Embedded Systems
2012/13

Basilica di Santa Maria di Collemaggio, 1287, L'Aquila

Lecture 3
Introduction to Formal Methods

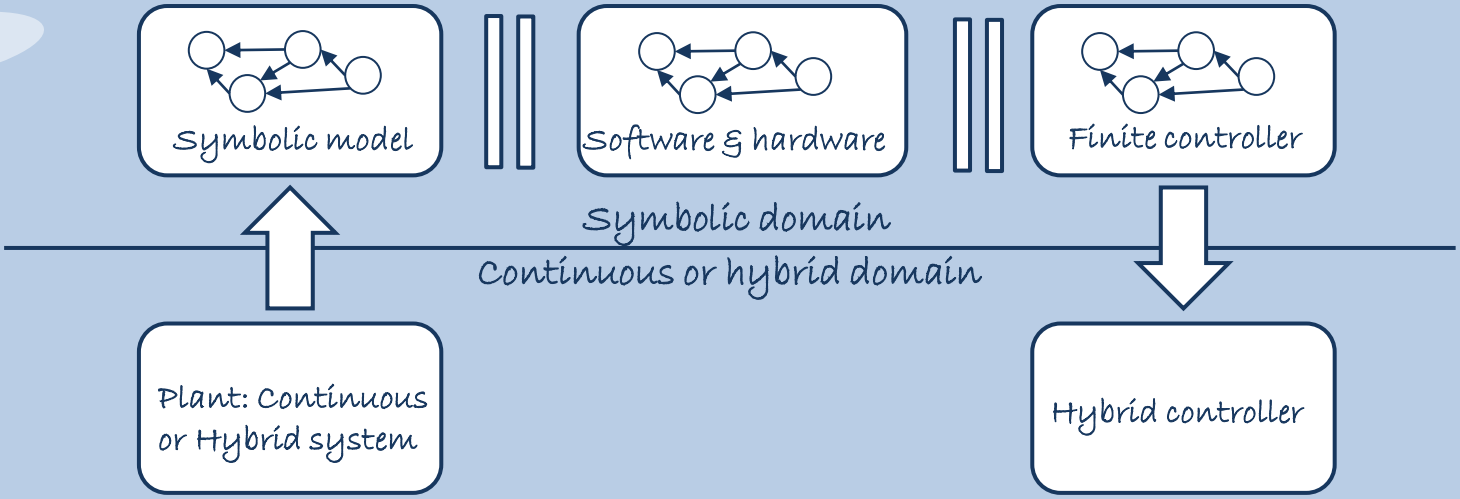
Maria Domenica Di Benedetto
Giordano Pola
Alessandro Borri
University of L'Aquila

Correct-by-design controller synthesis

DEWS CENTER OF EXCELLENCE

Correct-by-design embedded control software:

1. Construct a finite model T*(Σ) of the plant system Σ
2. Design a finite controller C that solves the specification S for T*(Σ)
3. Design a controller C' for Σ on the basis of C



Advantages:

- Integration of software and hardware constraints in the control design of purely continuous processes
- Use of computer science techniques to address complex specifications

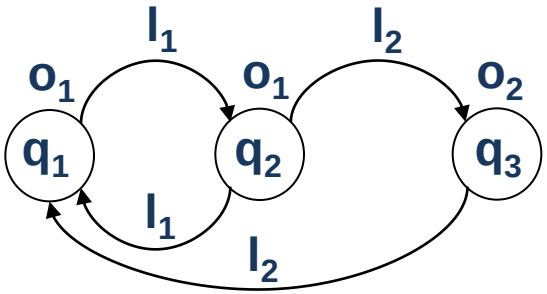
Preliminary definitions

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



We will follow standard practice and denote $(q, l, q') \in \longrightarrow$ by $q \xrightarrow{l} q'$

When $Q_0 = Q$ we write the LTS in a more compact notation as $T = (Q, L, \longrightarrow, O, H)$

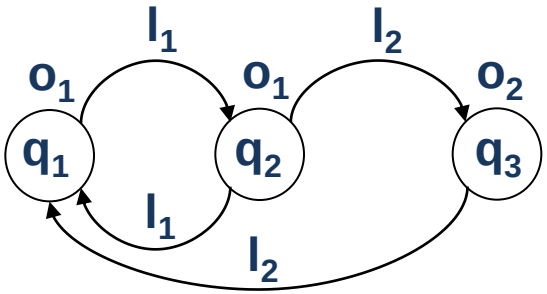
Preliminary definitions

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$

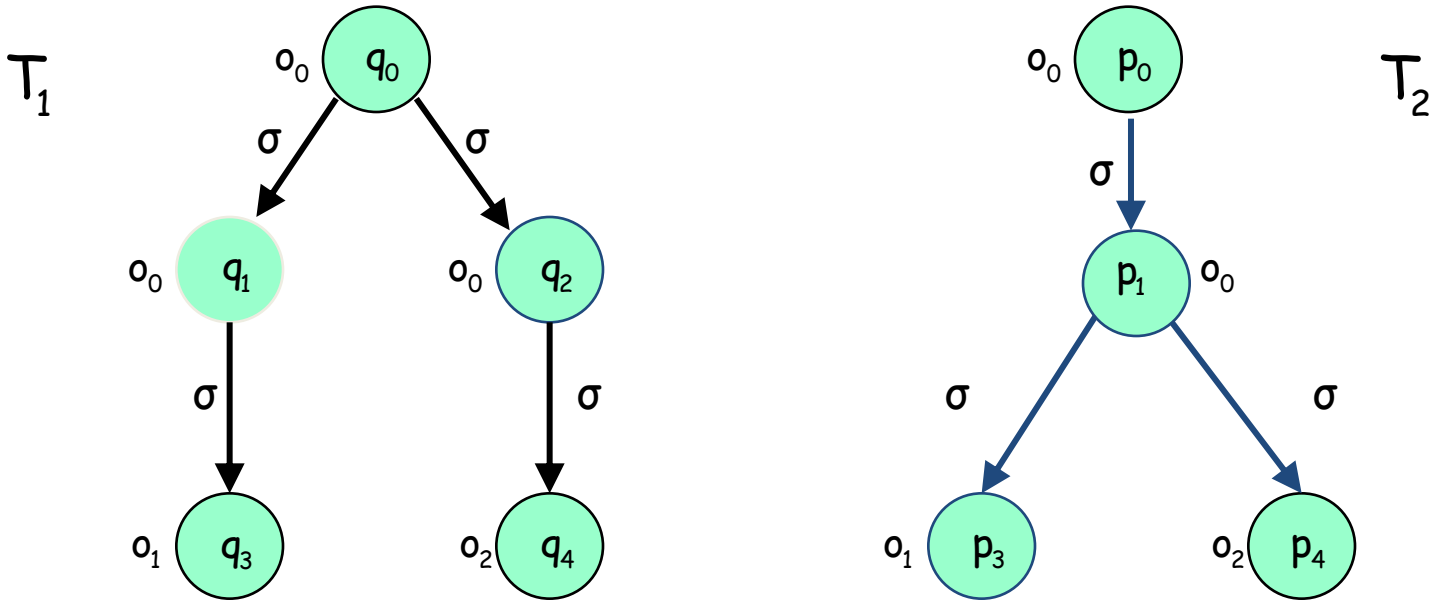


T is said:

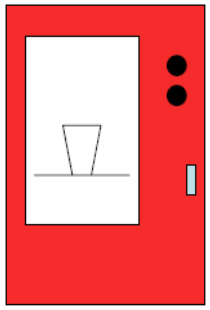
- countable if Q and L are countable sets
- symbolic/finite if Q and L are finite sets
- deterministic if for q and l there exists at most one p such that $q \xrightarrow{l} p$
- non-blocking if for any q there exist at least one l and one p such that $q \xrightarrow{l} p$
- metric if O is a metric space

A unified framework

Are T_1 and/or T_2 deterministic? Why?

