Embedded Systems - #3- #4

Formal methods to the analysis and control of embedded systems

Maria Domenica Di Benedetto
Giordano Pola

Center of Excellence for Research DEWS
Dept of Electrical and Information Engineering
University of L'Aquila, Italy
mariadomenica.dibenedetto,giordano.pola@univaq.it

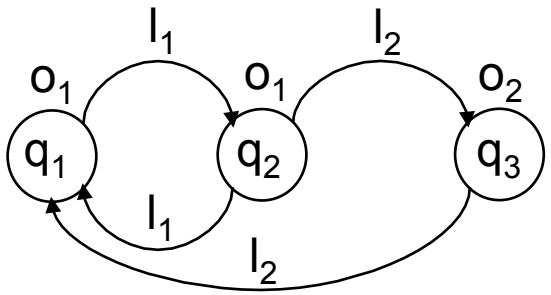
A unified framework

A Labelled Transition System (LTS) is a tuple:

$$T = (Q, L, \longrightarrow, O, H),$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation
- O output set
- $H: Q \rightarrow O$ output function



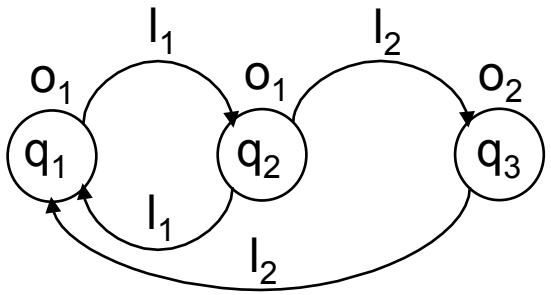
A unified framework

A Labelled Transition System (LTS) is a tuple:

$$T = (Q, L, \longrightarrow, O, H),$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation
- O output set
- $H: Q \to O$ output function



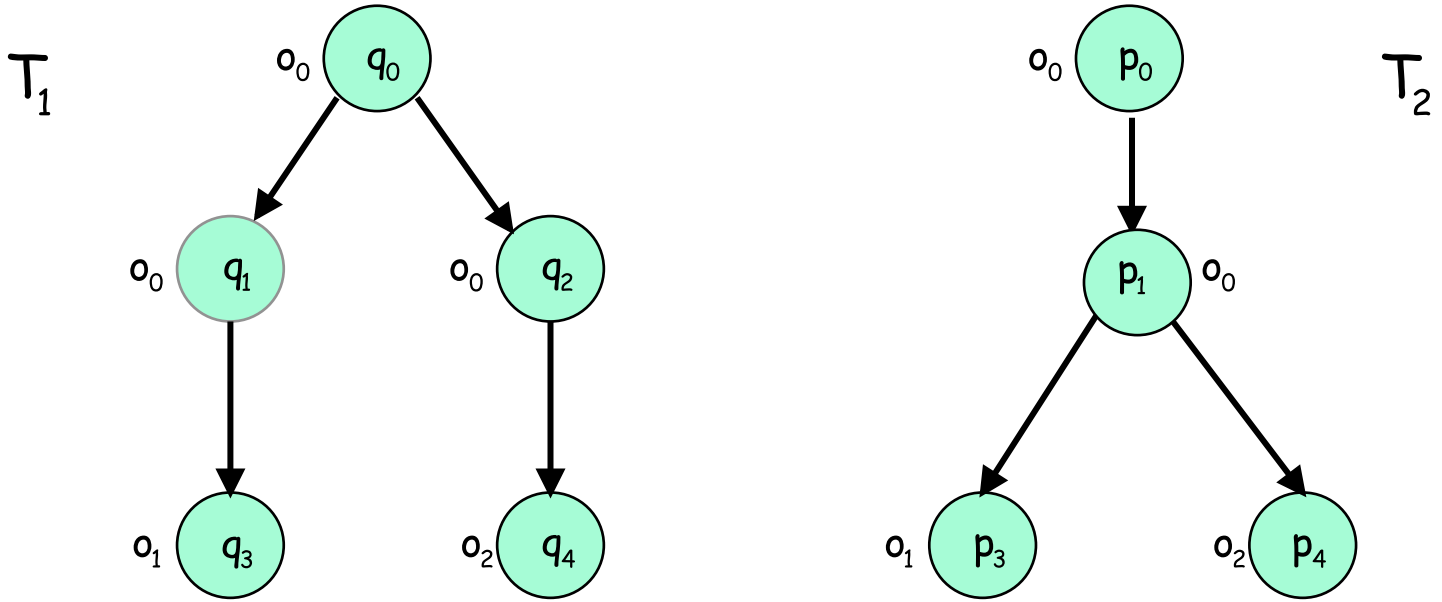
T is said:

- countable if Q and L are countable sets
- symbolic/finite if Q and L are finite sets
- deterministic if for q and l there exists at most one p so that $q \xrightarrow{l} p$
- nonblocking if for any q there exist at least one l and one p so that $q \xrightarrow{l} p$
- metric if O is a metric space

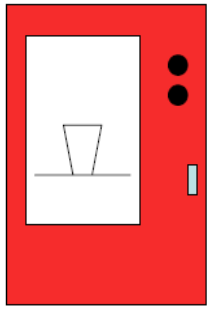
We will follow standard practice and denote $(q, u, p) \in \longrightarrow$ by $q \longrightarrow p$

A unified framework

Are T_1 and/or T_2 deterministic? Why?



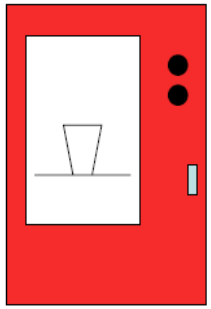
Vending machine



1. Insert coin(s)
2. Choose tea or coffee
3. Put the cup on the tray

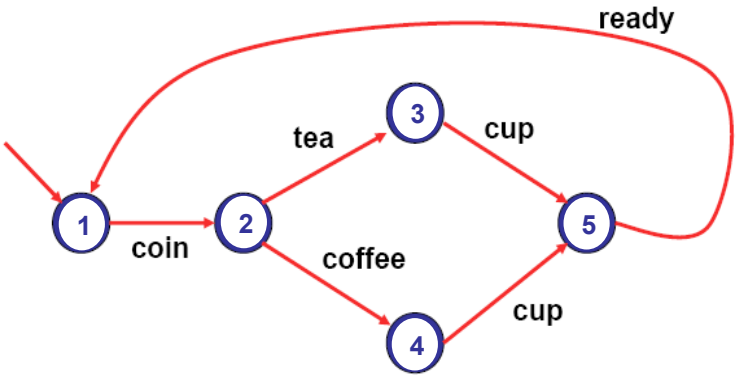
Vending machine

Vending machine

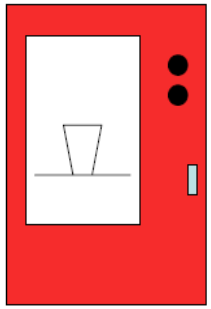


Vending machine

1. Insert coin(s)
2. Choose tea or coffee
3. Put the cup on the tray

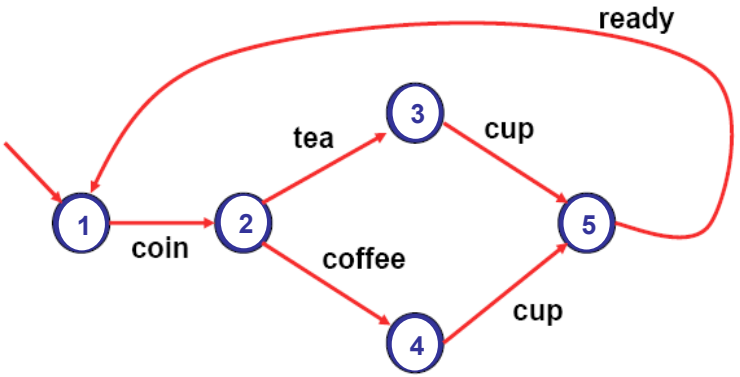


Vending machine



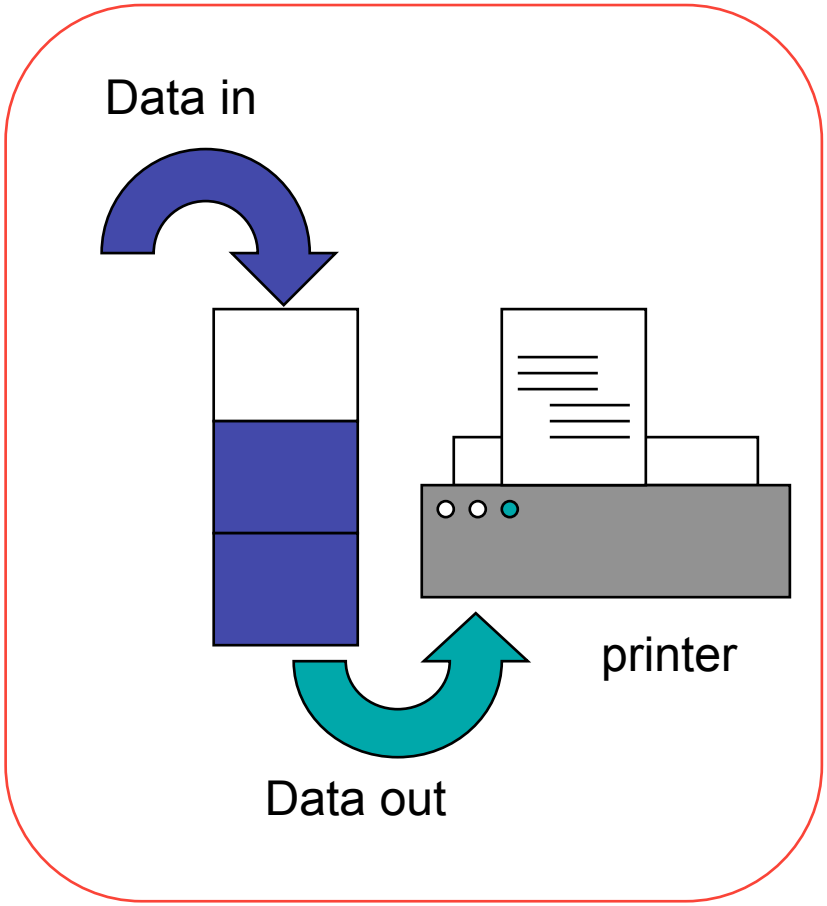
Vending machine

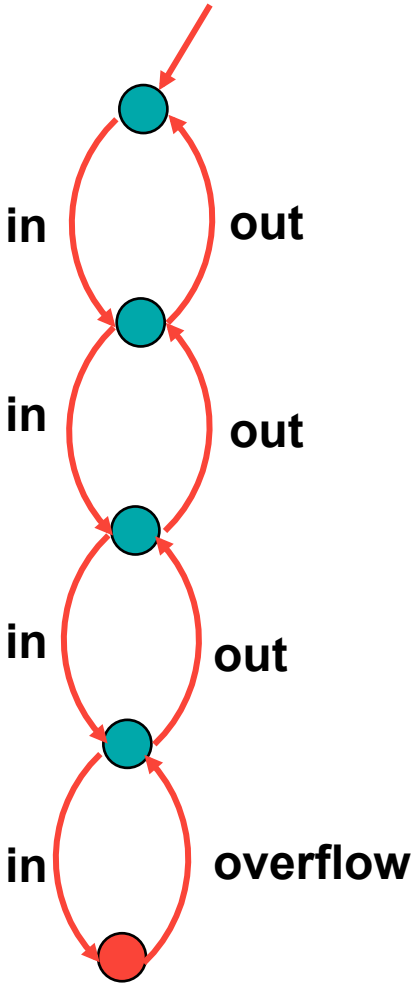
1. Insert coin(s)
2. Choose tea or coffee
3. Put the cup on the tray



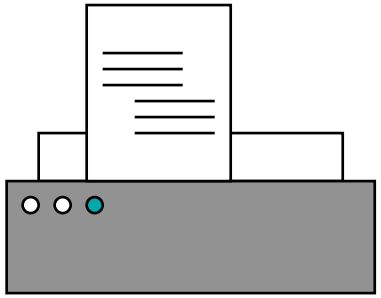
... Event driven versus time driven !

