A novel method for derivation of a test with guaranteed coverage for LTS

Gromov Maxim

gromov@sibmail.com

Tomsk State University

What is testing?

- Testing — is experimenting
- There is a formal specificaion
- Implementation is a «black box» with known interfaces
- Test — is prescription what to apply and what to observe at every moment
- There is a formal definition, what to consider conformant

FSM and LTS

Finite State Machine

- Specification is FSM
- Conformance relation is quasi-reduction
- Implementation is suposed to behavior as an FSM

Labelled Transition System

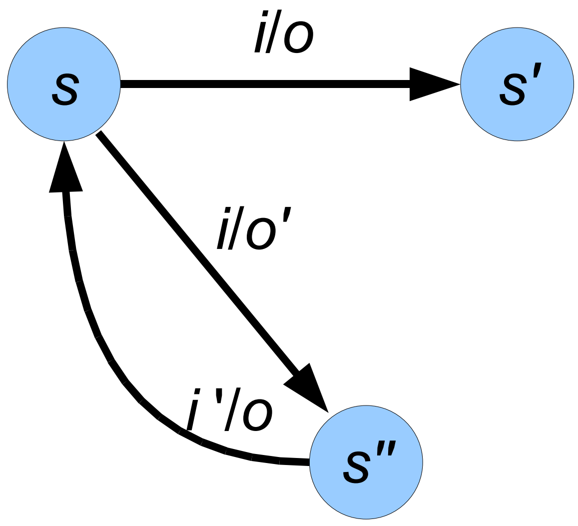
- Specification is LTS
- Conformance relation is **ioco**
- Implementation is suposed to behavior as an LTS

Finite State Machine

<S, I, O, s_0, λ>

- S – set of states
- I and O – disjoint sets of inputs and outputs
- $\lambda \subseteq S \times I \times O \times S$ – transition relation;

  $(s, i, o, s') \in \lambda$



Conformance with FSM

$$Im \leq Sp$$

$\forall \sigma \in Tr(Sp)$ (for any valid word of specification)

IF $\sigma \in Tr(Im)$ (it's valid for implementation) **THEN**

in(Im **after** σ**)** $\supseteq$ **in(**Sp **after** σ**)** (input behaviour of implementation **is larger** then one of the specification)

$\forall\ i \in$ **in(**Sp **after** σ**)**

out(Im **after** $\boldsymbol{\sigma}$, i**)** $\subseteq$ **out(**Sp **after** $\boldsymbol{\sigma}$, i**)**

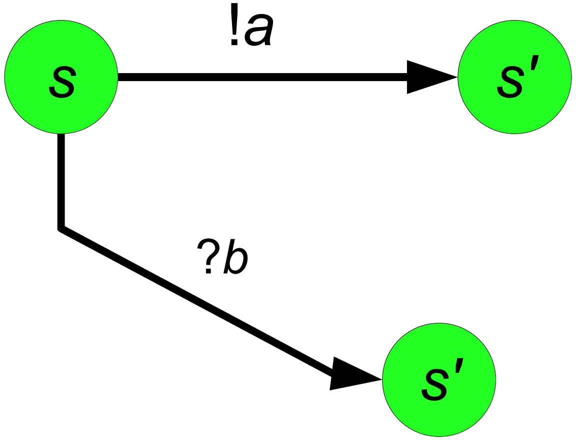
(output behaviour of implementation is **not larger** then one of the specification)

Labeled Transition System

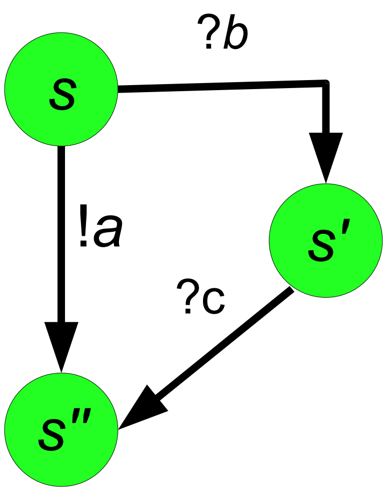
$<S, I, O, s_0, \lambda>$

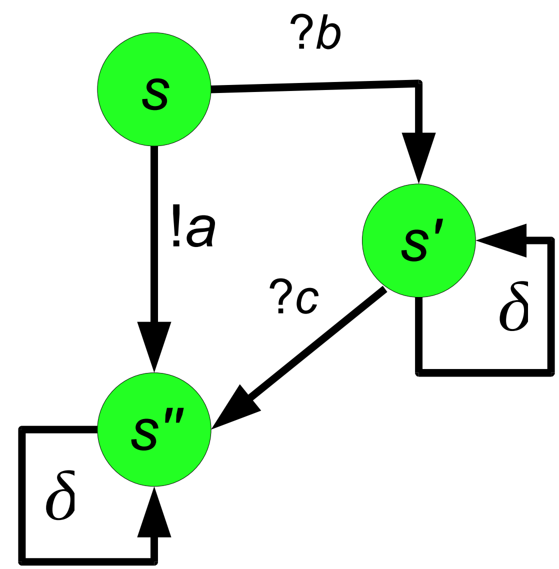
- S – set of states
- I and O – disjoint sets of inputs and outputs
- $\lambda \subseteq S \times (I \cup O) \times S$ – transition relation;

