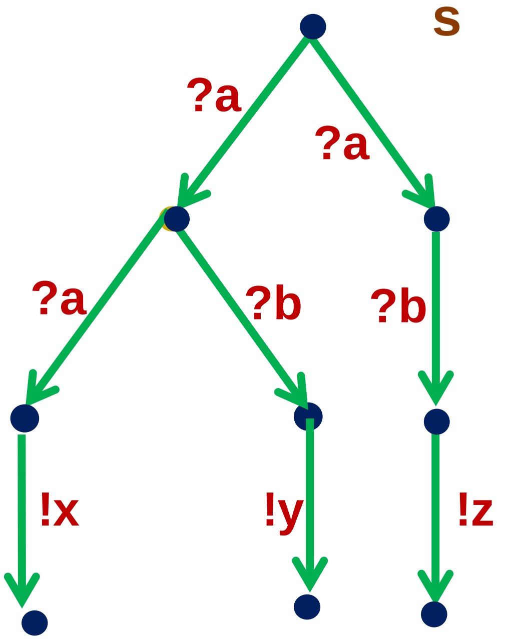
i ioco s $=_{def}$ $\forall \sigma \in$ ***Straces*** (s) : ***out*** (i after σ) $\subseteq$ ***out*** (s after σ)



i ~~ioco~~ s

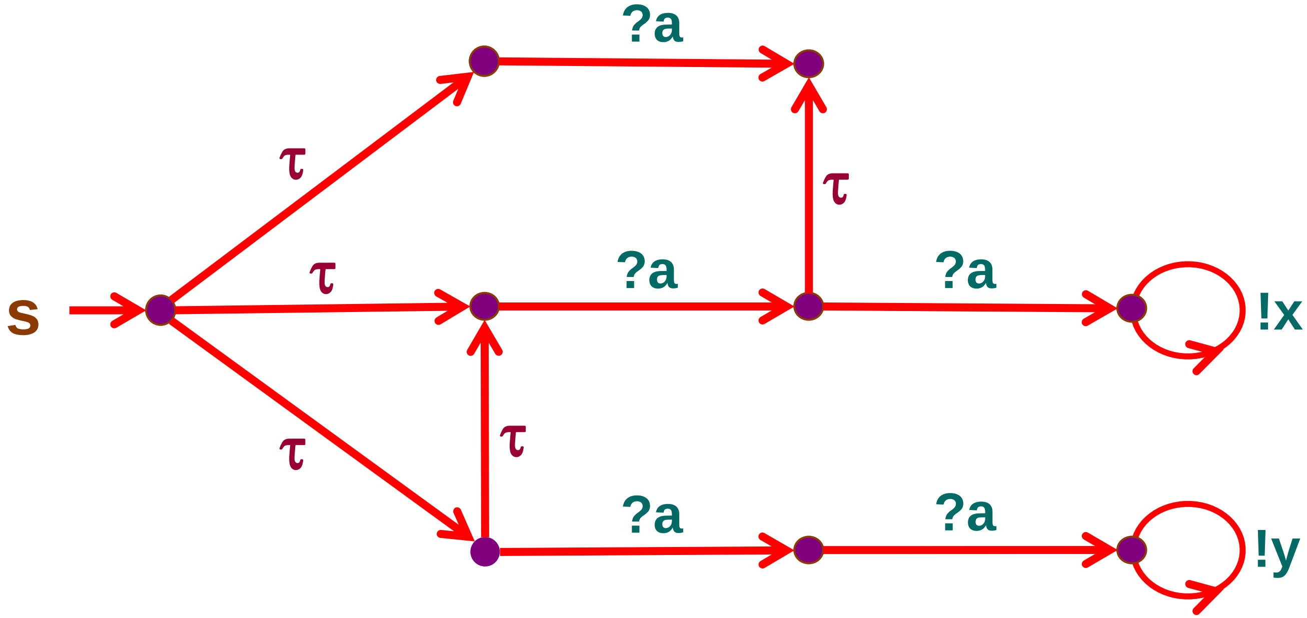
i uioco s

ioco $\subset$ uioco



Difference **ioco-uioco** is in *non-deterministic under-specification*

A non-*ioco*-implementable specification

