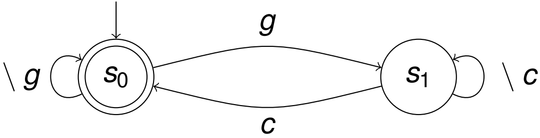
Embedded Systems Specification and Design

David Kendall

Introduction

- Specifying system properties
- Checking system properties
- Complement of a FSM
- Synchronous product of FSMs

Specifying a temporal property



- This FSM, M_{spec}, specifies the property that whenever the green button is pressed, eventually a cup of coffee is produced
- We abuse notation by writing
  $\setminus g$ to mean $A \setminus \{g\}$ where A is the alphabet of M_{sys}, i.e.
  $\setminus g = \{c, p, r, t\}$

Specifying a temporal property

- Notice that the specification is concerned with just one particular property.
- We may have many such specifications for different properties.
- The FSM is complete and deterministic. This turns out to be important.
- Given any FSM, it is possible to construct a complete, deterministic FSM that recognises the same language
- Now check if $L(M_{sys}) \subseteq L(M_{spec})$
- How?

Checking temporal properties

- Notice for any sets R and S
  $R \subseteq S$ is equivalent to $R \cap \overline{S} = \{\}$
- So we can check

$$L(M_{sys}) \cap \overline{L(M_{spec})} = \{\}$$

- This is very useful because there are 3 operations on FSMs that give us all we need to automate this test:
  - Complement: $\overline{M}$
    $L(\overline{M}) = \overline{L(M)} = A^* \setminus L(M)$
  - Synchronous product: $M_1 \otimes M_2$
    $L(M_1 \otimes M_2) = L(M_1) \cap L(M_2)$
  - Test for emptiness
    $L(M) = \{\}$ iff no cycle reachable from initial state contains an acceptance state

Checking temporal properties

What is actually checked?

$$L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$$

Checking temporal properties

What is actually checked?

$$L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$$

If true, we know...

$$L(M_{sys}) \subseteq L(M_{spec})$$

Checking temporal properties

What is actually checked?

$$L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$$

If true, we know...

$$L(M_{sys}) \subseteq L(M_{spec})$$

And so...

All actual behaviours are permissible
The system satisfies its specification

Complement of a FSM

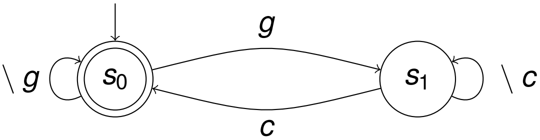
- Let $M = (S, s^0, A, E, F)$
- The complement of M is denoted $\overline{M}$ and defined by

$$\overline{M} = (S, s^0, A, E, S \setminus F)$$

- i.e. just ‘flip’ all states: acceptance $\rightarrow$ non-acceptance; non-acceptance $\rightarrow$ acceptance
- $L(\overline{M}) = A^* \setminus L(M) = \overline{L(M)}$
- Notice that this operation is defined only on complete FSMs

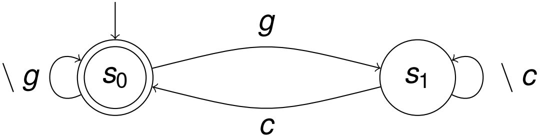
Complement of a FSM

- M

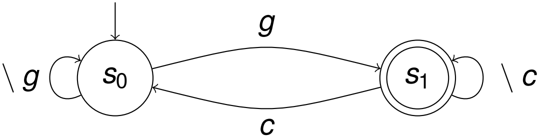


Complement of a FSM

- M



- $\overline{M}$



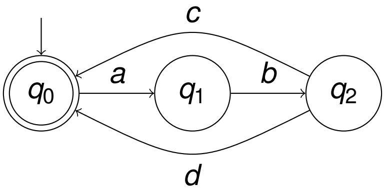
Product of FSMs

Definition ($M_1 \otimes M_2$)

Let $M_1 = (S_1, s_1^0, A_1, E_1, F_1)$ and $M_2 = (S_2, s_2^0, A_2, E_2, F_2)$ be FSMs. The synchronous product of M_1 and M_2 is denoted $M_1 \otimes M_2$ and is the FSM given by (S, s^0, A, E, F) where