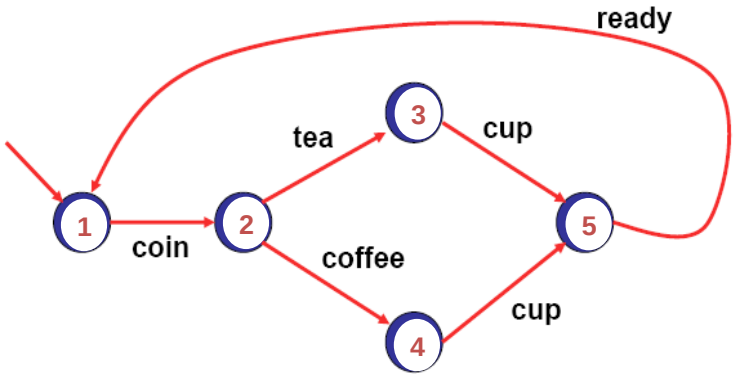


Vending machine

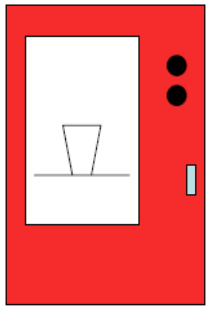


Vending machine

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

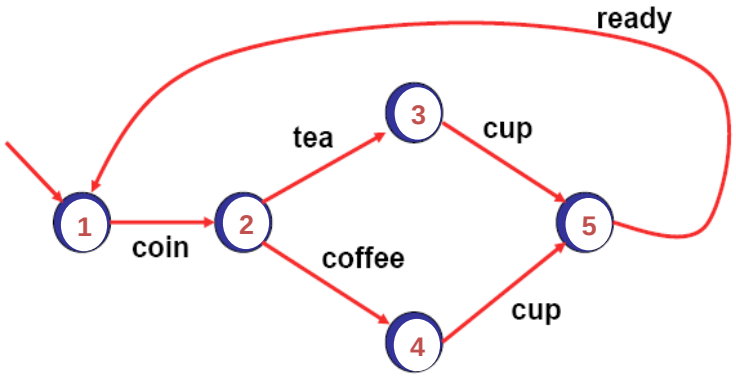


Vending machine



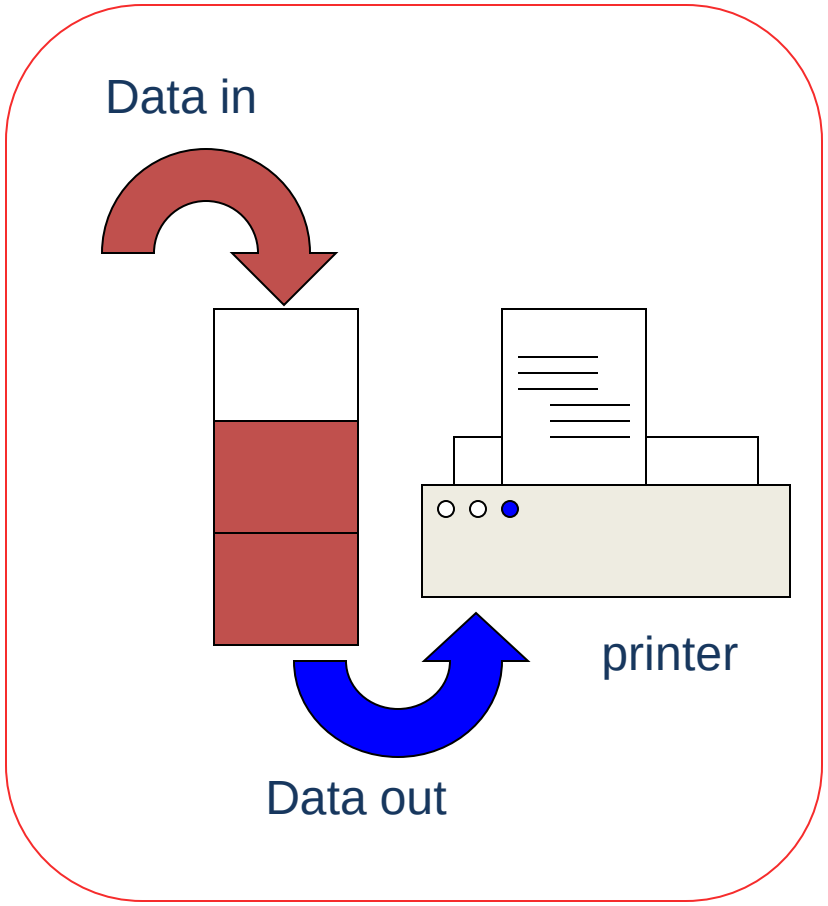
Vending machine

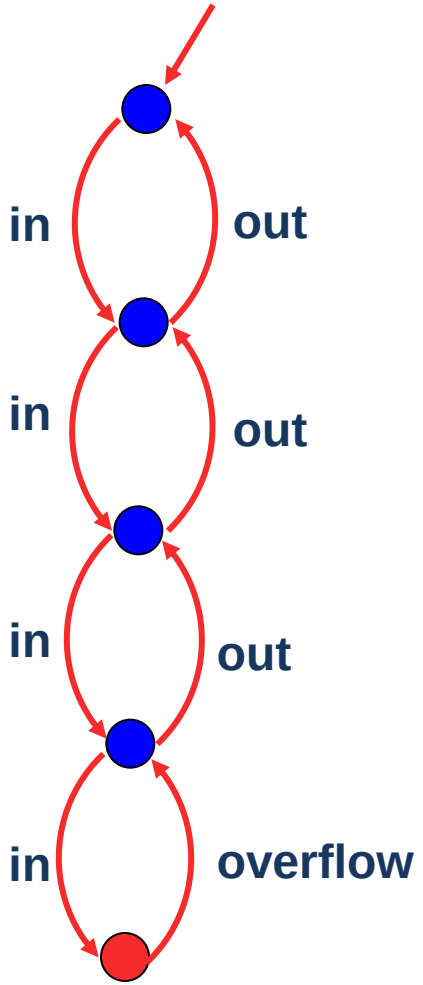
1. Insert coin(s)
2. Choose tea or coffee
3. Put the cup on the tray



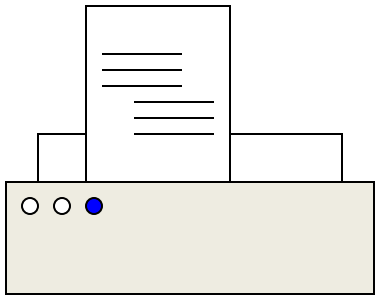
... Event driven versus time driven !