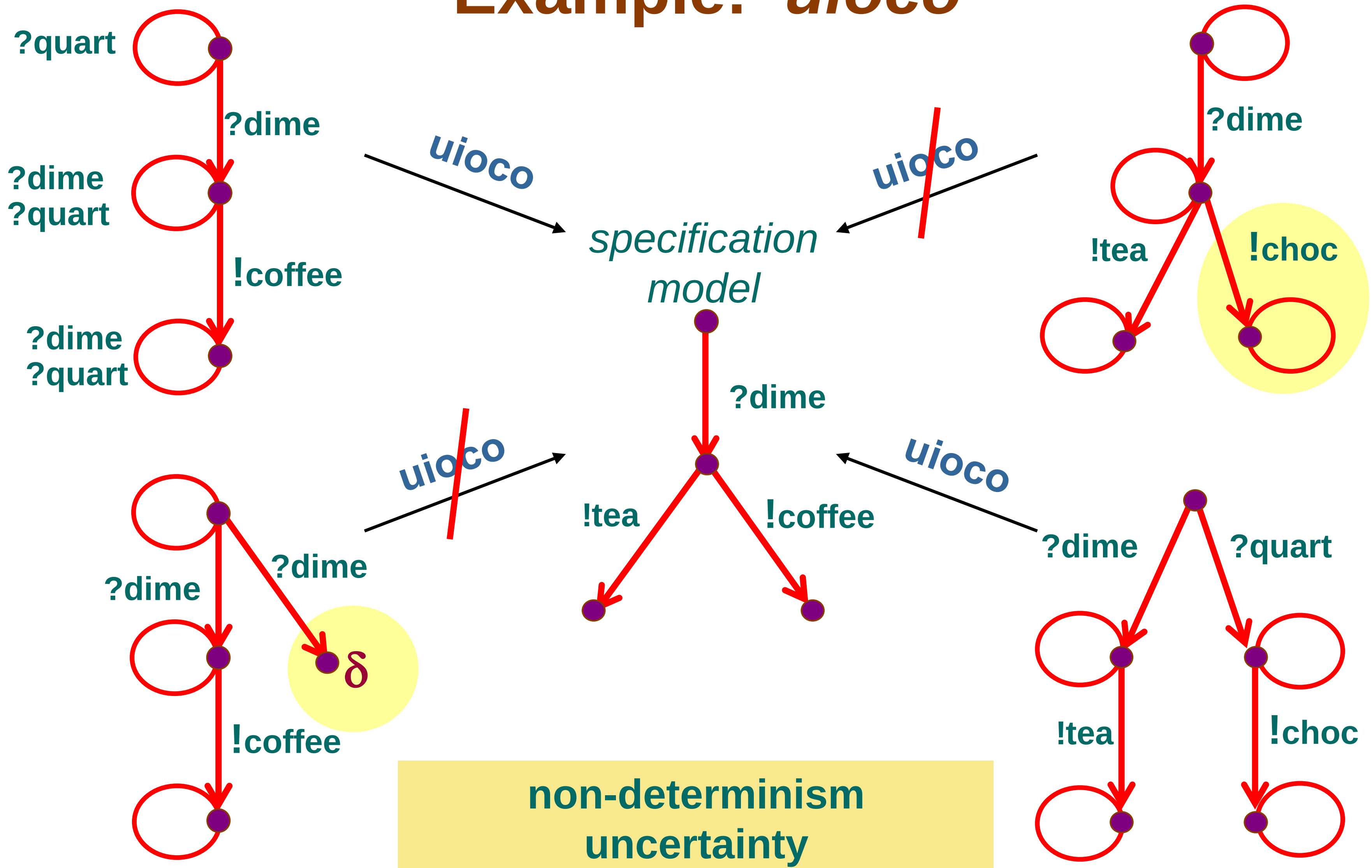


There is ***no*** implementation ***i*** **;** what would be ***out*** **(** ***i*** **after δ ?a δ ?a)** **?**

let *out* (***i*** after δ?aδ?a) = **X** then

- **X** ≠ ∅
- **X** ⊆ *out*(***i*** after δ?a?a) ⊆ *out*(***s*** after δ?a?a) = { !x }
- **X** ⊆ *out*(***i*** after ?aδ?a) ⊆ *out*(***s*** after ?aδ?a) = { !y }