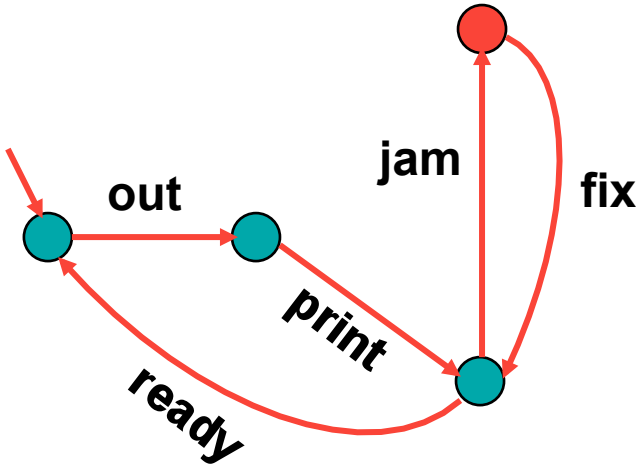
A printer data buffer





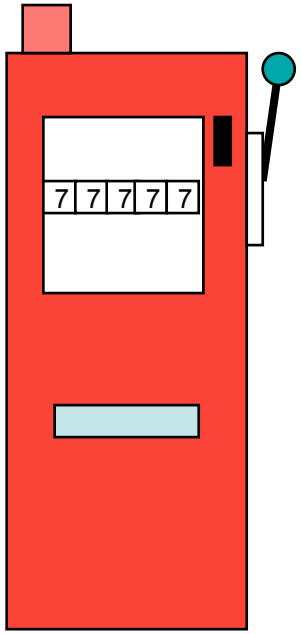
A printer



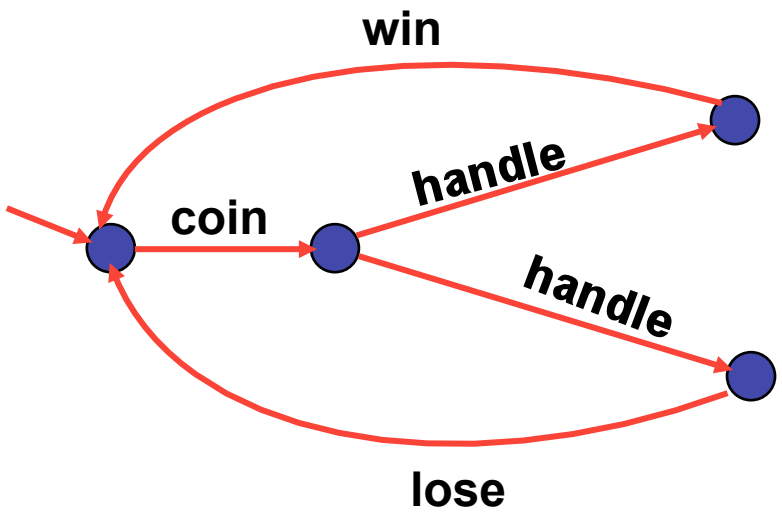


The printer receives data from the buffer, and print it out. Once the printout is ready, the printer is ready to receive new data. While printing, the paper can jam and need to be fixed before the printing process can resume.

A slot machine



1. Insert coin
2. Pull handle
3. Win if the combination is good, otherwise lose.



Modeling Recap

- Events are time-abstract.
- Just like modeling of continuous systems, the level of detail is ‘modeler dependent’.
- Compositionality is possible (to be discussed later).
- There can be non-determinism.

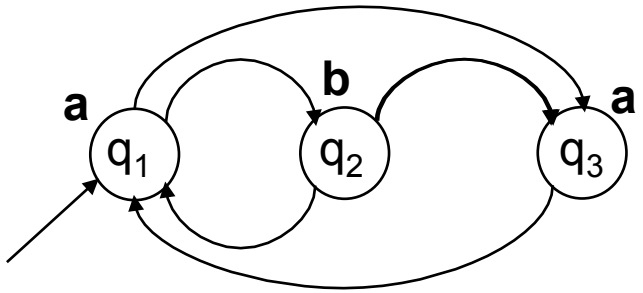
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is :
the corresponding output run of T is:

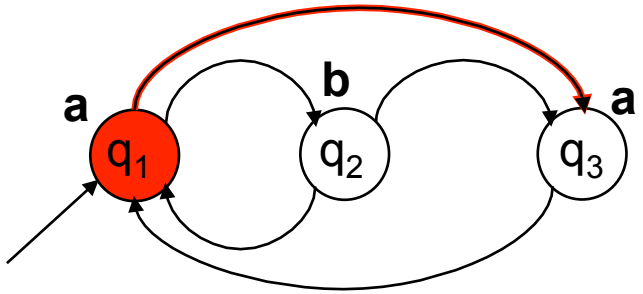
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : q_1
the corresponding output run of T is: a

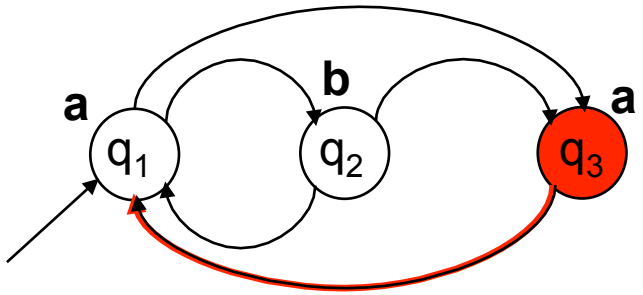
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : $q_1 q_3$
the corresponding output run of T is: a a

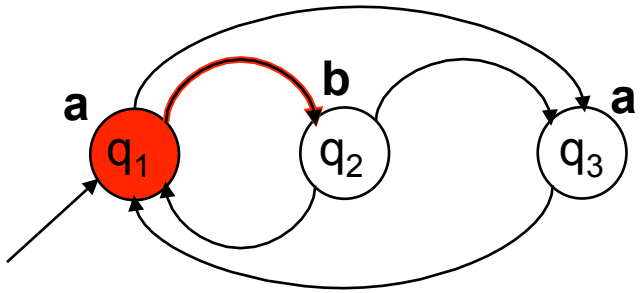
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : $q_1 q_3 q_1$
the corresponding output run of T is: a a a

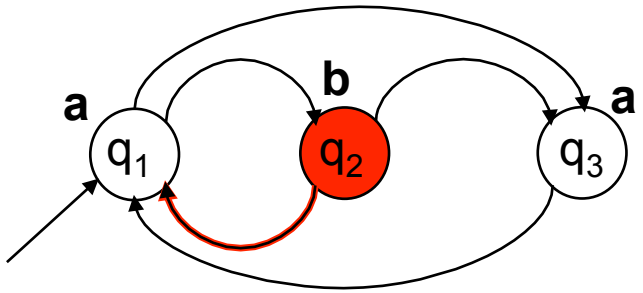
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : $q_1 q_3 q_1 q_2$
the corresponding output run of T is: a a a b

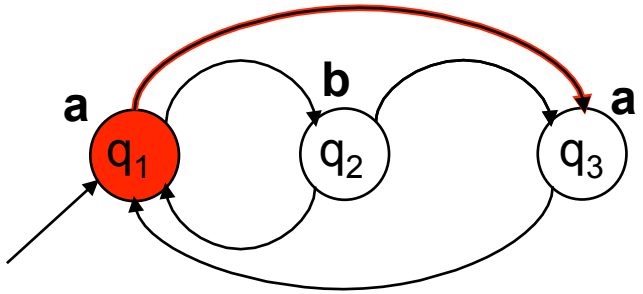
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : $q_1 q_3 q_1 q_2 q_1$
the corresponding output run of T is: a a a b a

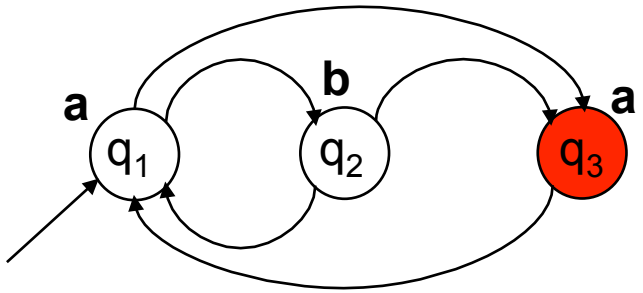
A unified framework

Definition A transition system T is a tuple

$$T = (Q, L, \longrightarrow, O, H)$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation;
- O output or observation set;
- $H: Q \rightarrow O$ output or observation map.



Starting from q_1 with observation a,

a possible run of T is : $q_1 q_3 q_1 q_2 q_1 q_3 \ldots$

the corresponding output run of T is: a a a b a a …

The language L(T) of T is the set of all output runs generated by T

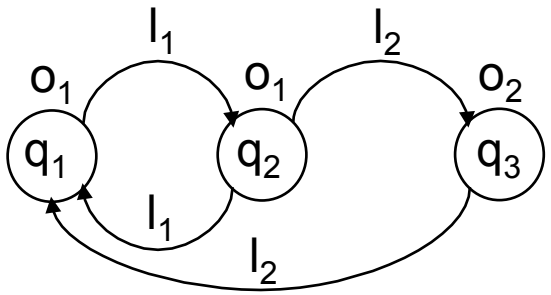
A unified framework: discrete processes

Definition A Labelled Transition System (LTS) is a tuple:

$$T = (Q, L, \longrightarrow, O, H),$$

where:

- Q set of states
- L set of labels
- $\longrightarrow \subseteq Q \times L \times Q$ transition relation
- O output set
- $H: Q \rightarrow O$ output function



We can formally model software as a LTS. The states are all the possible memory configurations and the transition relation describes how the memory contents are changed by the execution of instructions.

Example: Software

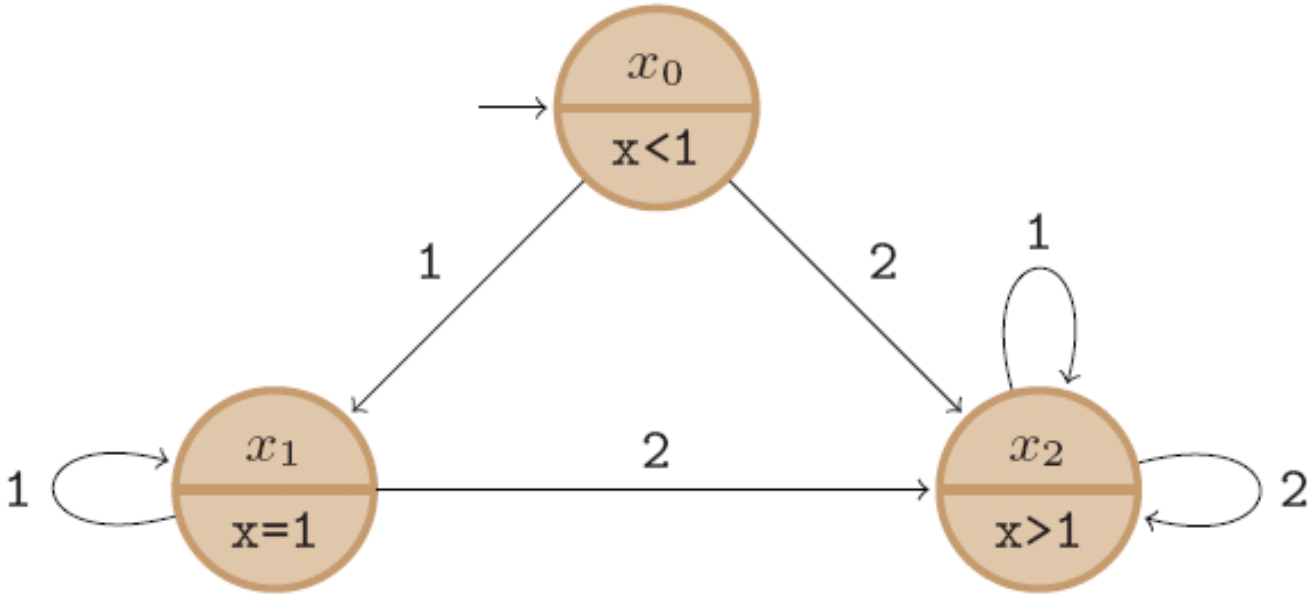
Suppose that we want to compute the average of a stream of numbers but we do not know a priori the length of the stream. One possible way to compute the average is to update the average upon the reception of a new number in the stream.