$(s, a, s') \in \lambda$



Quiescence





Conformance with LTS

Im **iocop** *Sp*

$\forall \sigma \in s\text{-}traces(Sp)$ (for any valid word of specification)

IF $\sigma \in s\text{-}traces(Im)$ (it's valid for implementation)

THEN

in(*Im* **after** σ**)** $\supseteq$ **in(***Sp* **after** σ**)** (input behaviour of implementation **is larger** then one of the specification)

out(*Im* **after** σ**)** $\subseteq$ **out(***Sp* **after** σ**)** (output behaviour of implementation is **not larger** then one of the specification)

Fault model

FSM

<*F*, ≤, *E*>, where

F – specification (a state machine)

≤ – quasi-reduction

E – set of FSMs, describing faulty behaviour

LTS

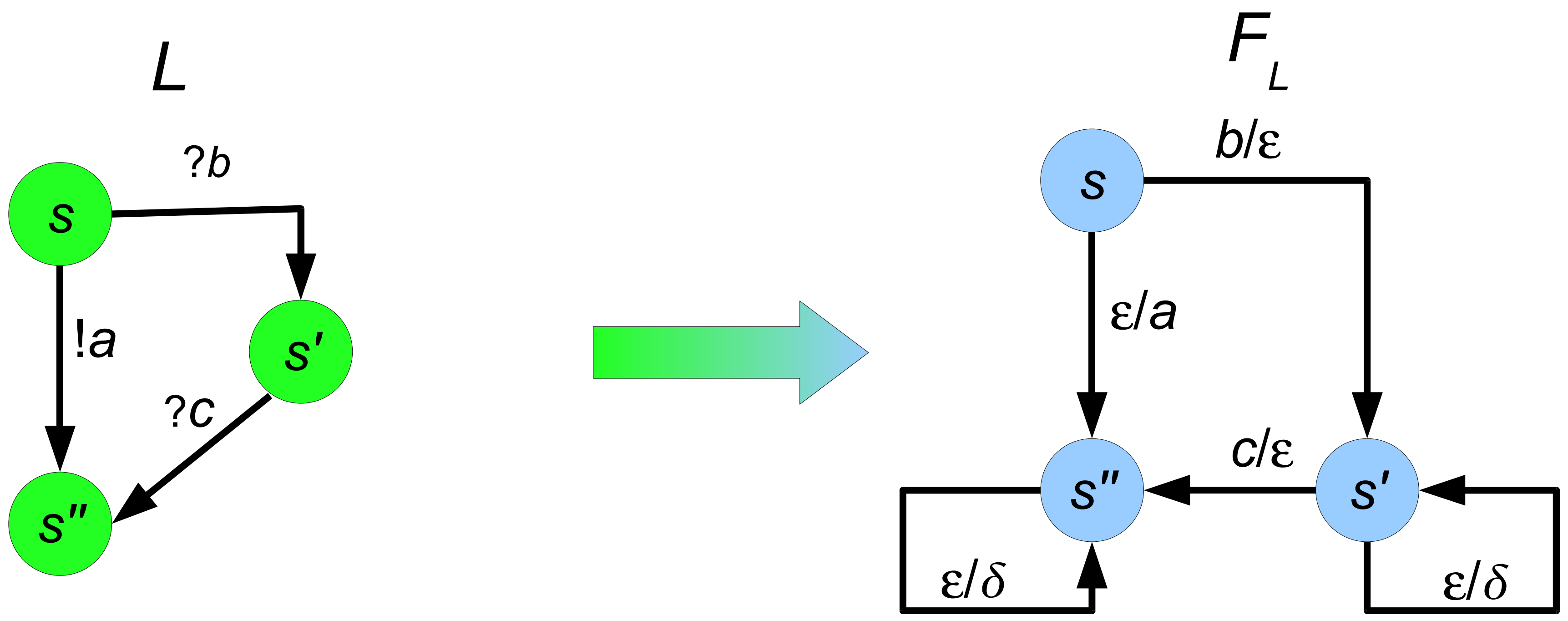
<*L*, iocop, *E*>, where

L – specification (a state machine)