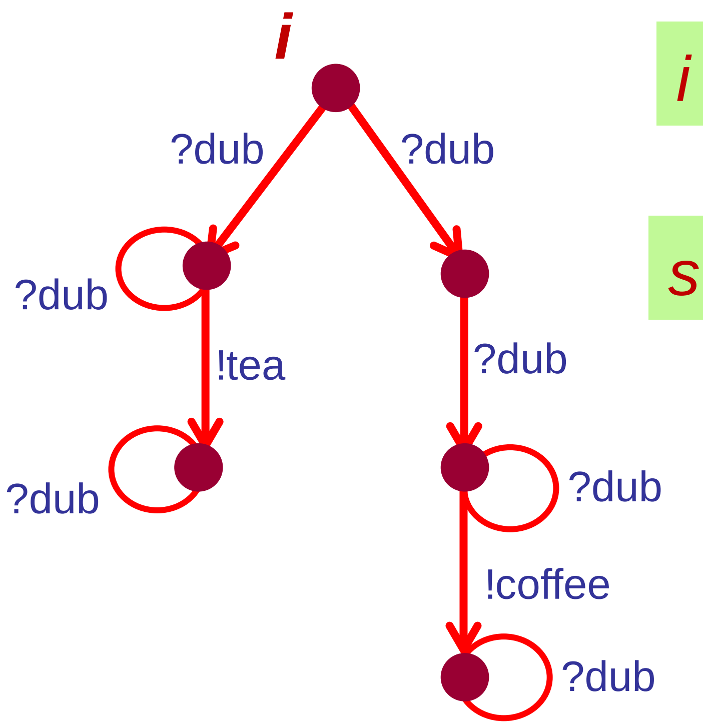
Example: *uioco*



non-determinism
uncertainty
under-specification

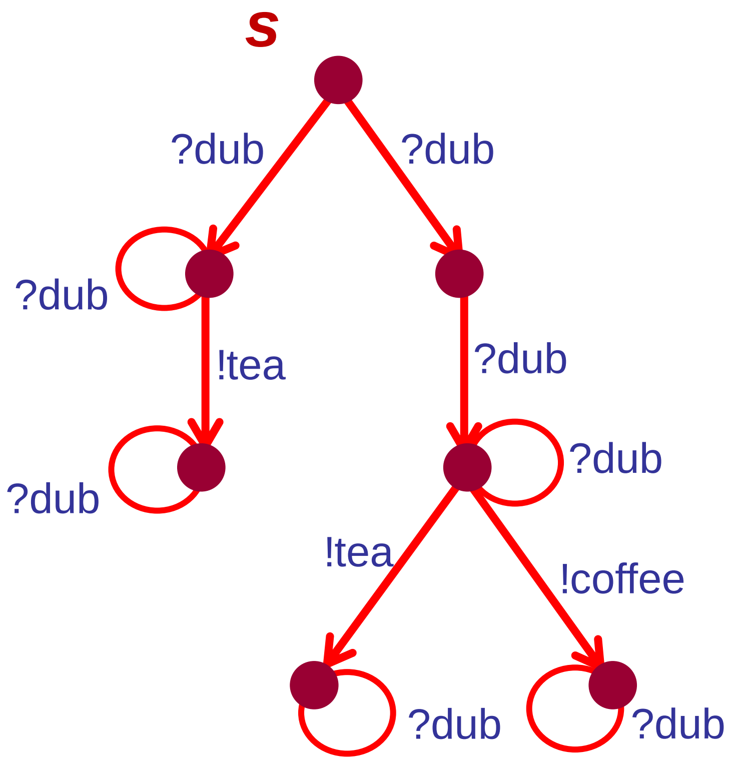
Example: (*u*)*ioco*

i uioco s $=_{def}$
$\forall\ \sigma \in$ *Utraces* (s) :
out (i after σ) $\subseteq$ *out* (s after σ)