$x{:=}0;$
$n{:=}0;$
while *true* **do**
  $y{:=}\text{read}(\text{input});$
  $x := x\frac{n}{n+1} + y\frac{1}{n+1};$
  $n := n + 1;$
end

Taken from Tabuada, Verification and Control of Hybrid Systems, Springer Verlag, 2009.

Example: Software

Assume that we are interested in knowing if x is smaller, equal, or greater than 1 when y is restricted to assume values in the set {1,2}. One possible finite-state model capturing the dynamics of x is represented in the following:



Taken from Tabuada, Verification and Control of Hybrid Systems, Springer Verlag, 2009.

A unified framework: continuous processes

Given a control system Σ:

$$\dot{x} = f(x,u), \qquad x \in R^n,\ u \in R^m$$

we can define the following LTS

$$T(\Sigma) = (Q, L, \longrightarrow, O, H),$$

where:

- $Q = R^n$
- L is the collection of control signals $u : R \to R^m$
- $p \xrightarrow{u} q$, if $x(\tau,p,u) = q$ for some $\tau \geq 0$
- $O = R^n$
- H is the identity function

$T(\Sigma)$ captures all information contained in Σ

… by using similar arguments I can associate a LTS to a hybrid system!

A unified framework: continuous processes

Given a control system Σ:

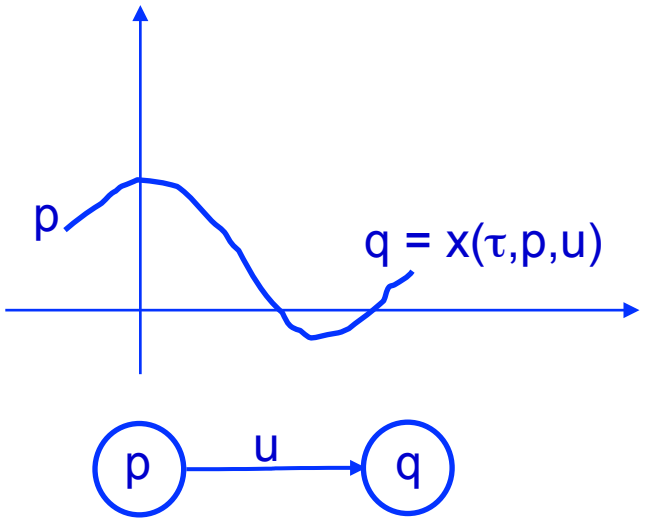
$$\dot{x} = f(x,u), \qquad x \in R^n,\ u \in R^m$$

we can define the following LTS

$$T(\Sigma) = (Q, L, \longrightarrow, O, H),$$

where:

- $Q = R^n$
- L is the collection of control signals $u : R \to R^m$
- $p \xrightarrow{u} q$, if $x(\tau,p,u) = q$ for some $\tau \geq 0$
- $O = R^n$
- H is the identity function



$T(\Sigma)$ captures all information contained in Σ

… by using similar arguments an LTS can be associated to a hybrid system!

A unified framework

Given a control system Σ:

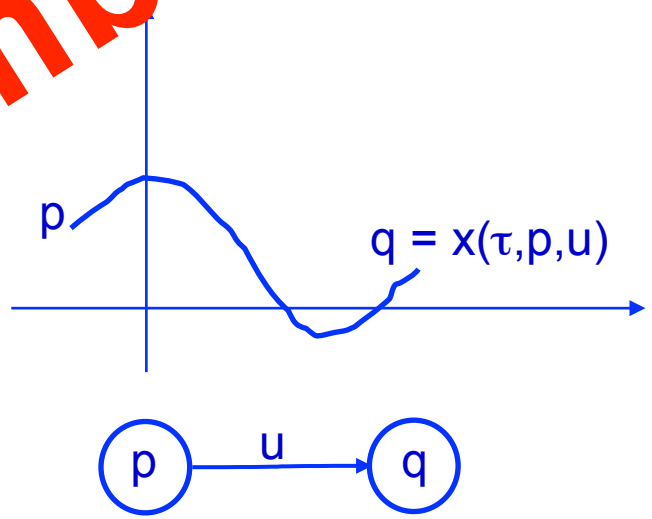
$$\dot{x} = f(x,u), \qquad x \in R^n,\ u \in R^m$$

we can define the following LTS

$$T(\Sigma) = (Q, L, \longrightarrow, O, H),$$

where:

- $Q = R^n$
- L is the collection of control signals $u: R \to R^m$
- $p \longrightarrow q$, if $x(\tau,p,u) = q$ for some $\tau \geq 0$
- $O = R^n$
- H is the identity function



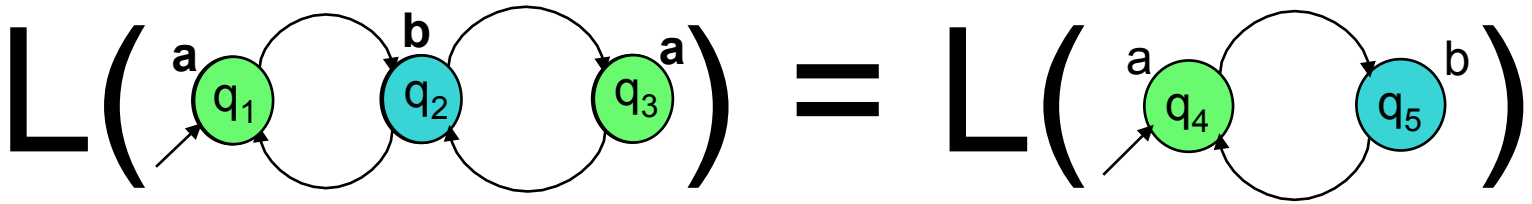
$T(\Sigma)$ captures all information contained in Σ

… by using similar arguments I can associate a LTS to a hybrid system!

T(Σ) is not symbolic !

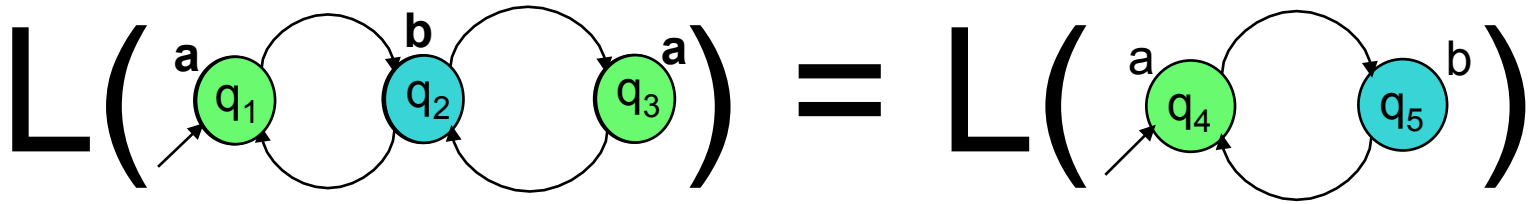
Language equivalence

Definition T_1 and T_2 are language equivalent if $L(T_1) = L(T_2)$



Language equivalence

Definition T_1 and T_2 are language equivalent if $L(T_1) = L(T_2)$



Language equivalence is an equivalence relation on the space of transition systems, i.e. :

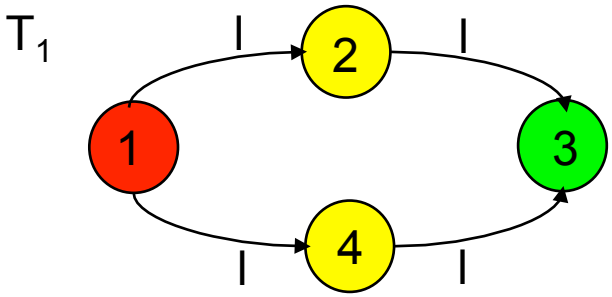
- (reflexivity) $L(T_1) = L(T_1)$
- (symmetry) $L(T_1) = L(T_2) \Rightarrow L(T_2) = L(T_1)$
- (transitivity) $L(T_1) = L(T_2) \wedge L(T_2) = L(T_3) \Rightarrow L(T_1) = L(T_3)$

Bisimulation equivalence

R. Milner (1989), Communication and Concurrency. Prentice Hall

D.M.R. Park (1981), Concurrency and automata on infinite sequences. LNCS, vol. 104

... the intuitive idea!

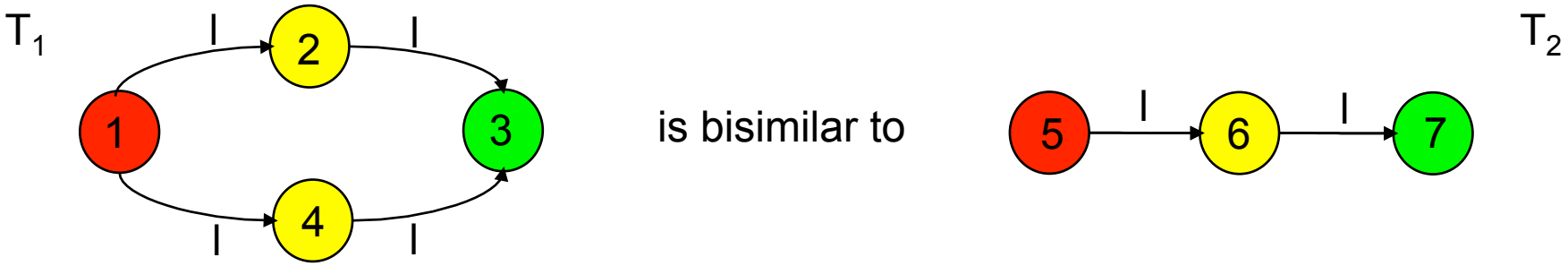


Bisimulation equivalence

R. Milner (1989), Communication and Concurrency. Prentice Hall

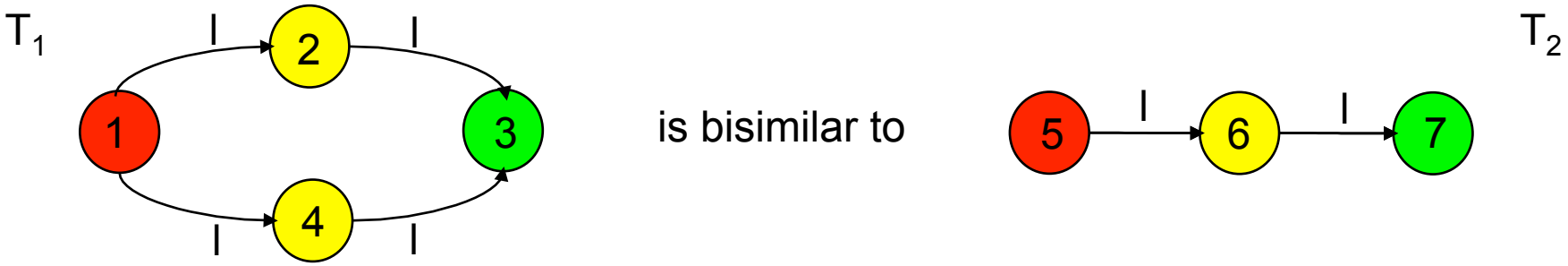
D.M.R. Park (1981), Concurrency and automata on infinite sequences. LNCS, vol. 104

… the intuitive idea!