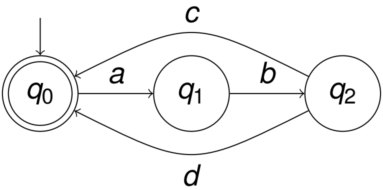
- $S = S_1 \times S_2$
- $s^0 = (s_1^0, s_2^0)$
- $A = A_1 \cup A_2$
- $E =$
  $\{((s_1, s_2), a, (s_1', s_2')) \mid (s_1, a, s_1') \in E_1 \wedge (s_2, a, s_2') \in E_2\}$
- $F = F_1 \times F_2$

Product of FSMs (Example)

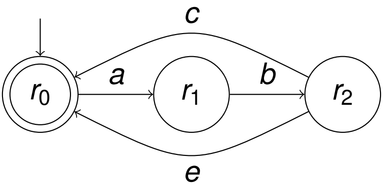


M_1

Product of FSMs (Example)

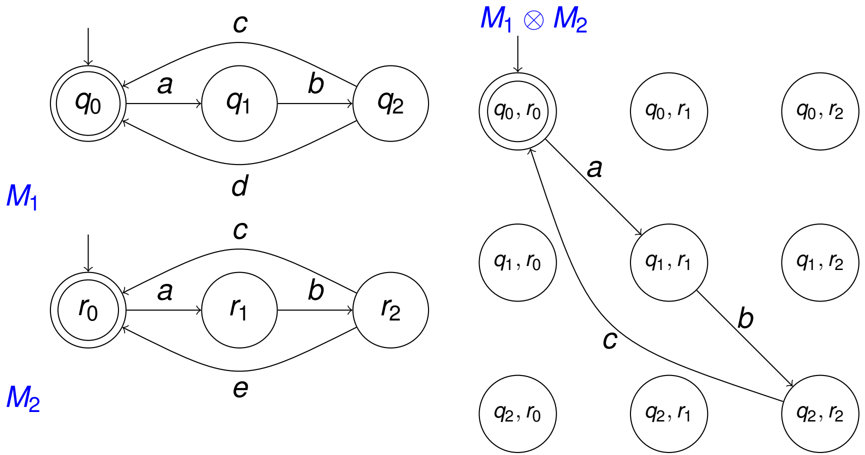


M_1



M_2

Product of FSMs (Example)

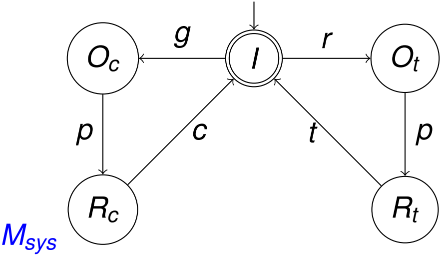


Checking the drinks machine

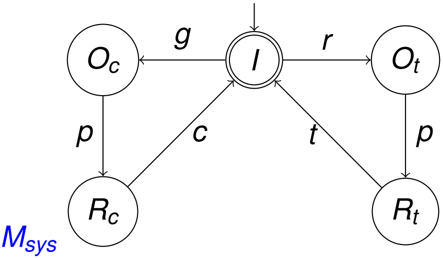
- When the green button is pressed, eventually we get a coffee

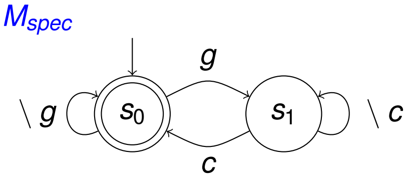
Checking the drinks machine

- When the green button is pressed, eventually we get a coffee



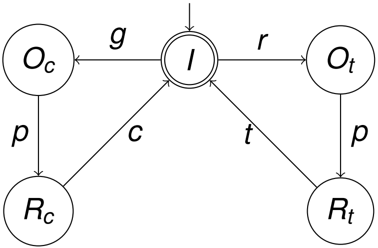
- When the green button is pressed, eventually we get a coffee