iocop – iocop relation

E – set of LTSes, describing faulty behaviour

From LTS to FSM

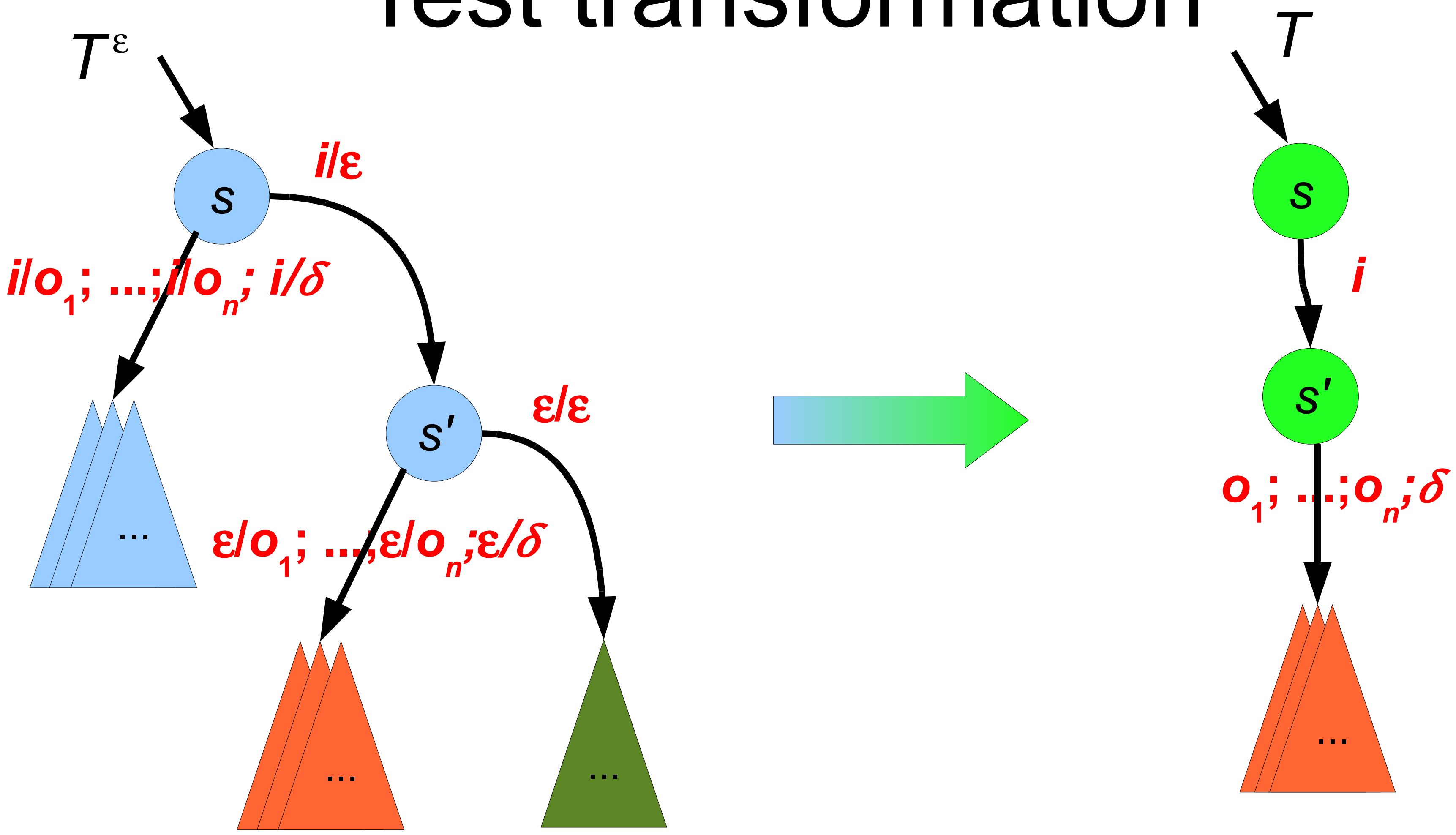


$$L_1 \to F_1^{\varepsilon}$$

$$L_2 \to F_2^{\varepsilon}$$

$$L_1 \,\mathbf{iocop}\, L_2 \Leftrightarrow F_1^{\varepsilon} \leq F_2^{\varepsilon}$$

Test transformation



If F^{ε} passes T^{ε} then L passes T

If F^{ε} fails T^{ε} then L fails T

Test derivation for an LTS

- Given an LTS fault model;
- Transform all LTSes to FSMs to get an FSM fault model;
- Build complete test for the FSM fault model;
- Transform FSM test to LTS test;
- Obtained test will be complete in respect to the given LTS fault model

Questions?

Thank you for your attention!!