Bisimulation equivalence

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $H_1(p_1) = H_2(p_2)$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $H_1(p_1) = H_2(p_2)$



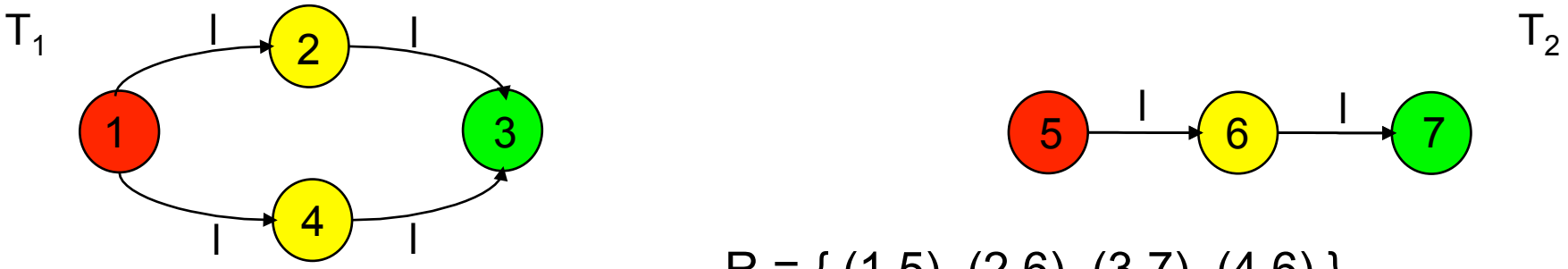
Bisimulation equivalence

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$



R = { (1,5), (2,6), (3,7), (4,6) }

Bisimulation equivalence

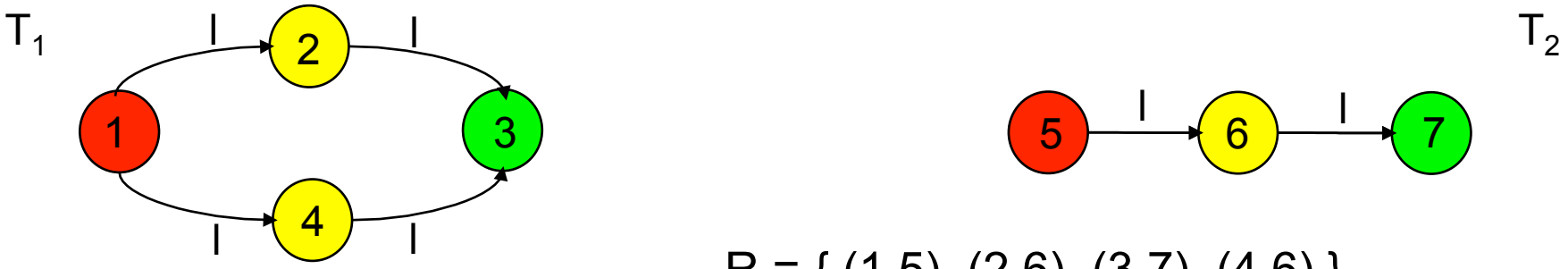
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$



$R = \{ (1,5), (2,6), (3,7), (4,6) \}$

Bisimulation equivalence

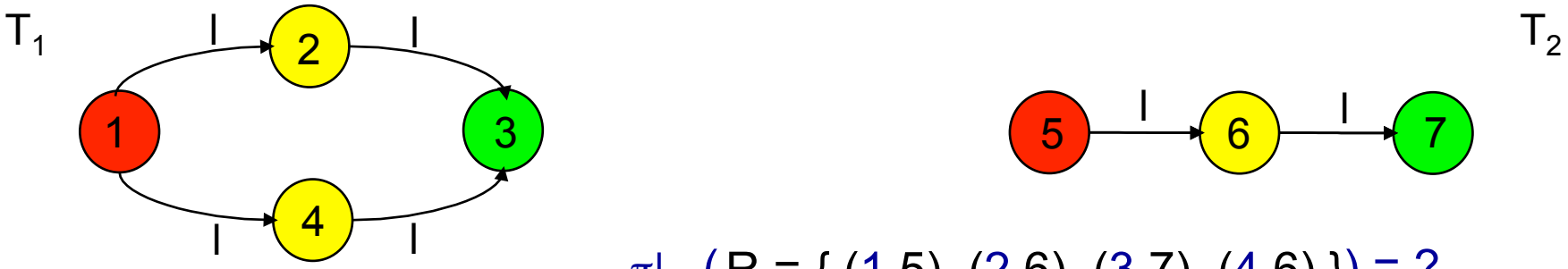
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$



$\pi|_{Q_1}(R = \{ (1,5), (2,6), (3,7), (4,6) \}) = ?$

Bisimulation equivalence

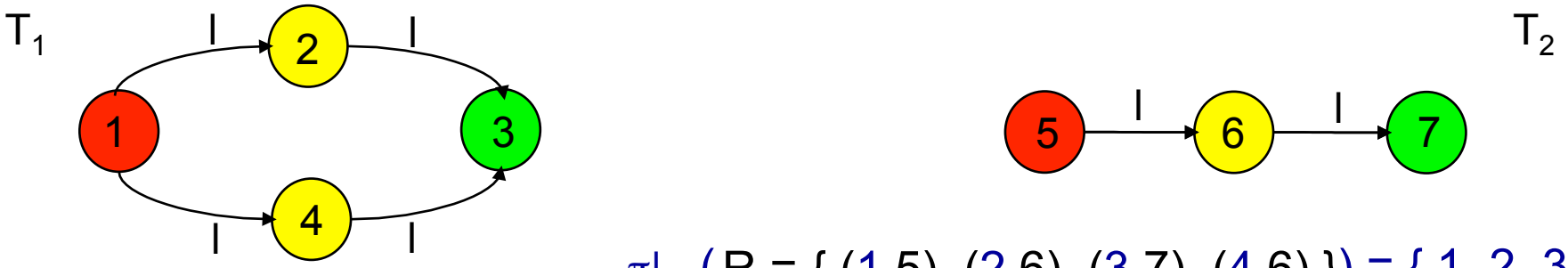
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$



$\pi|_{Q_1}(R = \{ (1,5), (2,6), (3,7), (4,6) \}) = \{ 1, 2, 3, 4 \}$

Bisimulation equivalence

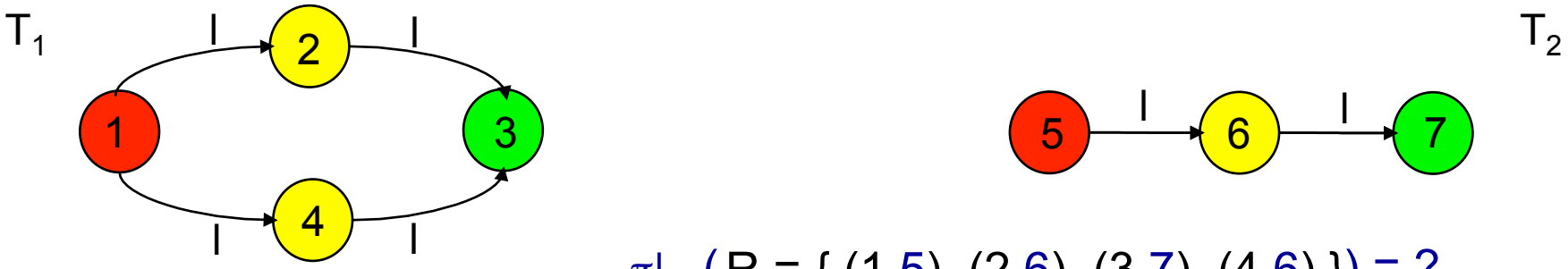
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$



$\pi|_{Q_2}(R = \{ (1,5), (2,6), (3,7), (4,6) \}) = ?$

Bisimulation equivalence

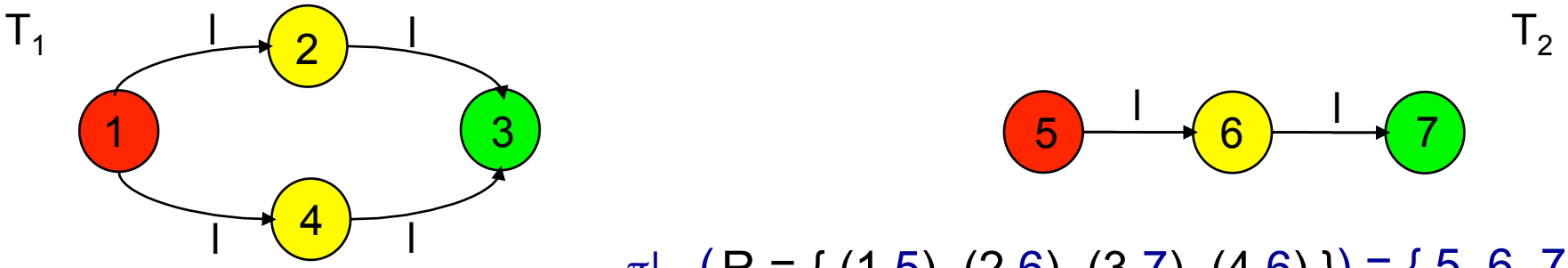
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$



$\pi|_{Q_2}(R = \{ (1,5), (2,6), (3,7), (4,6) \}) = \{ 5, 6, 7 \}$

Bisimulation equivalence

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$

Bisimulation equivalence
preserves
most of the dynamical properties
of interest!

Bisimulation equivalence

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$

Bisimulation equivalence is an equivalence relation on the space of transition systems, i.e. :

- (reflexivity) $T_1 \sim T_1$
- (symmetry) $T_1 \sim T_2 \Rightarrow T_2 \sim T_1$
- (transitivity) $T_1 \sim T_2 \wedge T_2 \sim T_3 \Rightarrow T_1 \sim T_3$

Bisimulation equivalence

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$

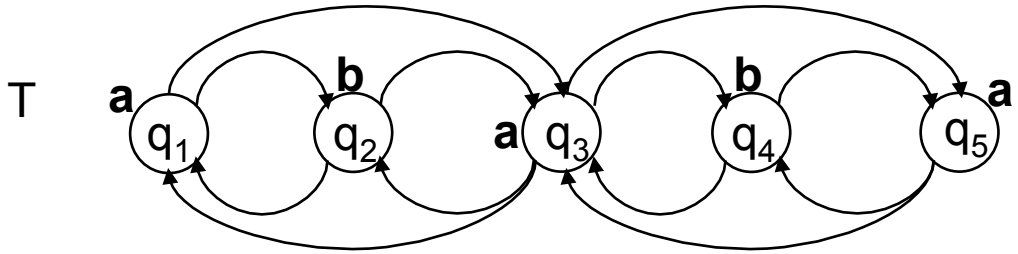
Bisimulation equivalence is an equivalence relation on the space of transition systems, i.e. :

- (reflexivity) $T_1 \sim T_1$
- (symmetry) $T_1 \sim T_2 \Rightarrow T_2 \sim T_1$
- (transitivity) $T_1 \sim T_2 \wedge T_2 \sim T_3 \Rightarrow T_1 \sim T_3$

... what is R in the three cases?