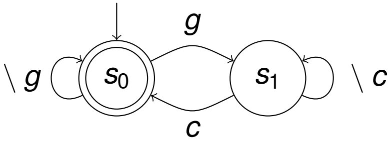
Checking the drinks machine

- When the green button is pressed, eventually we get a coffee

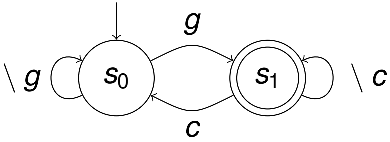


M_{sys}

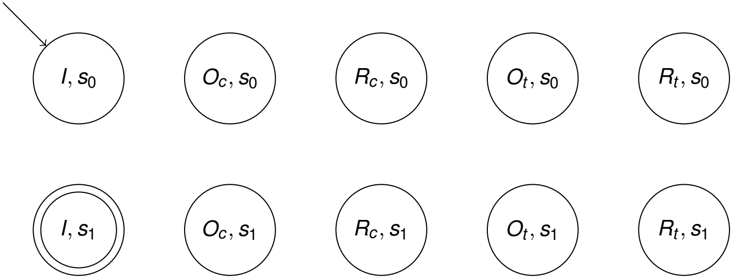
M_{spec}



$\overline{M_{spec}}$

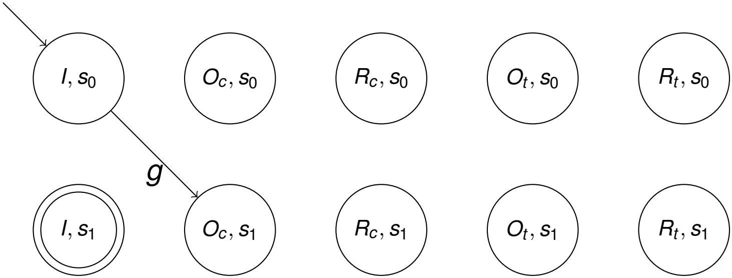


Checking the drinks machine



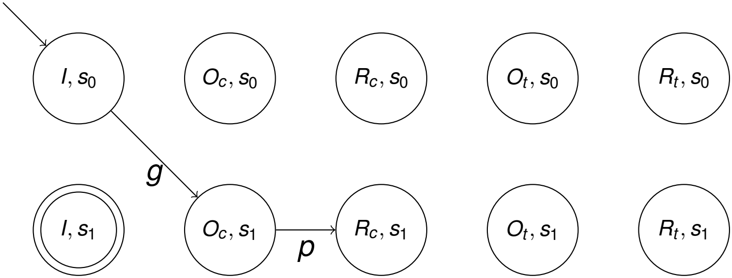
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



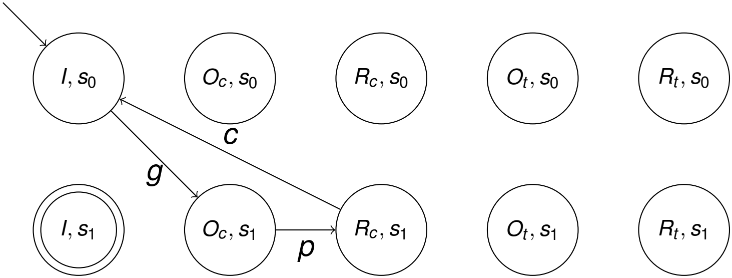
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



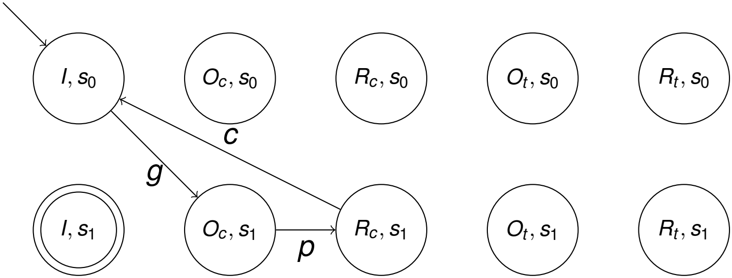
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

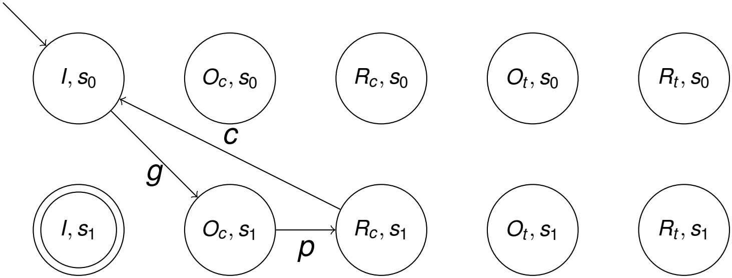
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- No acceptance cycle reachable from initial state