i uioco *s*

s $\not\text{uioco}$ *i*

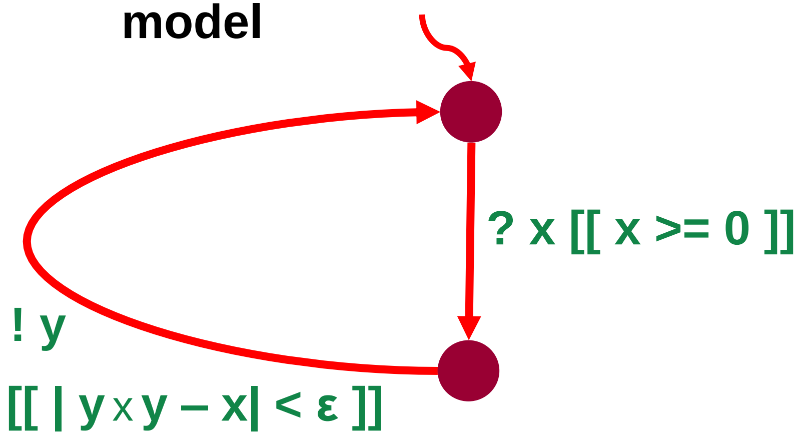


out (***i*** **after** ?dub.?dub) = *out* (***s*** **after** ?dub.?dub) = { !tea, !coffee }

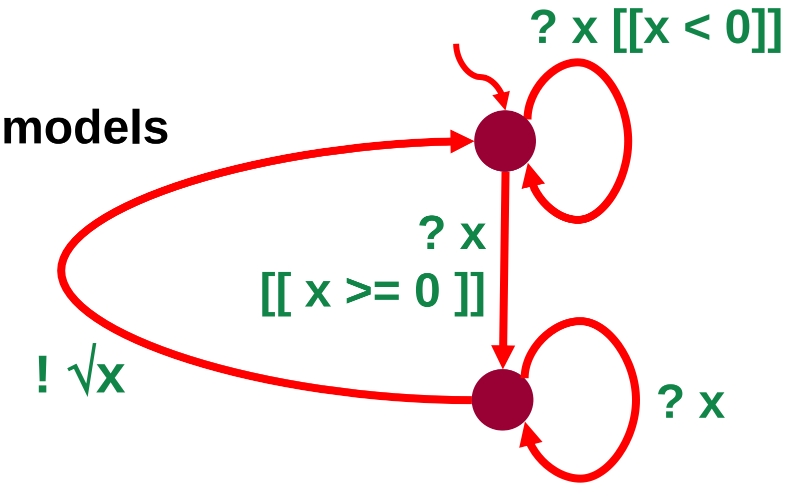
out (***i*** **after** ?dub.δ.?dub) = { !coffee } ≠ *out* (***s*** **after** ?dub.δ.?dub) = { !tea, !coffee }

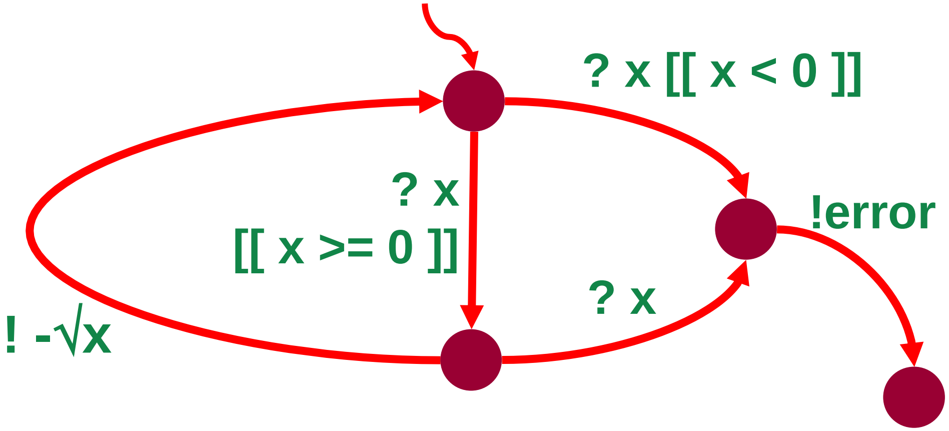
MBT : Nondeterminism, Underspecification