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

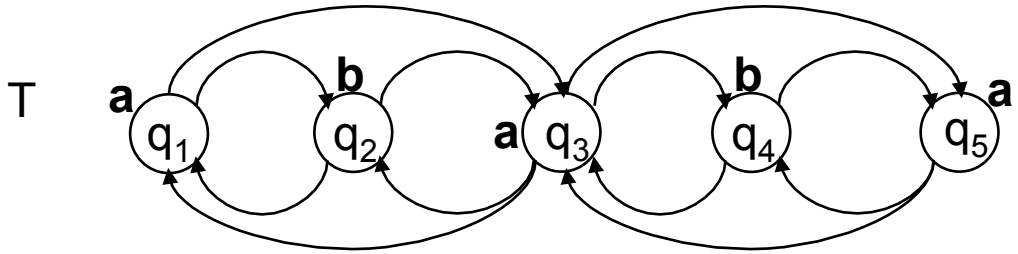
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R^*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T^* of T induced by R^*, i.e.
- T^* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

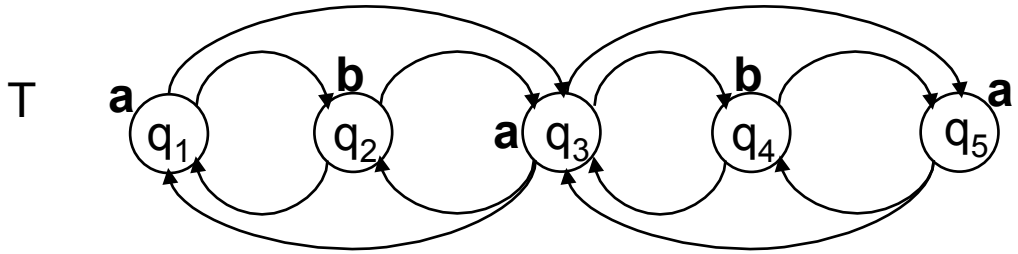
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

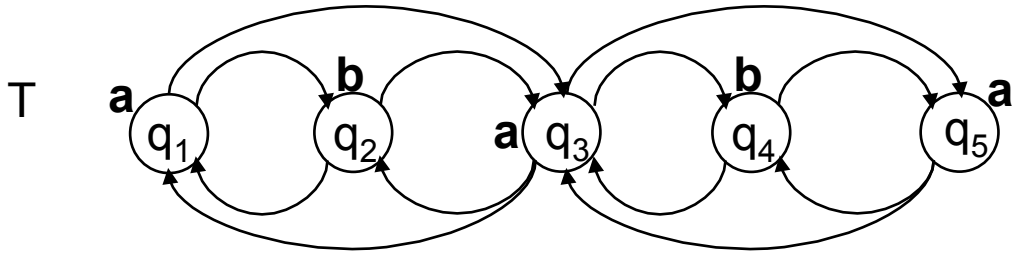
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

- Maximal bisimulation relation is an equivalence relation on Q,
i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$

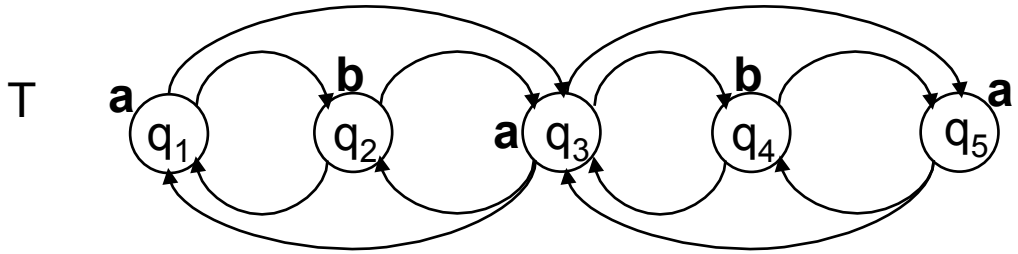
- Partition the state space of T as $Q = C_1 \cup C_2$

- Construct the quotient T* of T induced by R*, i.e.

- T* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

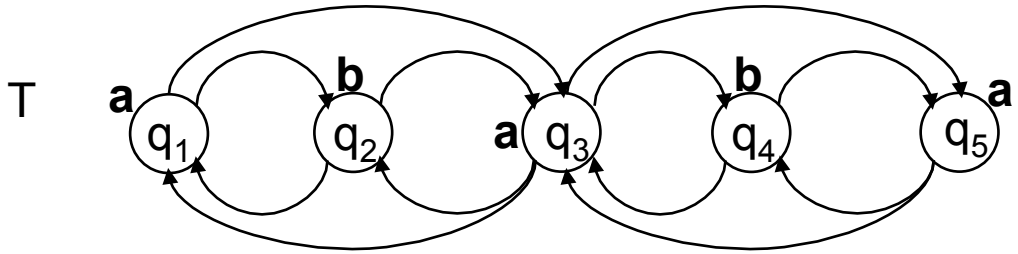
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive (not all bisimulation relations are so!)

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

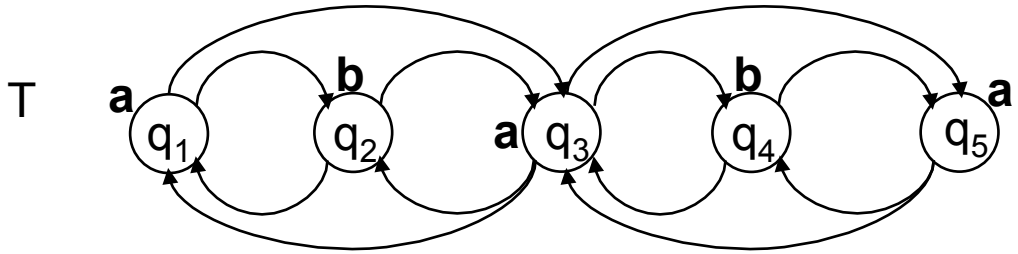
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

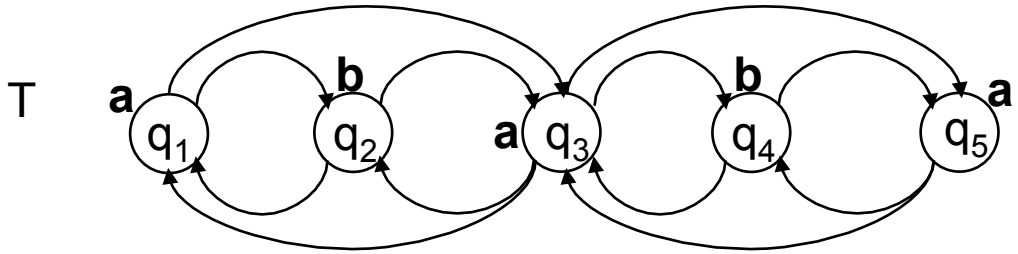
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R^*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T^* of T induced by R^*, i.e.
- T^* is minimal!

Bisimulation equivalence

Bisimulation as a tool for reduction:



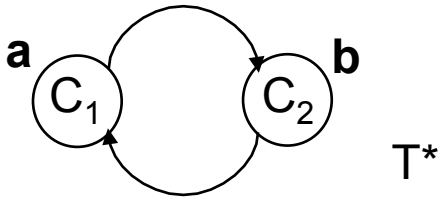
- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

- Maximal bisimulation relation is an equivalence relation on Q,

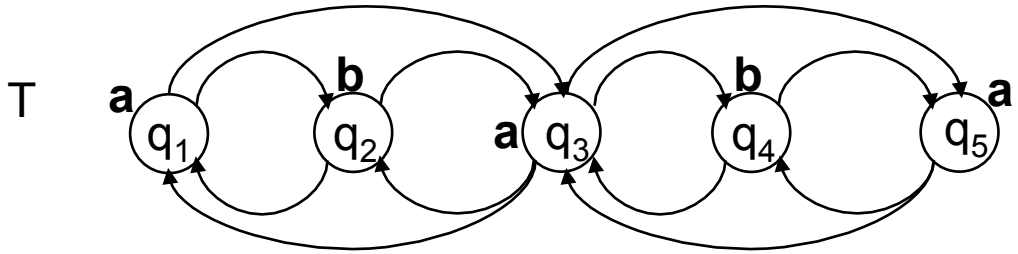
i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!



Bisimulation equivalence

Bisimulation as a tool for reduction:



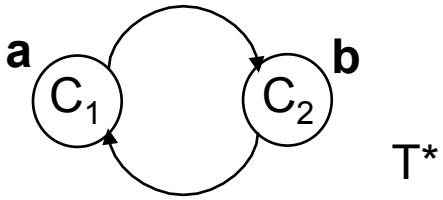
- Maximal bisimulation relation between T and T:

$R^* = \{(q_1,q_1), (q_2,q_2), (q_3,q_3), (q_4,q_4), (q_5,q_5), (q_1,q_3), (q_3,q_1), (q_1,q_5), (q_5,q_1), (q_3,q_5), (q_5,q_3), (q_2,q_4), (q_4,q_2)\}$

- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!



Bisimulation equivalence

Bisimulation as a tool for reduction:

The core problem is the computation of R*? How?

Bisimulation equivalence

Bisimulation as a tool for reduction:

The core problem is the computation of R*? How?

$$B(0) = \{ (q,p) \in Q \times Q \text{ s.t. } H(q) = H(p) \}$$

$$B(k+1) = \{ (q,p) \in Q \times Q \text{ s.t.}$$

$$\forall\, q \xrightarrow{l} q' \; \exists\, p \xrightarrow{l'} p' \text{ s.t. } (q',p') \in B(k) \wedge$$

$$\forall\, p \xrightarrow{l} p' \; \exists\, q \xrightarrow{l'} q' \text{ s.t. } (q',p') \in B(k) \}$$

Bisimulation equivalence

Bisimulation as a tool for reduction:

The core problem is the computation of R*? How?

$$B(0) = \{ (q,p) \in Q \times Q \text{ s.t. } H(q) = H(p) \}$$

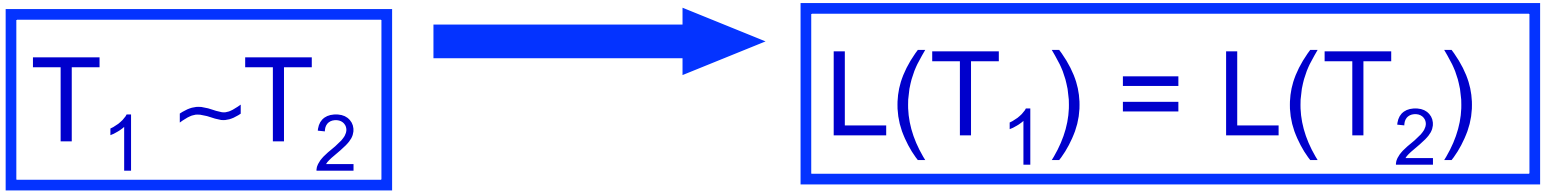
$$B(k+1) = \{ (q,p) \in Q \times Q \text{ s.t.}$$
$$\forall\, q \xrightarrow{l} q' \; \exists\, p \xrightarrow{l'} p' \text{ s.t. } (q',p') \in B(k) \wedge$$
$$\forall\, p \xrightarrow{l} p' \; \exists\, q \xrightarrow{l'} q' \text{ s.t. } (q',p') \in B(k) \}$$