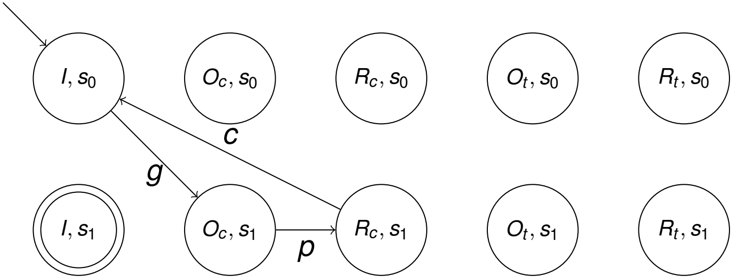
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- No acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$

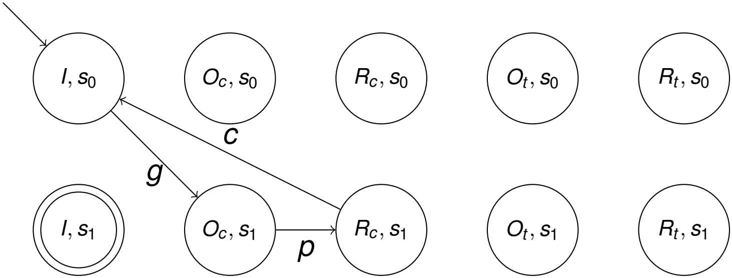
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- No acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$
- $L(M_{sys}) \subseteq L(M_{spec})$

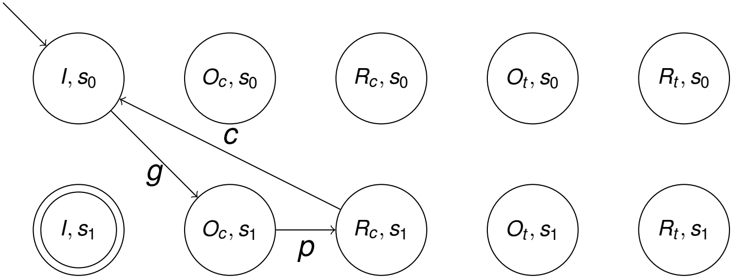
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- No acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$
- $L(M_{sys}) \subseteq L(M_{spec})$
- Property is satisfied

Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

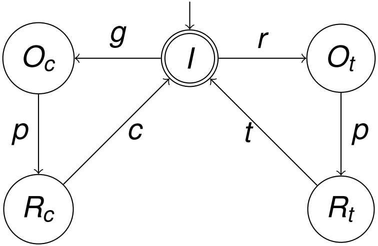
- No acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) = \{\}$
- $L(M_{sys}) \subseteq L(M_{spec})$
- Property is satisfied
- Note that here only edges whose source state is reachable from the initial state are shown.

Checking the drinks machine

- When the red button is pressed, eventually we get a coffee

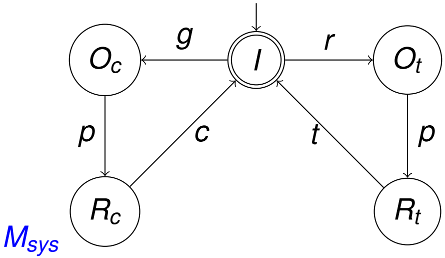
Checking the drinks machine

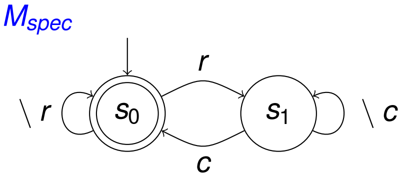
- When the red button is pressed, eventually we get a coffee



M_{sys}

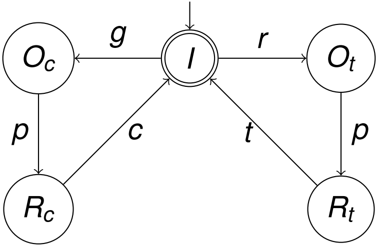
- When the red button is pressed, eventually we get a coffee