specification
model



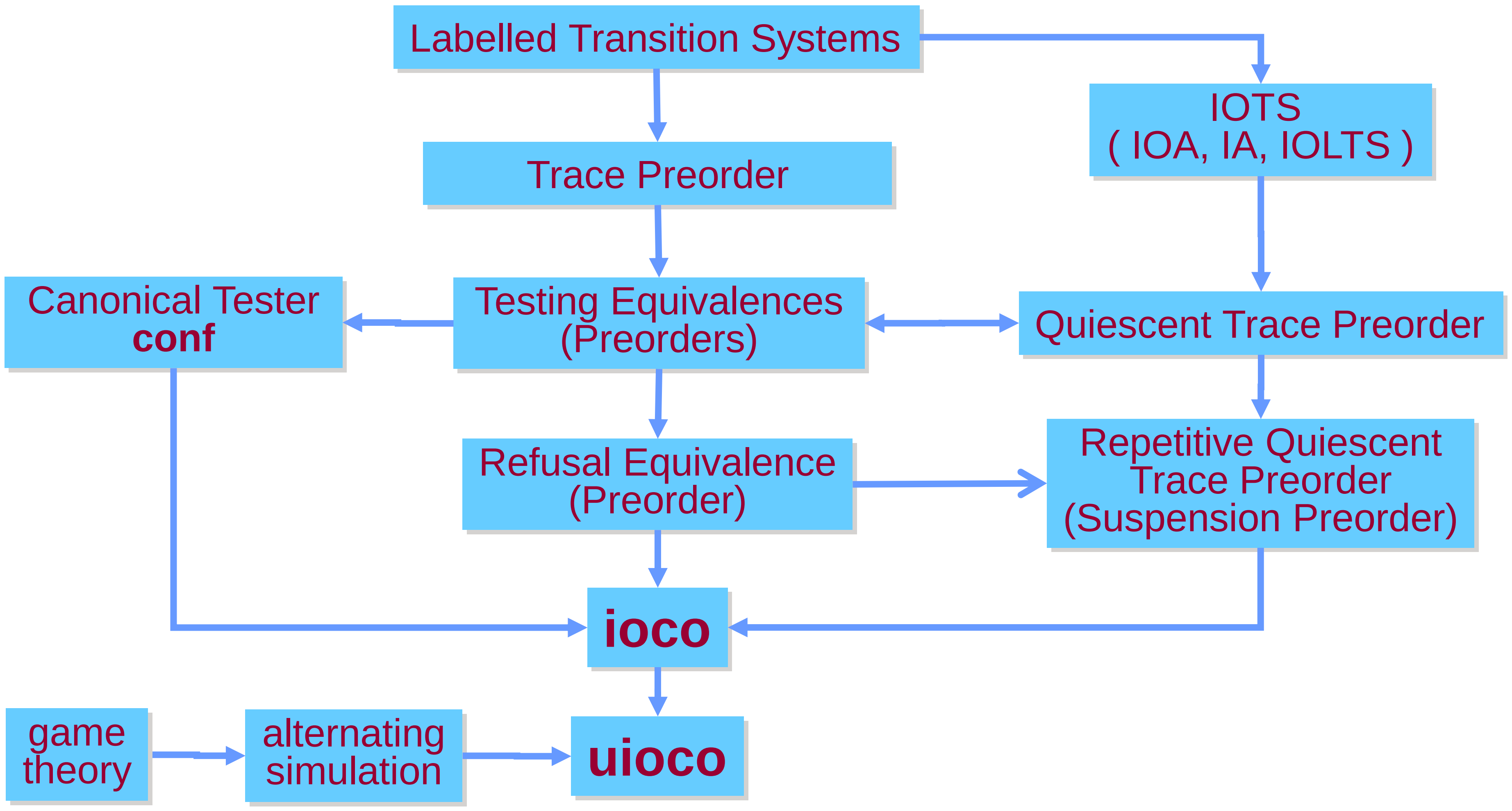
SUT models





- non-determinism
- under-specification
- specification of properties rather than construction