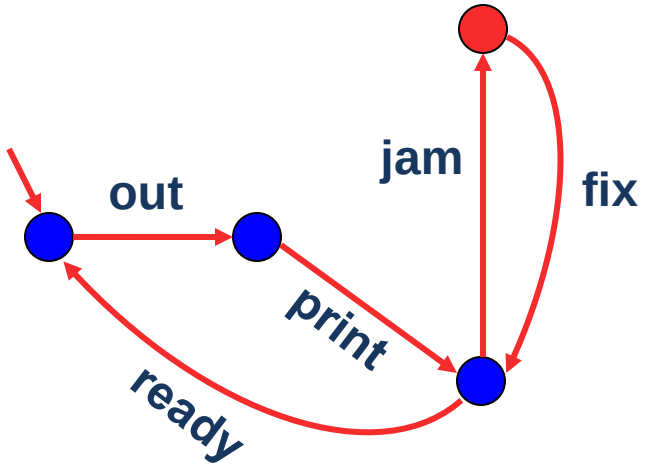
A printer data buffer





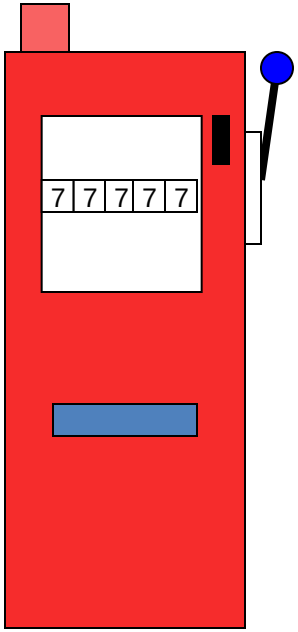
A printer



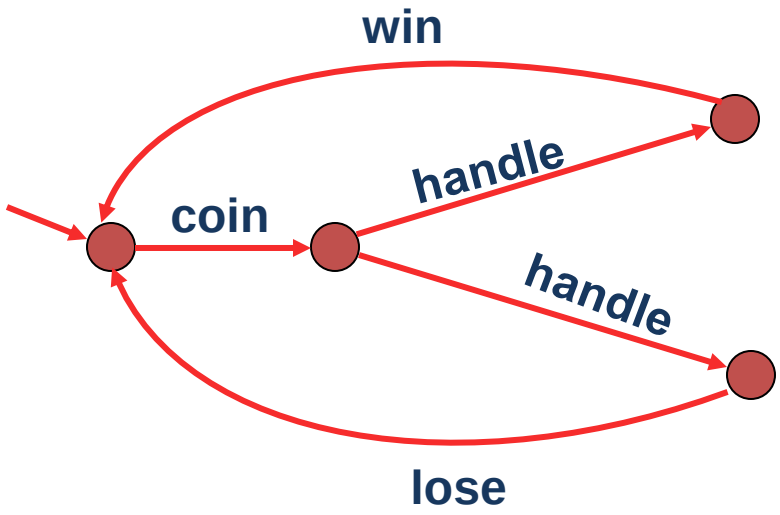


The printer receives data from the buffer, and print it out. Once the printout is ready, the printer is ready to receive new data. While printing, the paper can jam and need to be fixed before the printing process can resume.

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.

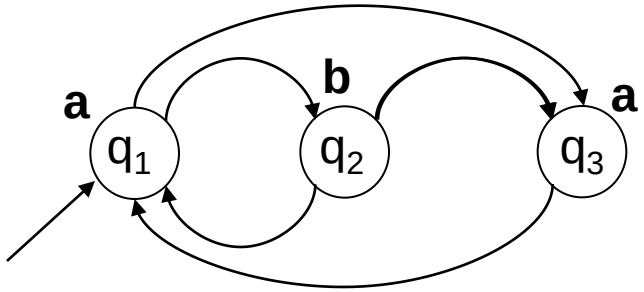
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



Starting from q_1 with observation a,

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

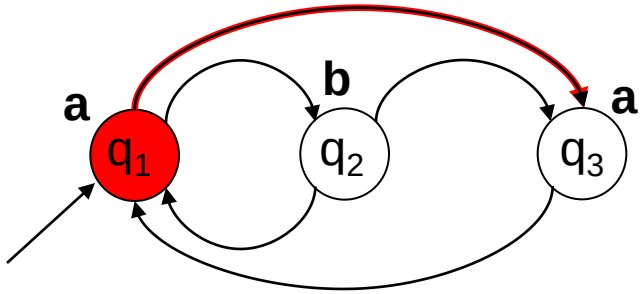
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



Starting from q_1 with observation a,

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

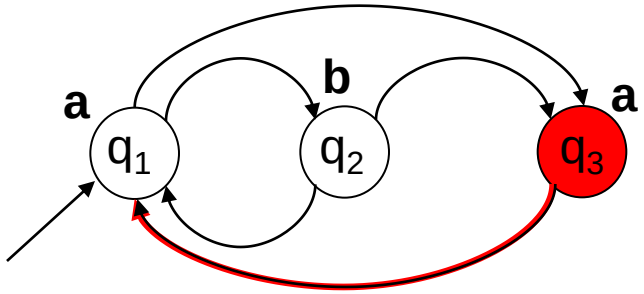
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



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

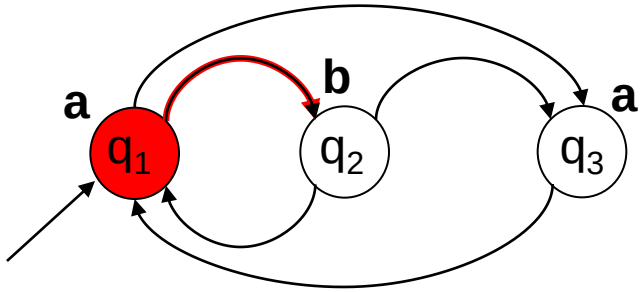
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



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

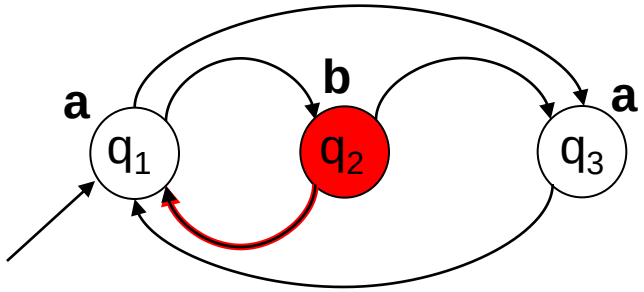
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



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

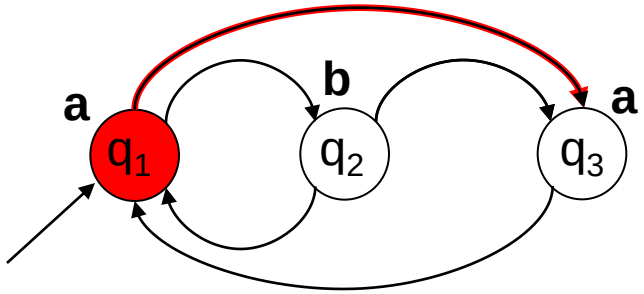
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



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

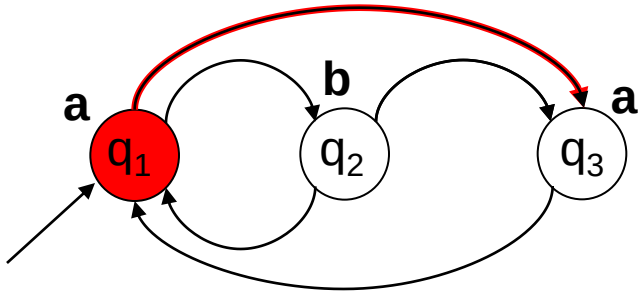
A unified framework

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



Starting from q_1 with observation a,

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

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

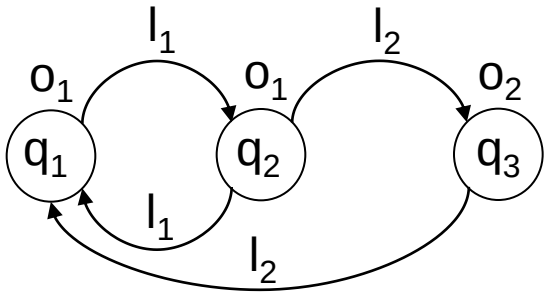
A unified framework: discrete processes

Definition A Labelled Transition System (for short also called LTS or System) is a tuple:

$$T = (Q, Q_0, L, \longrightarrow, O, H),$$

consisting of:

- a set of states Q
- a set of initial states Q_0
- a set of labels L
- a transition relation $\longrightarrow \subseteq Q \times L \times Q$
- a set of outputs O
- an output function $H: Q \rightarrow O$



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.