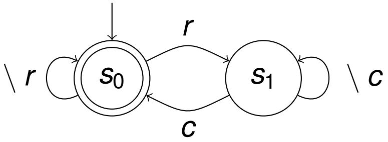
Checking the drinks machine

- When the red button is pressed, eventually we get a coffee

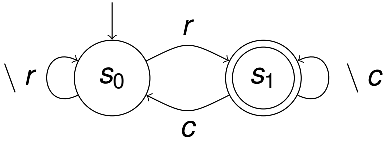


M_{sys}

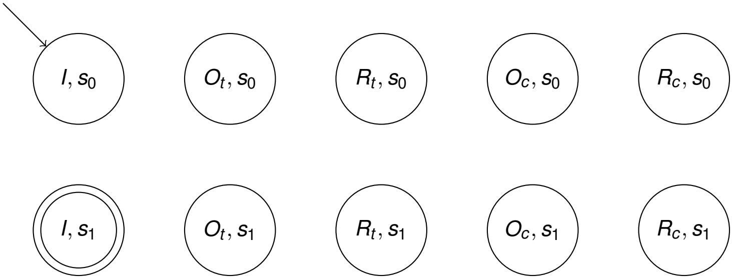
M_{spec}



$\overline{M_{spec}}$

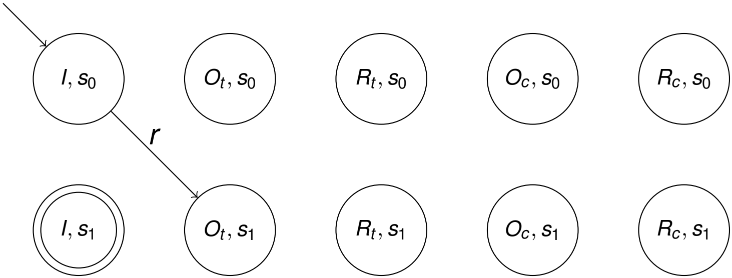


Checking the drinks machine



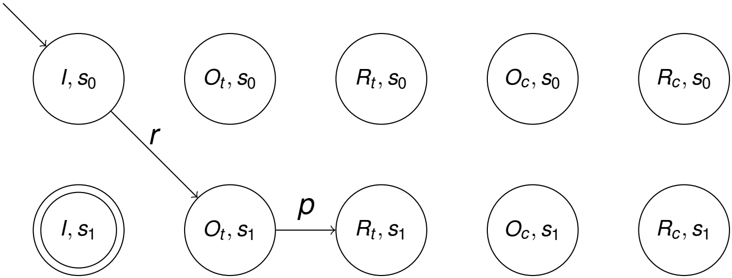
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



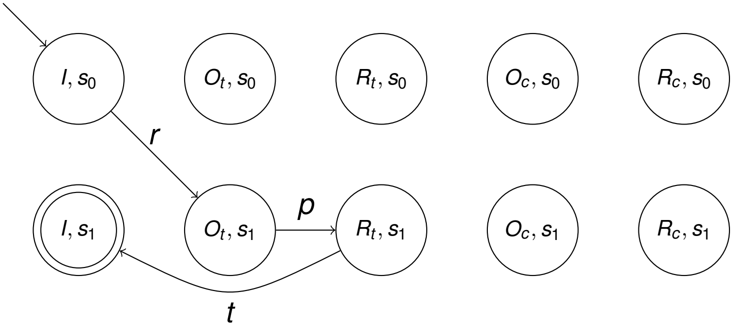
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



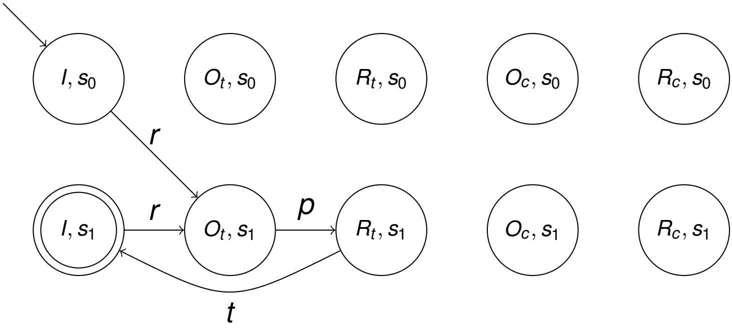
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