If $\exists\, k^*$ s.t. $B(k^*) = B(k^*+1)$ then $R^* = B(k^*)$

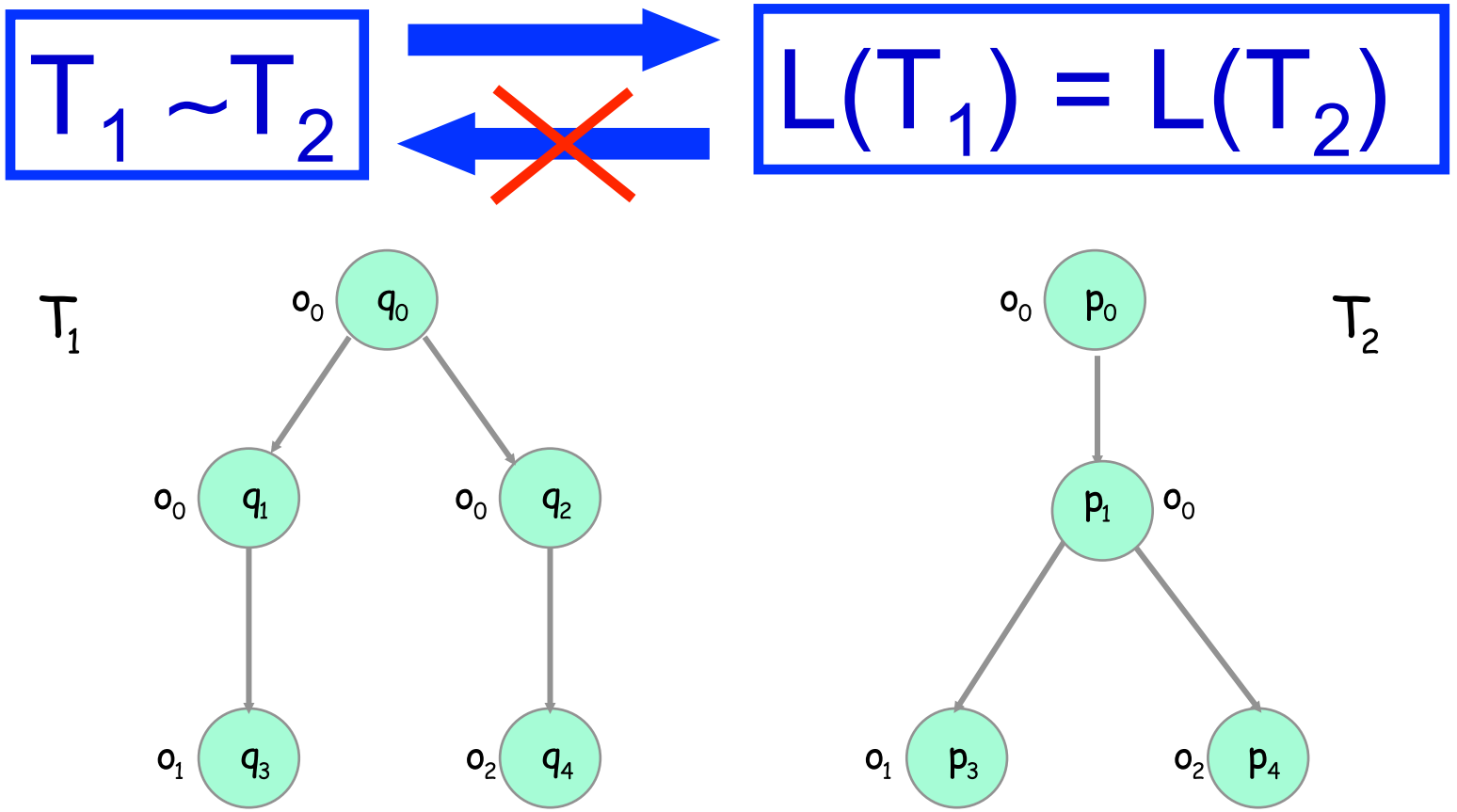
If Q is finite, termination of the algorithm is guaranteed in polynomial time!

If Q is infinite, (as in the case of control systems) …

Language and bisimulation equivalence



Language and bisimulation equivalence



Bisimulation equivalence

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***bisimulation relation*** between T_1 and T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \sim T_2$, if $\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$

Bisimulation equivalence
preserves
most of the dynamical properties
of interest!

Abstraction and simulation

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$R \subseteq Q_1 \times Q_2$

simulation ***from*** ***to***

is a ~~***bisimulation relation***~~ ~~between~~ T_1 ~~and~~ T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$
- ~~$q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 so that $(p_1, p_2) \in R$~~

LTSs T_1 ~~and T_2 are ***bisimilar***~~, denoted ~~$T_1 \sim T_2$~~, if ~~$\pi|_{Q_1}(R) = Q_1$ and $\pi|_{Q_2}(R) = Q_2$~~

is ***simulated*** from T_2 $\quad T_1 \leq T_2$

Abstraction and simulation

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***simulation relation*** from T_1 to T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$

LTSs T_1 is ***simulated*** from T_2 , denoted $T_1 \leq T_2$, if $\pi|_{Q_1}(R) = Q_1$

Abstraction and simulation

Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

is a ***simulation relation*** from T_1 to T_2 if for all $(q_1, q_2) \in R$

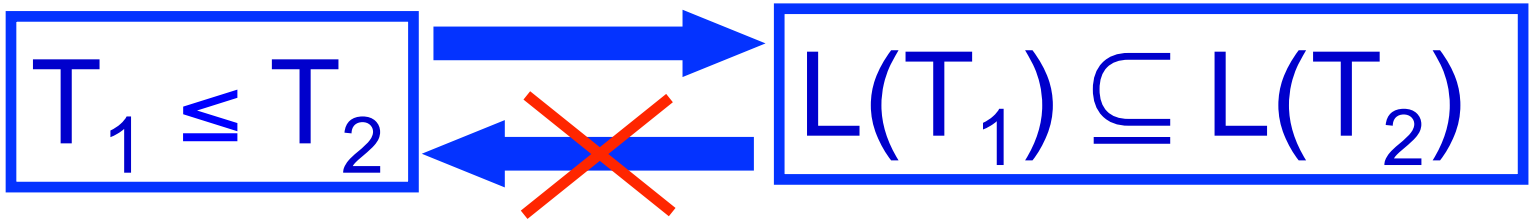
- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 so that $(p_1, p_2) \in R$

LTSs T_1 is ***simulated*** from T_2 , denoted $T_1 \leq T_2$, if $\pi|_{Q_1}(R) = Q_1$

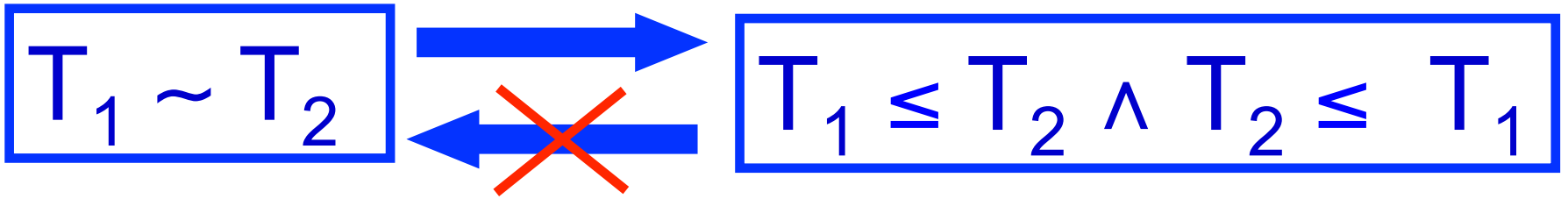
Simulation relation is not equivalence relation
on the space of transition systems, but :

- (reflexivity) $T_1 \leq T_1$
- (symmetry) $T_1 \leq T_2 \not\Rightarrow T_2 \leq T_1$
- (transitivity) $T_1 \leq T_2 \wedge T_2 \leq T_3 \Rightarrow T_1 \leq T_3$

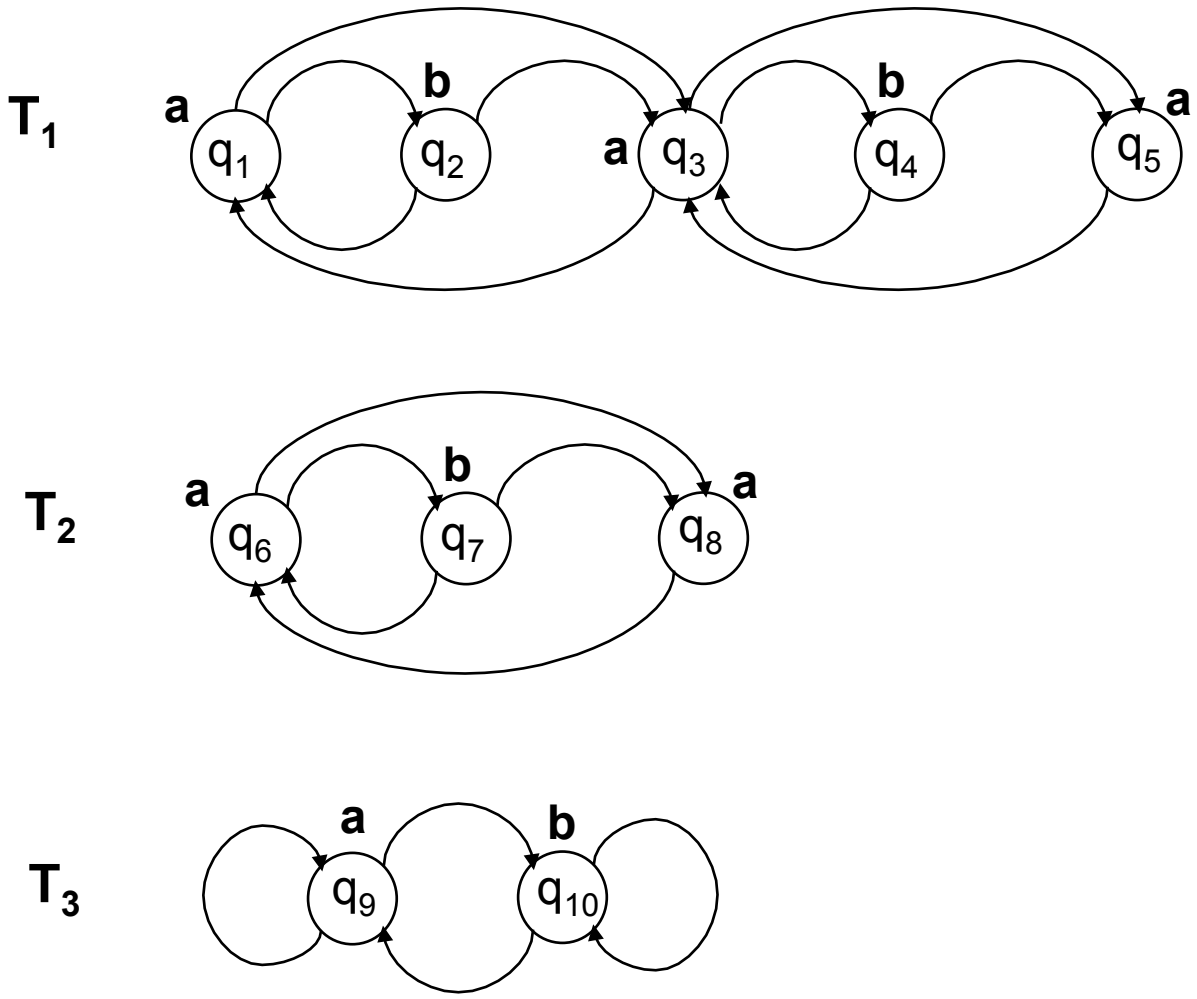
Language inclusion and simulation relation