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 $u : R \rightarrow R^m$
- $p \xrightarrow{u} q$, 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:

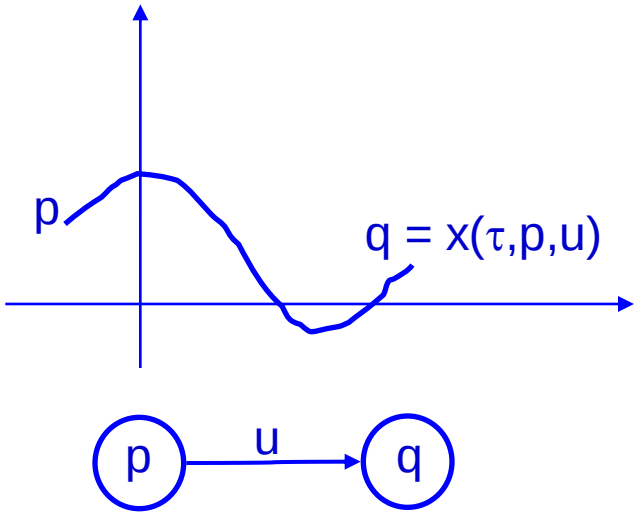
$$\dot{x} = f(x,u) \qquad x \in R^n,\ u \in R^m$$

we can define the following LTS

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

where:

- $Q = R^n$
- L is the collection of control signals $u : R \rightarrow R^m$
- $p \xrightarrow{u} q$, 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: continuous processes

Given a control system Σ:

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

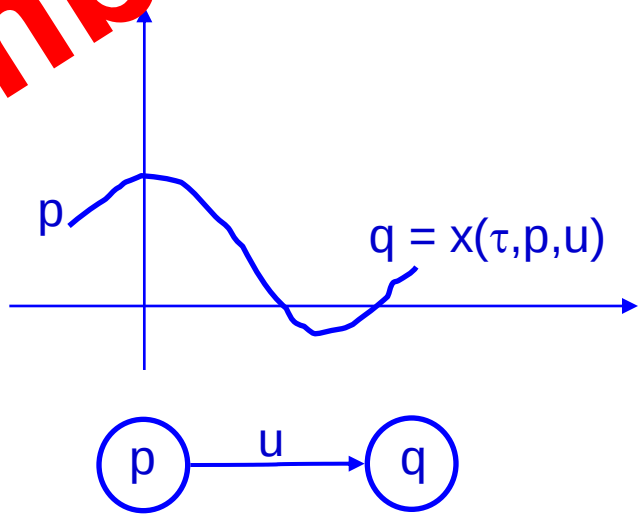
we can define the following LTS

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

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

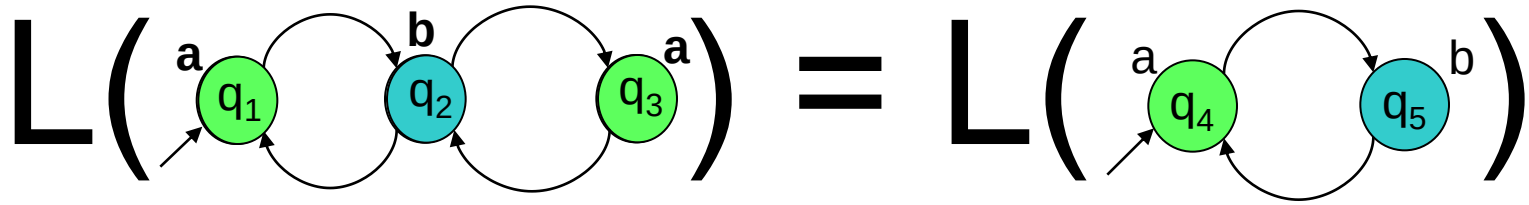


T(Σ) is not symbolic !

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

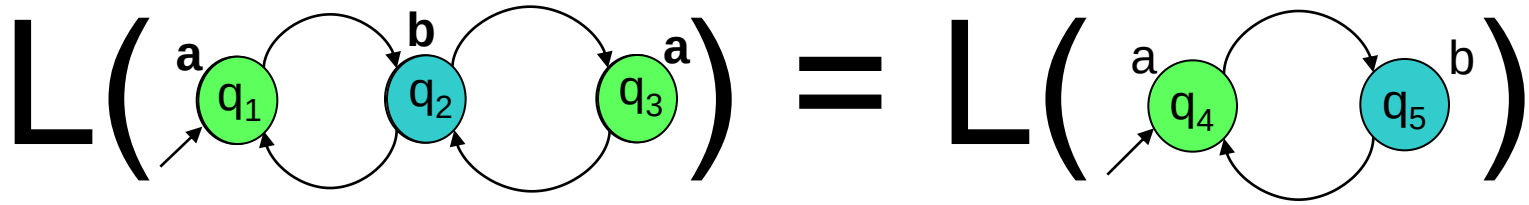
Language equivalence

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. :

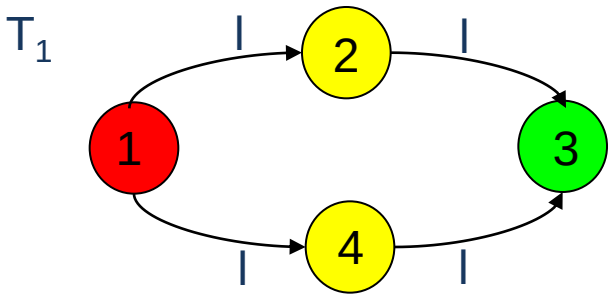
- (reflexivity) $L(T_1) = L(T_1)$
- (symmetry) $L(T_1) = L(T_2) \Rightarrow L(T_2) = L(T_1)$
- (transitivity) $L(T_1) = L(T_2) \wedge L(T_2) = L(T_3) \Rightarrow L(T_1) = L(T_3)$

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!

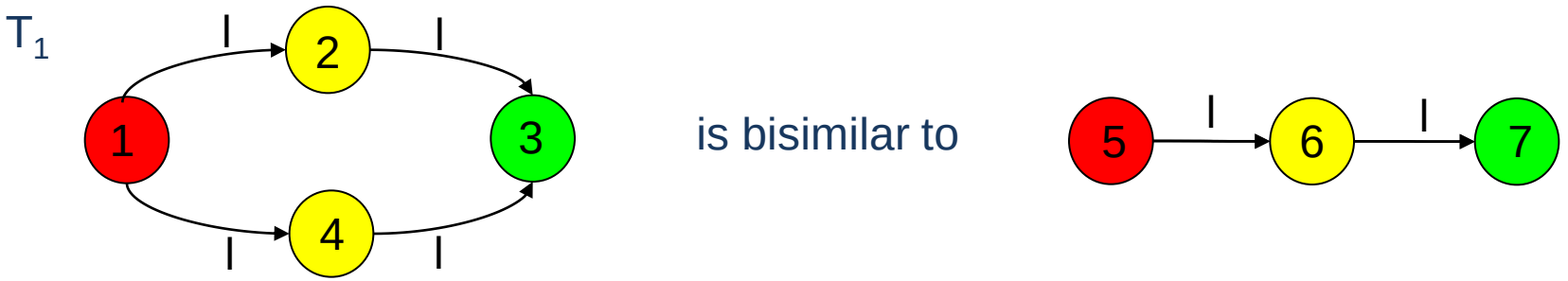


Bisimulation equivalence

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

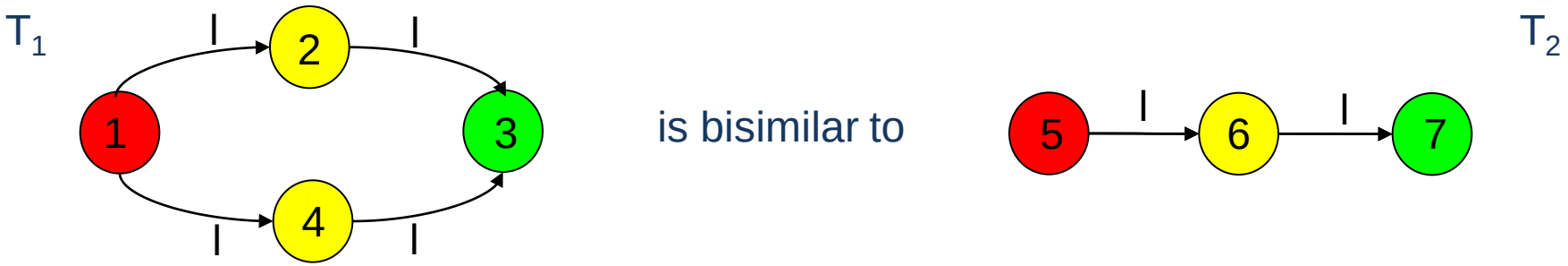
D.M.R. Park (1981), Concurrency and automata on infinite sequences. LNCS, vol. 104

… 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 such 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 such that $H_1(p_1) = H_2(p_2)$