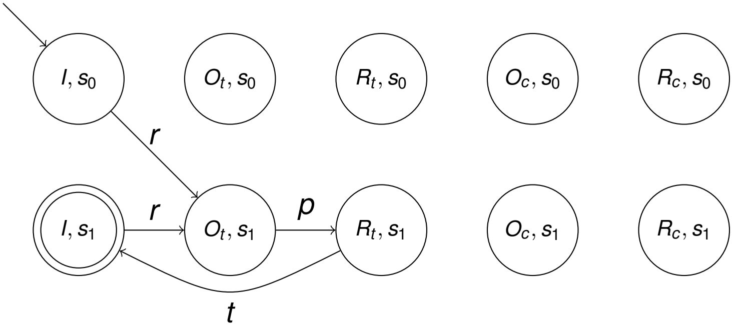
$M_{sys} \otimes \overline{M_{spec}}$

Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

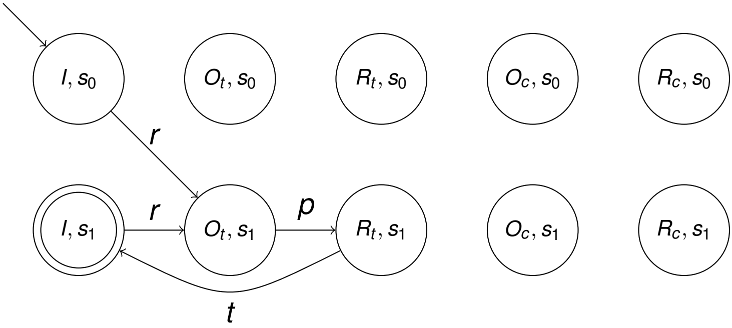
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- Acceptance cycle reachable from initial state

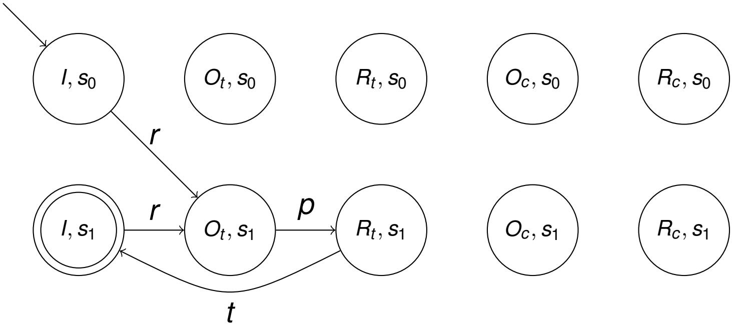
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- Acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) \neq \{\}$

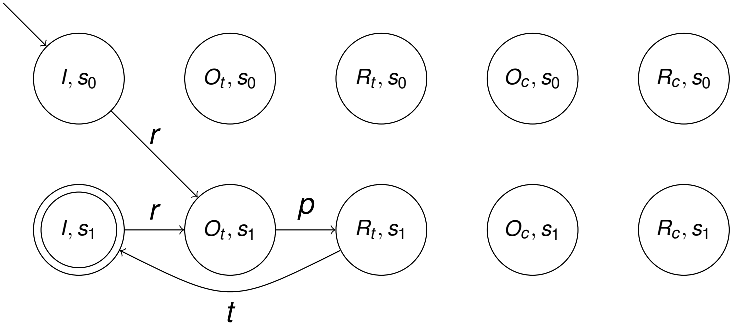
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- Acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) \neq \{\}$
- $L(M_{sys}) \not\subseteq L(M_{spec})$

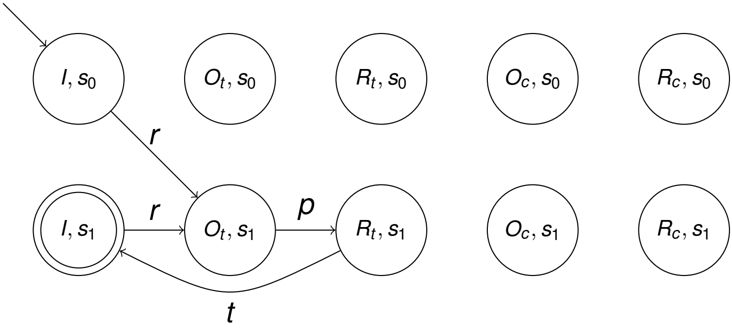
Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- Acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) \neq \{\}$
- $L(M_{sys}) \not\subseteq L(M_{spec})$
- Property is not satisfied

Checking the drinks machine



$M_{sys} \otimes \overline{M_{spec}}$

- Acceptance cycle reachable from initial state
- $L(M_{sys} \otimes \overline{M_{spec}}) \neq \{\}$
- $L(M_{sys}) \not\subseteq L(M_{spec})$
- Property is not satisfied
- Note that here only edges whose source state is reachable from the initial state are shown.

Summary

- We have introduced the theoretical principles of FSMs that allow us to check language inclusion ($L(M_1) \subseteq L(M_2)$)
- This allows us to check when one FSM, acting as a system model, satisfies a property, represented by another FSM
- This approach is applied in existing verification tools in industry
- Next we will see how to represent properties using formulas of temporal logic and how to translate such formulas into FSMs so that the verification approach that we have just seen can be applied.