Genealogy of (*u*)*ioco*



Labelled Transition Systems

IOTS
(IOA, IA, IOLTS)

Trace Preorder

Canonical Tester
conf

Testing Equivalences
(Preorders)

Quiescent Trace Preorder

Refusal Equivalence
(Preorder)

Repetitive Quiescent
Trace Preorder
(Suspension Preorder)

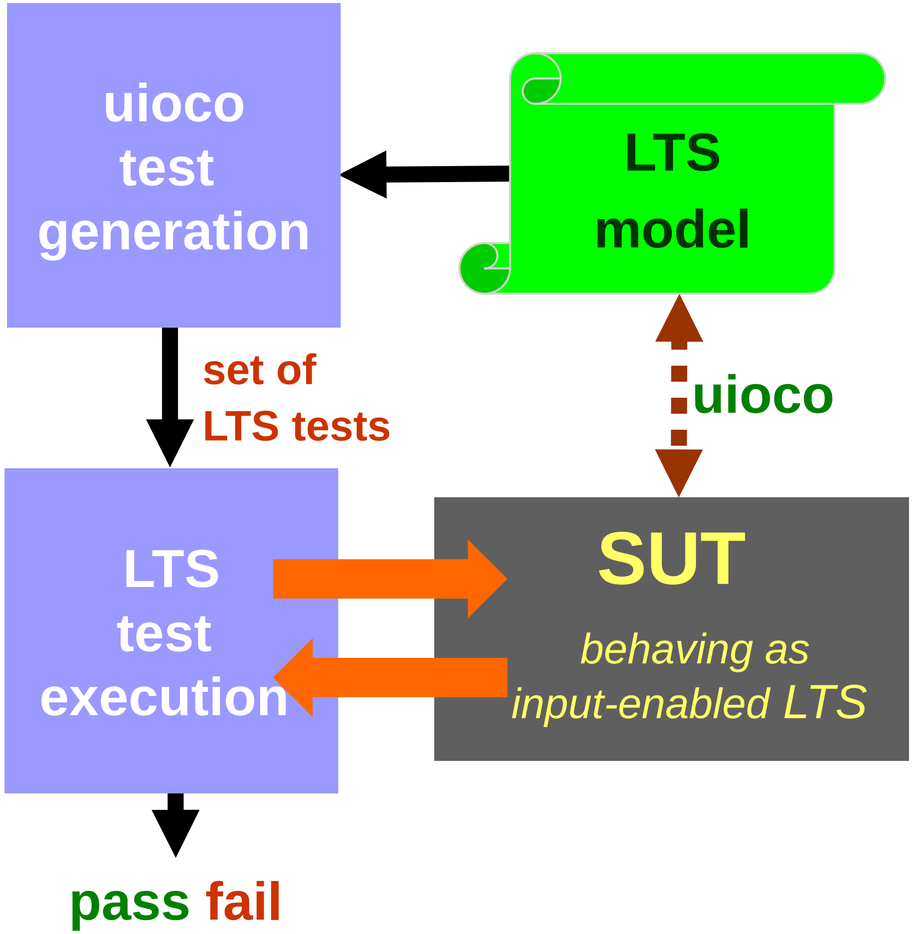
ioco

game
theory

alternating
simulation

uioco

MBT : Labelled Transitions Systems



MBT with LTS topics:

- ☞ specification model
- ☞ system under test (SUT)
- ☞ SUT model
- ☞ conformance **uioco**
- ☞ test cases
- ☞ test generation
- ☞ test execution
- ☞ test result analysis
- ☞ sound & exhaustive