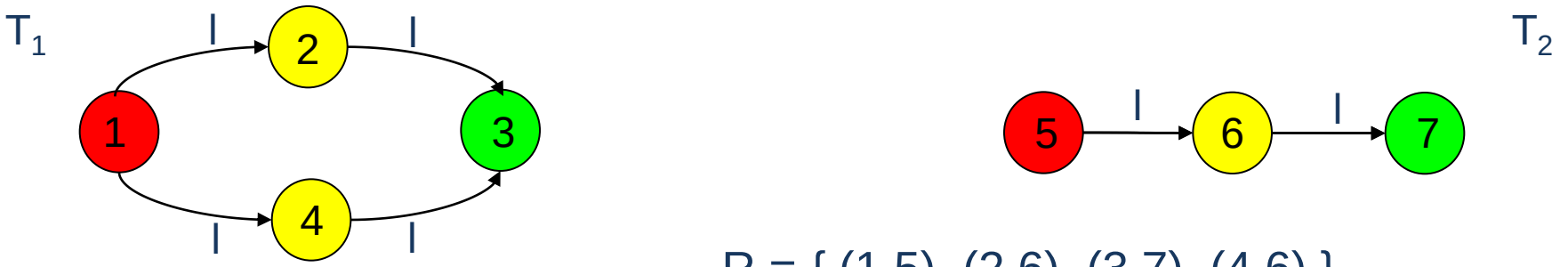
Bisimulation equivalence

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 such 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 such that $H_1(p_1) = H_2(p_2)$



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

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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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

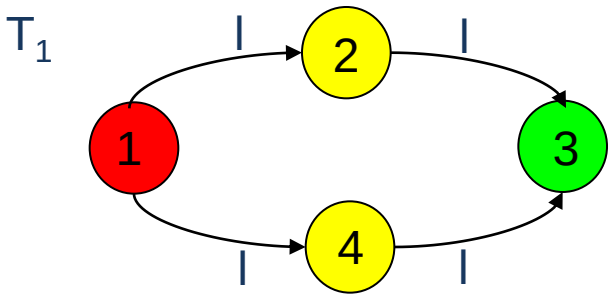
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

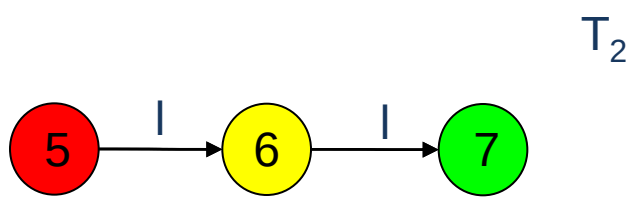
$$R \subseteq Q_1 \times Q_2$$

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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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

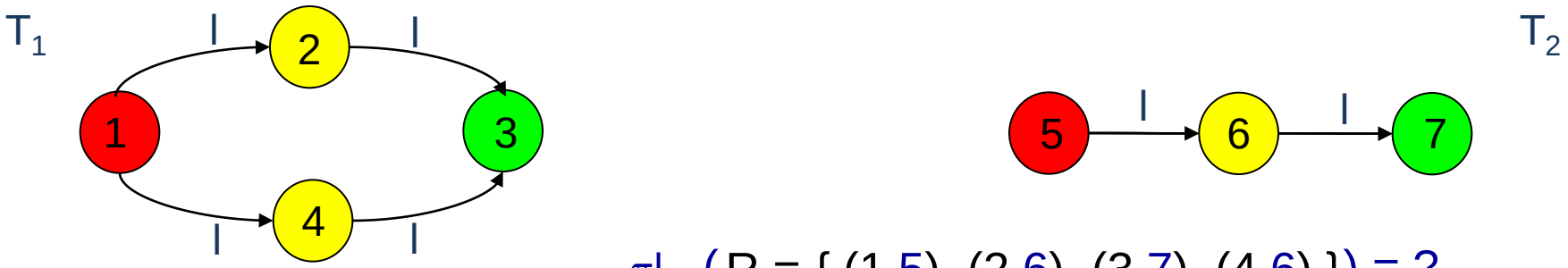
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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

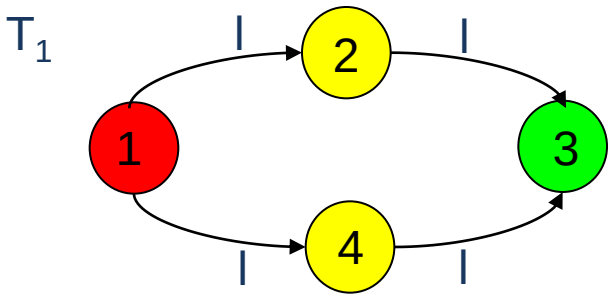
Given $T_1 = (Q_1, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, L_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

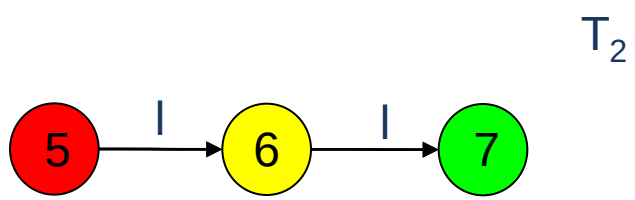
$$R \subseteq Q_1 \times Q_2$$

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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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 such 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 such that $H_1(p_1) = H_2(p_2)$

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

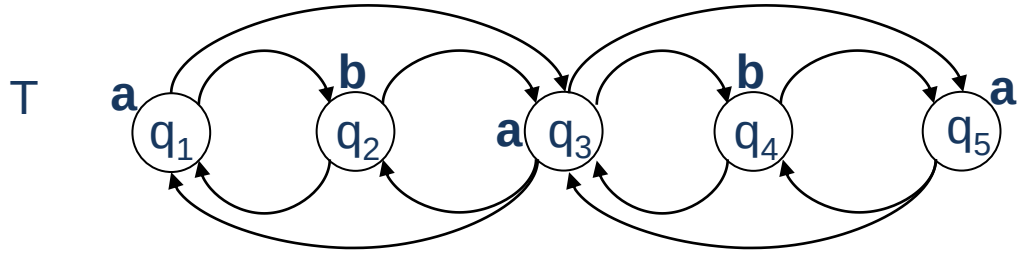
Bisimulation equivalence is an equivalence relation on the space of transition systems, i.e. :

- (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:

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

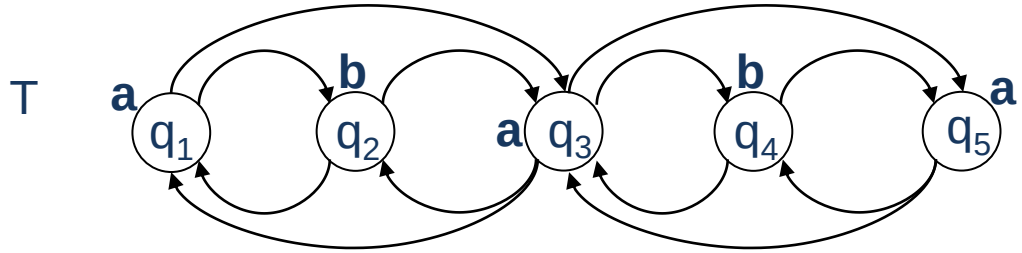
- Maximal bisimulation relation is an equivalence relation on Q,

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:

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

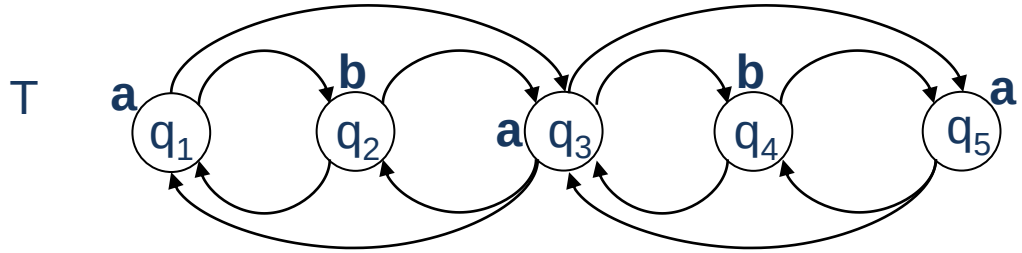
- Maximal bisimulation relation is an equivalence relation on Q,

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:

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

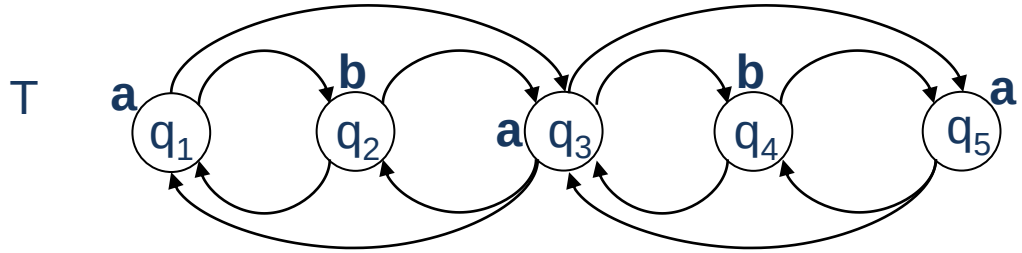
- Maximal bisimulation relation is an equivalence relation on Q,

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:

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

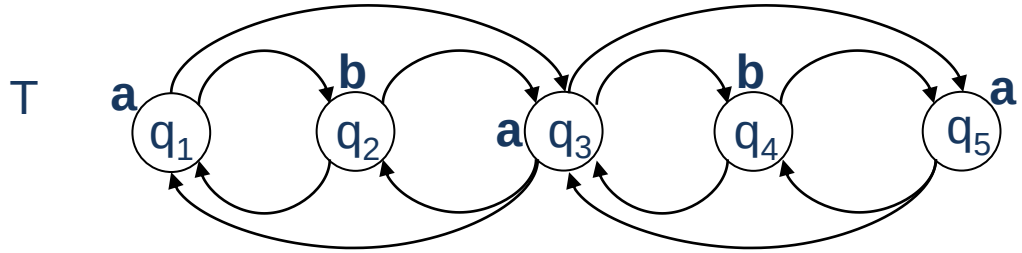
- Maximal bisimulation relation is an equivalence relation on Q,

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:

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

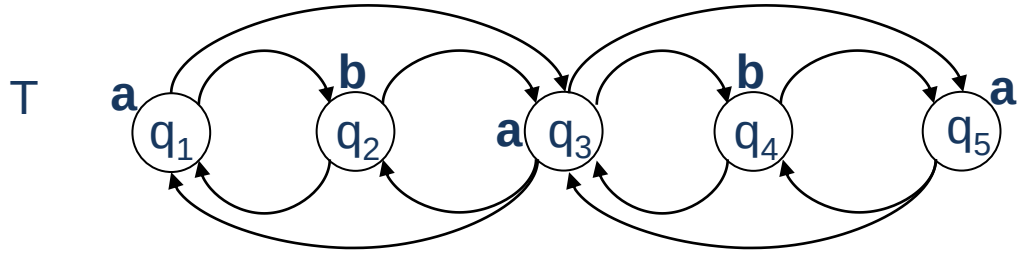
- Maximal bisimulation relation is an equivalence relation on Q,

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:

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

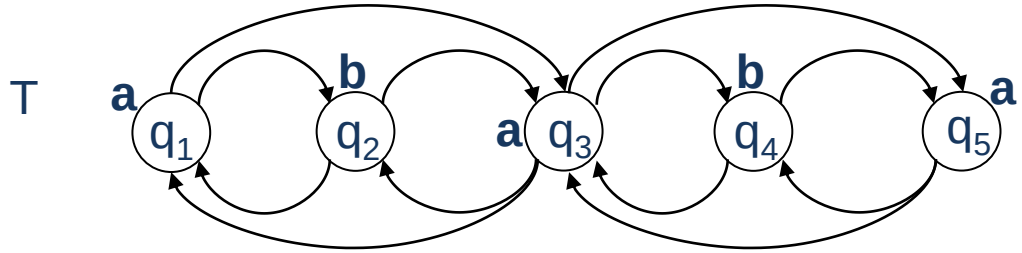
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive (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:

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

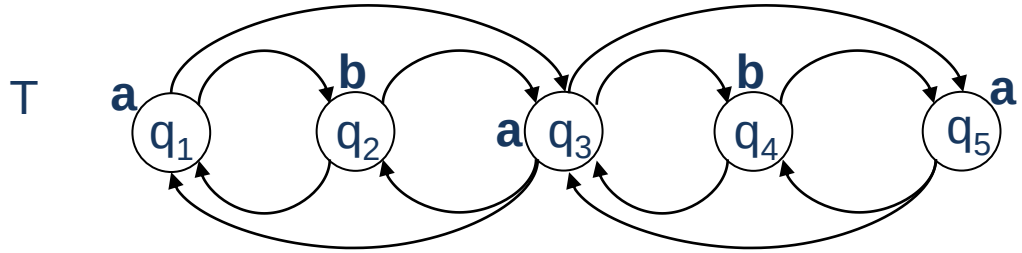
- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive (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,

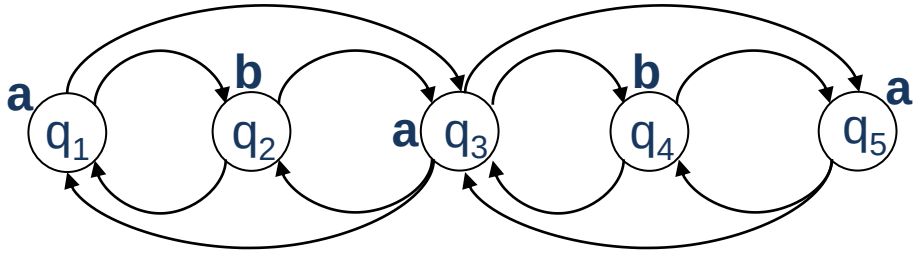
i.e. it is reflexive, symmetric and transitive (not all bisimulation relations are so!)

- 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:

T



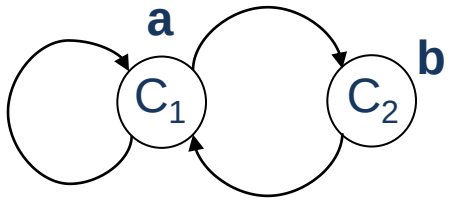
- Maximal bisimulation relation between T and T:

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

- 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!

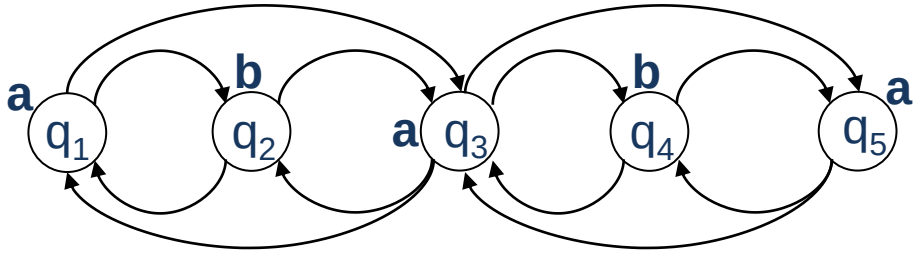


T*

Bisimulation equivalence

Bisimulation as a tool for reduction:

T



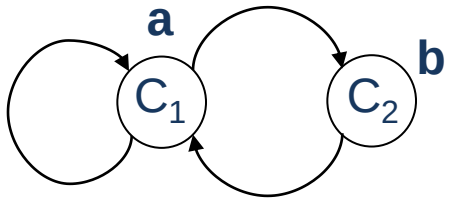
- Maximal bisimulation relation between T and T:

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

- Maximal bisimulation relation is an equivalence relation on Q,

i.e. it is reflexive, symmetric and transitive

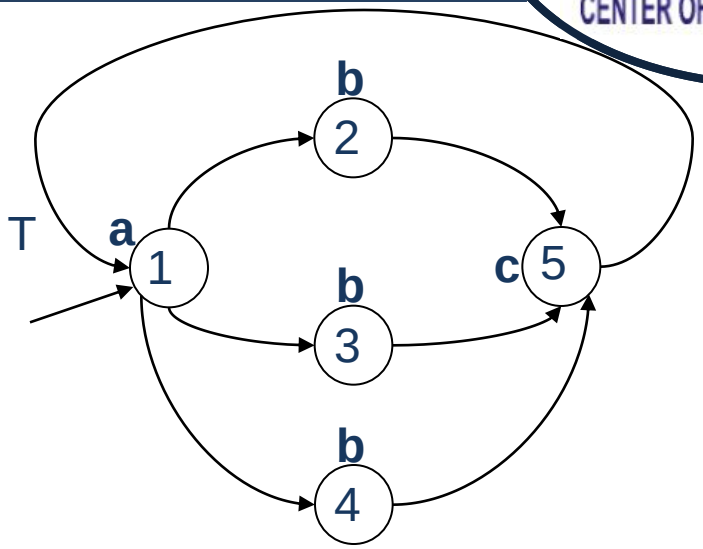
- Construct equivalence classes induced by R*, i.e. $C_1 = \{q_1,q_3,q_5\}$ and $C_2 = \{q_2,q_4\}$
- Partition the state space of T as $Q = C_1 \cup C_2$
- Construct the quotient T* of T induced by R*, i.e.
- T* is minimal!