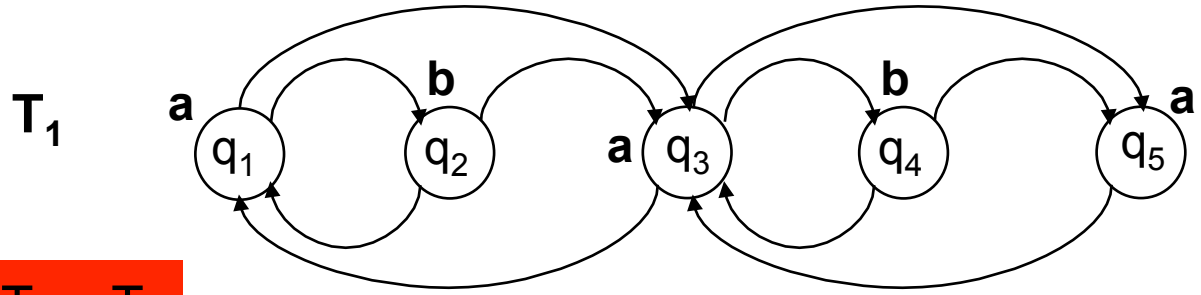
Simulation relation and bisimulation equivalence



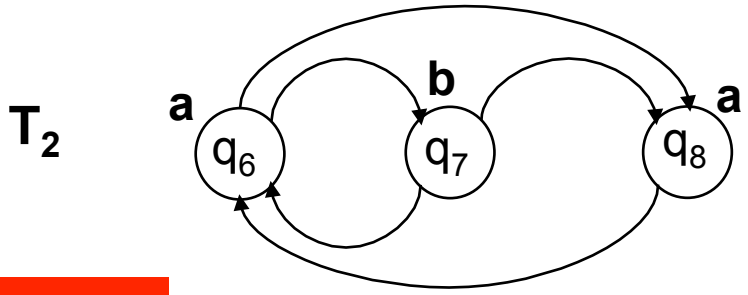
Abstraction and simulation



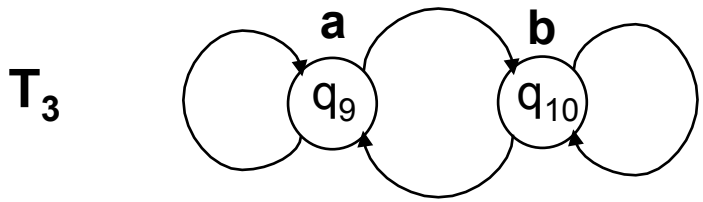
Abstraction and simulation



$T_1 \leq T_2$

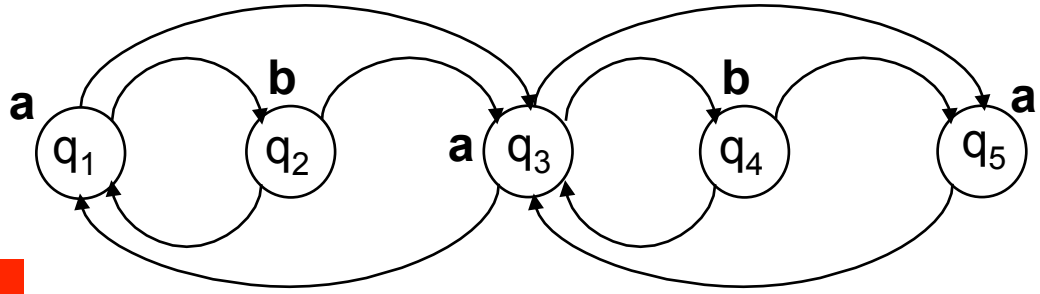


$T_2 \leq T_3$



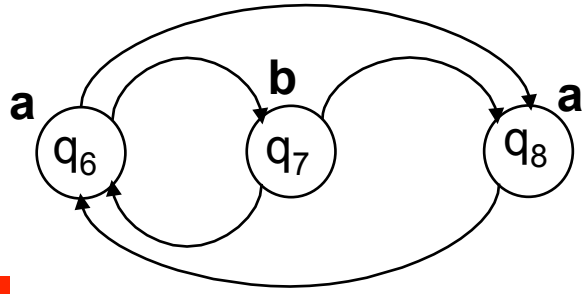
Abstraction and simulation

T_1



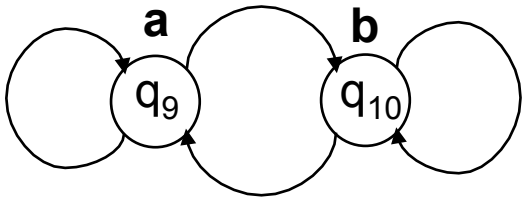
$T_1 \leq T_2$

T_2



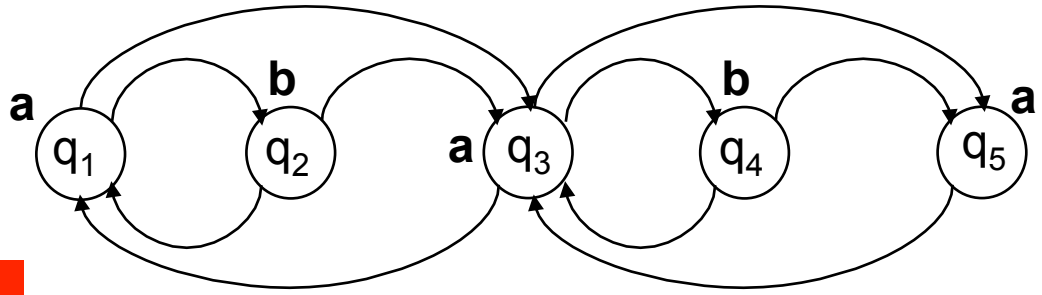
$T_2 \leq T_3$

T_3

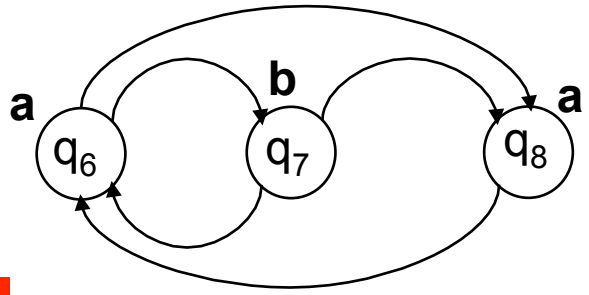


... who is R in the two cases?