T*

Bisimulation equivalence

Bisimulation as a tool for reduction:



EXAMPLE:

All relations listed below are total bisimulation relations between T and itself

- R={(1,1),(2,2),(3,3),(4,4),(5,5)} is closed wrt reflexivity, symmetry and transitivity (last one is trivial) . Hence it is an equivalence relation.
- R={(1,1),(3,3),(4,4),(5,5),(2,3),(3,2),(3,4),(4,3),(2,4),(4,2)} is closed wrt symmetry and transitivity but not reflexivity (the pair (2,2) is missing). Hence it is not an equivalence relation.
- R={(1,1),(2,2),(3,3),(4,4),(5,5),(2,3),(3,4),(2,4)} is closed wrt reflexivity and transitivity but not symmetry (pairs (3,2), (4,3) and (4,2) are missing). Hence it is not an equivalence relation.
- R={(1,1),(2,2),(3,3),(4,4),(5,5),(2,3),(3,2),(3,4),(4,3)} is closed wrt reflexivity and symmetry but not transitivity (pairs (2,4) and (4,2) are missing). Hence it is not an equivalence relation.

Bisimulation equivalence

Bisimulation as a tool for reduction:

Consider an LTS $T = (Q, L, \longrightarrow, O, H)$ and the maximal bisimulation relation R^* between T and itself. The quotient of T induced by R^* is the LTS

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

Where:

- Q^* is the collection of equivalence classes C_i induced by R^* on Q
- $L^* = L$
- $C_1 \xrightarrow{l}{}^* C_2$ if there exist $q_i \in C_i$ (i=1,2) s.t. $q_1 \xrightarrow{l}{}^* q_2$ (in fact all states in C_1 reach a state in C_2 when label l is applied)
- $O^* = O$
- $H^*(C_i) = H(q_i)$ for any $q_i \in C_i$

(The above definition works with any bisimulation and equivalence relation (not only R^*))

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?

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

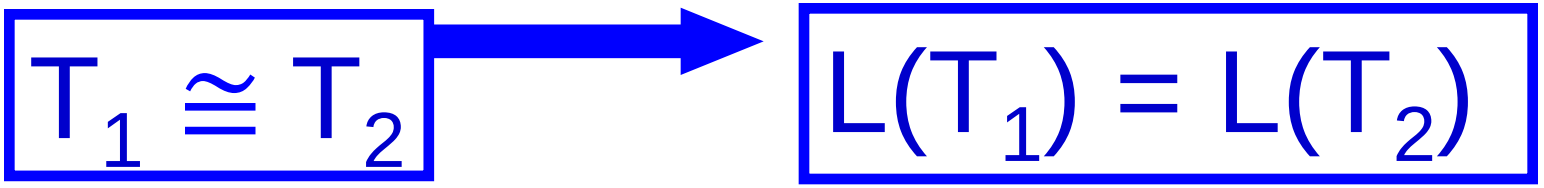
$$B(k+1) = \{ (q,p) \in Q \times Q \text{ s.t.}$$
$$\forall\, q \xrightarrow{l} q' \;\exists\, p \xrightarrow{l'} p' \text{ s.t. } (q',p') \in B(k) \wedge$$
$$\forall\, p \xrightarrow{l} p' \;\exists\, q \xrightarrow{l'} q' \text{ s.t. } (q',p') \in B(k) \}$$

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

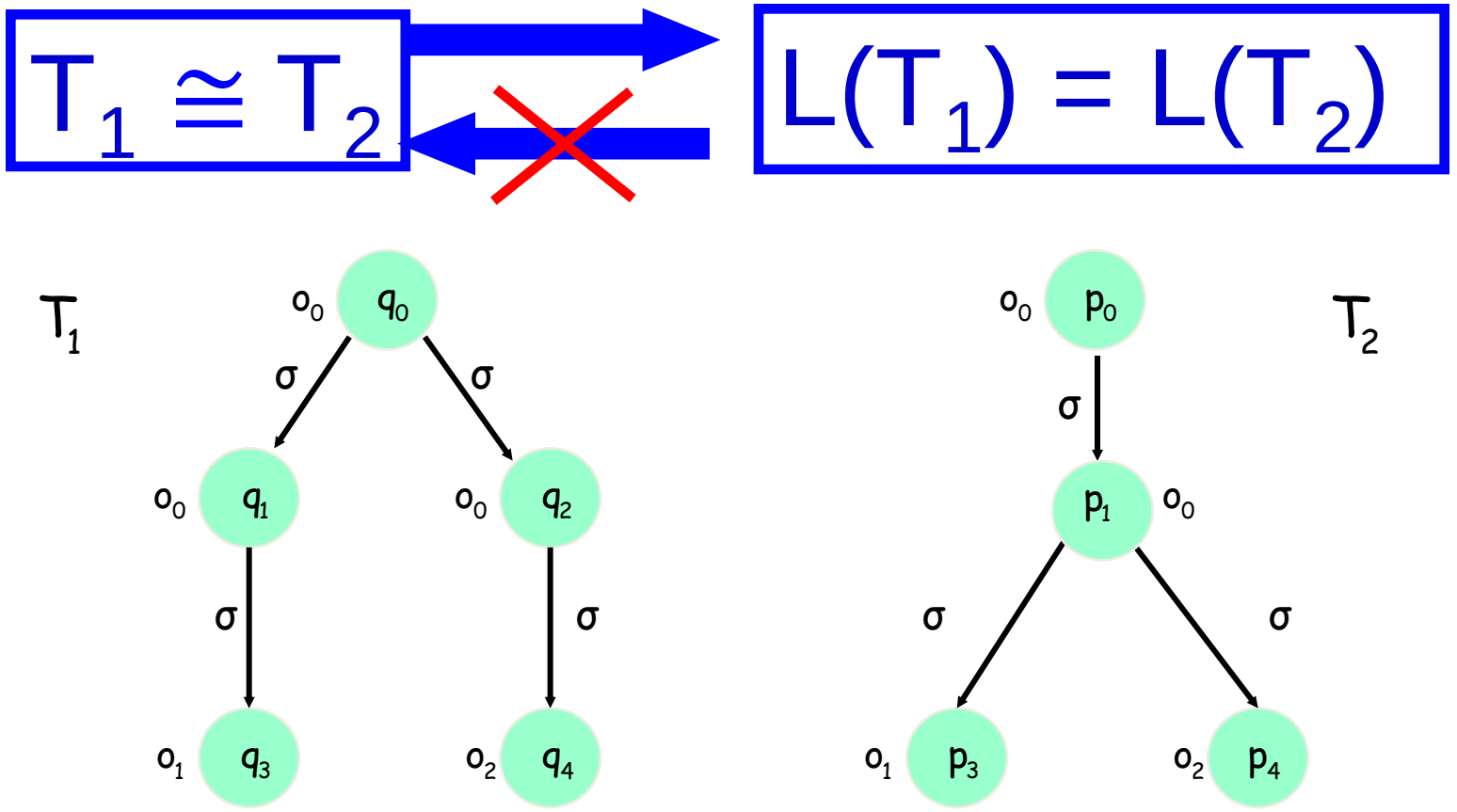
If Q is finite, termination of the algorithm is guaranteed in polynomial time!