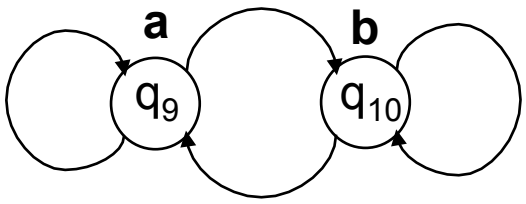
Abstraction and simulation



$T_1 \leq T_2$



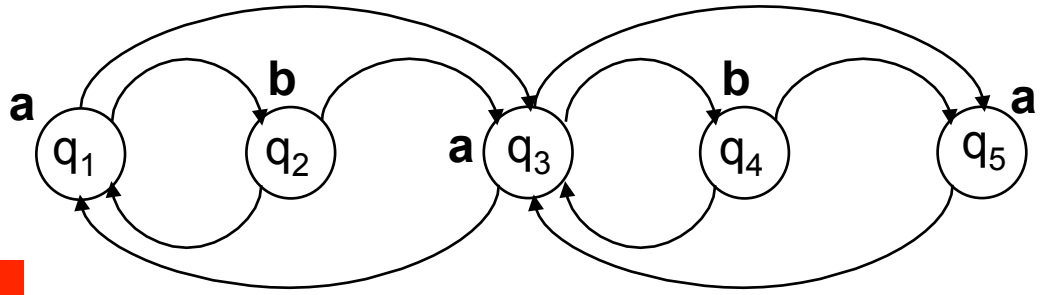
$T_2 \leq T_3$



Asbstraction

Refinement

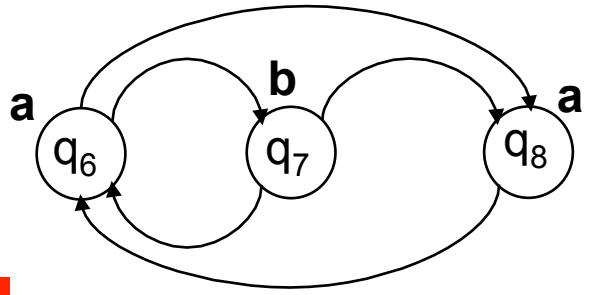
Abstraction and simulation



T_1

5 states

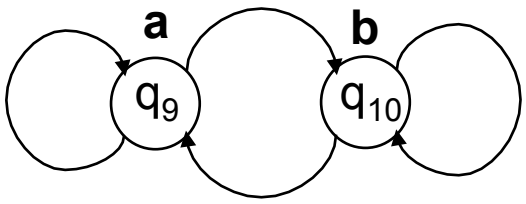
$T_1 \leq T_2$



T_2

3 states

$T_2 \leq T_3$



T_3

2 states

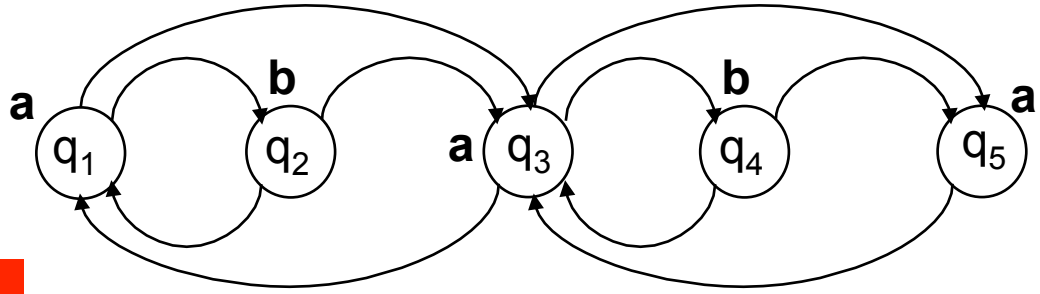
Asbstraction

Refinement

Abstraction and simulation

T_0 … infinite number of states…

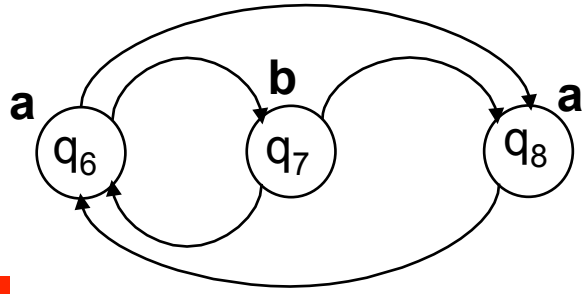
T_1



5 states

$T_1 \leq T_2$

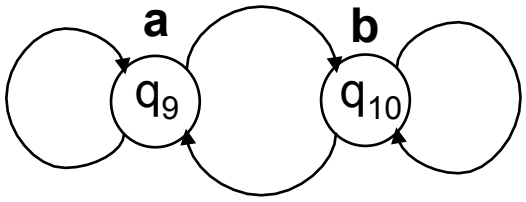
T_2



3 states

$T_2 \leq T_3$

T_3



2 states

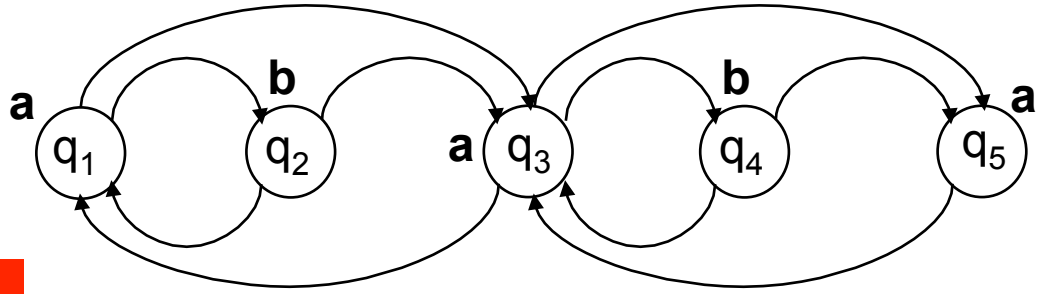
Asbstraction

Refinement

Abstraction and simulation

T_0 … infinite number of states…

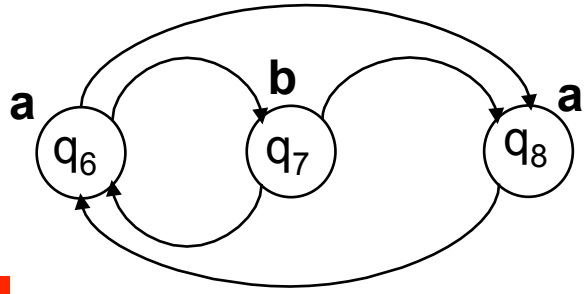
T_1



5 states

$T_1 \leq T_2$

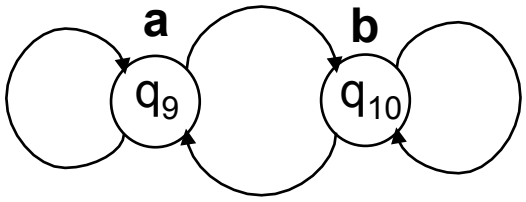
T_2



3 states

$T_2 \leq T_3$

T_3

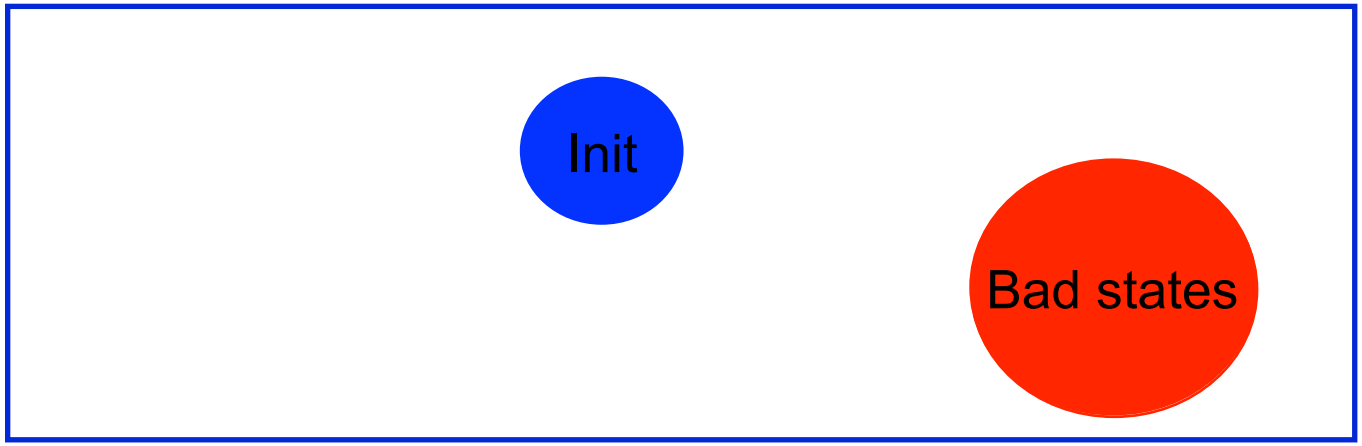


2 states

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

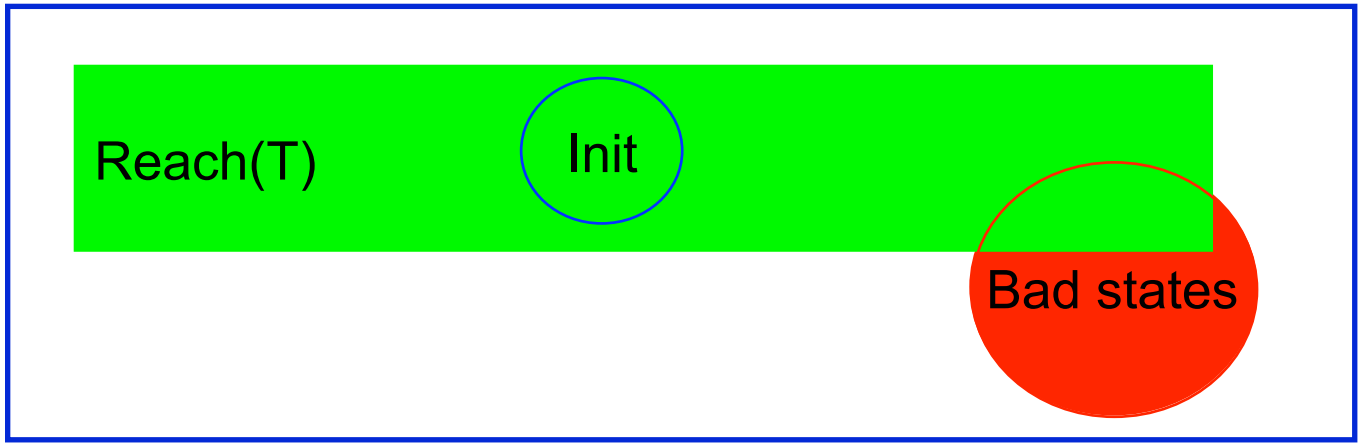


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

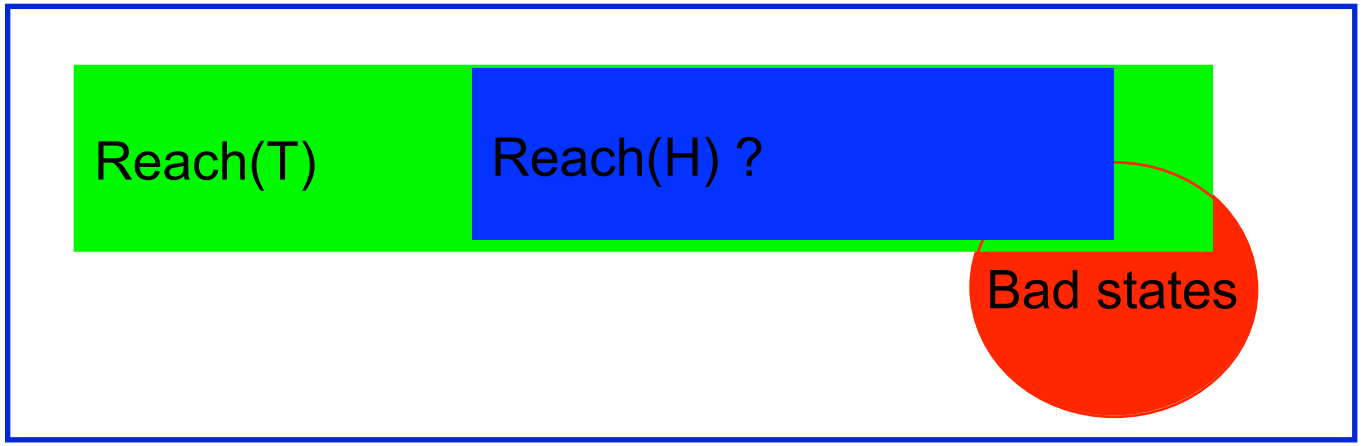


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

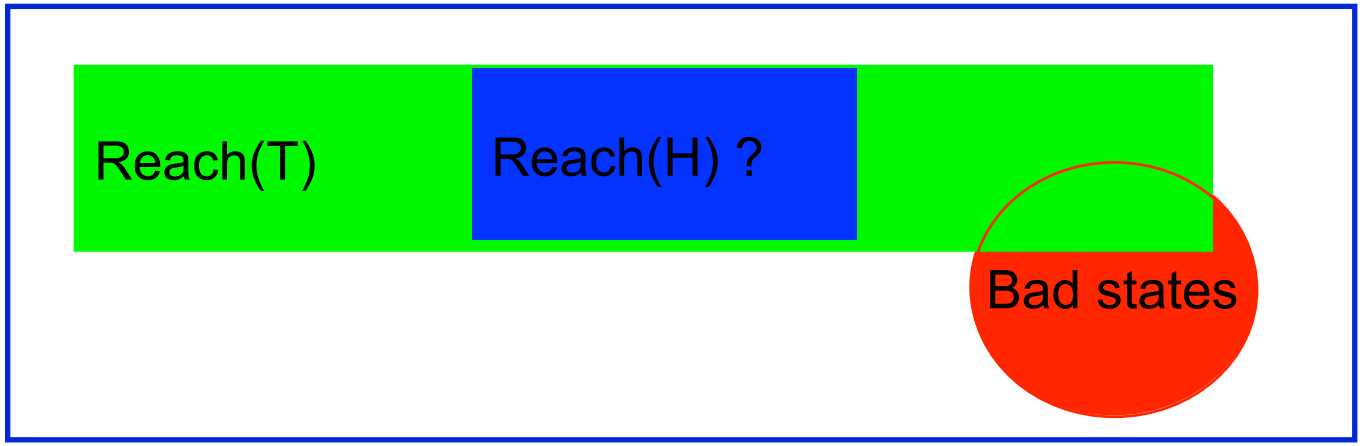


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

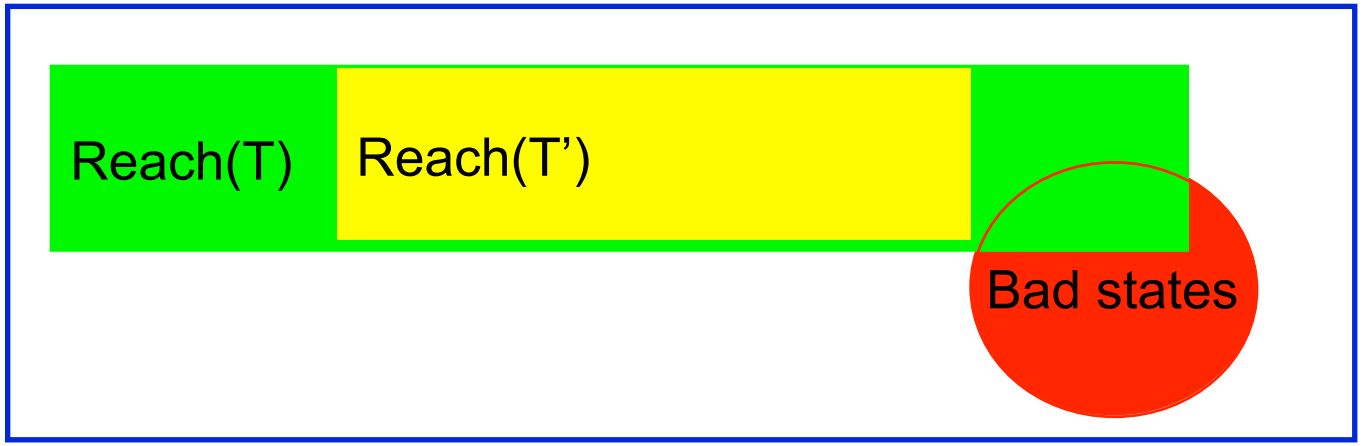


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?



- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

Why abstraction?

Abstraction and simulation

Let us consider a Hybrid system H and let us consider the problem of checking whether or not the output of H avoids a region (of bad states) starting from Init?

- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!