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 such 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 such that $H_1(p_1) = H_2(p_2)$

LTSs T_1 and T_2 are ***bisimilar***, denoted $T_1 \cong T_2$, 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 such 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 such that $H_1(p_1) = H_2(p_2)$~~

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

is ***simulated*** by 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 such that $H_1(p_1) = H_2(p_2)$

LTS 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 $(q_1, q_2) \in R$

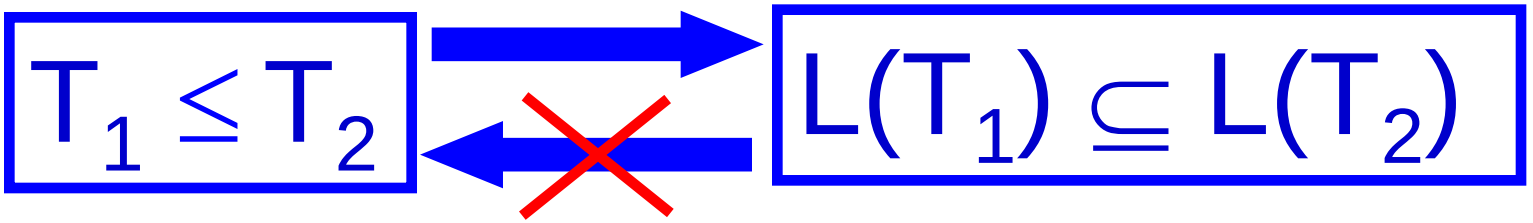
- $H_1(q_1) = H_2(q_2)$
- $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 such that $H_1(p_1) = H_2(p_2)$

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

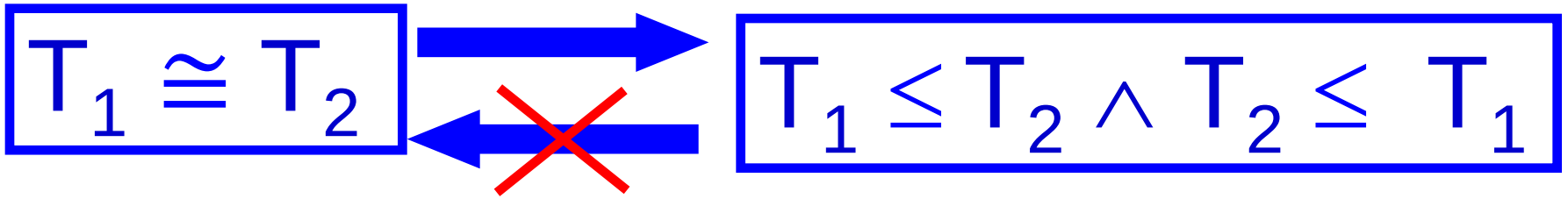
Simulation relation is not equivalence relation on the space of transition systems, indeed:

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

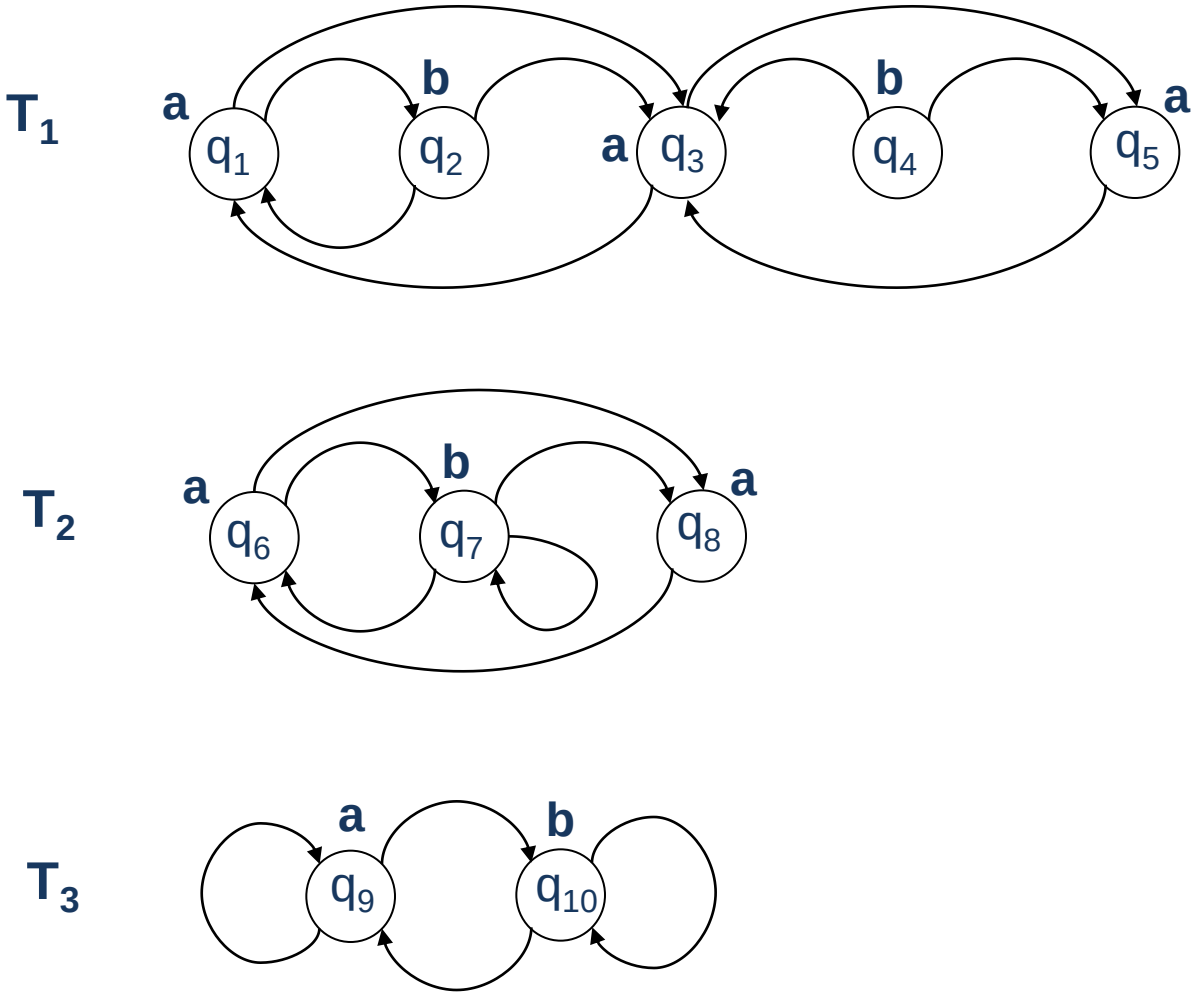
Language inclusion and simulation relation



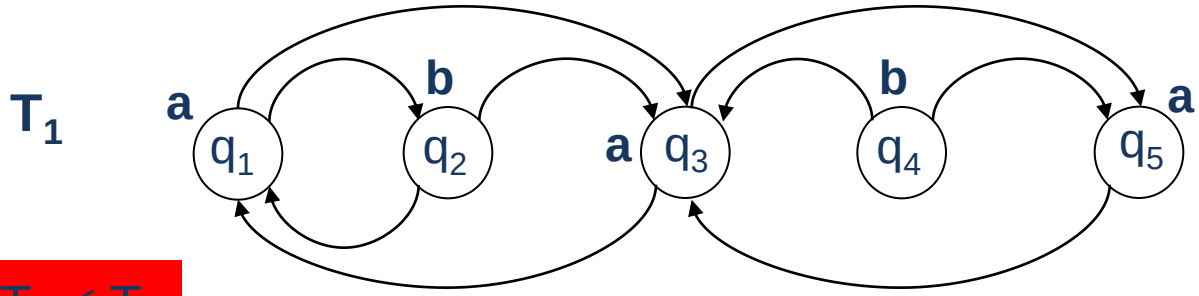
Simulation relation and bisimulation equivalence



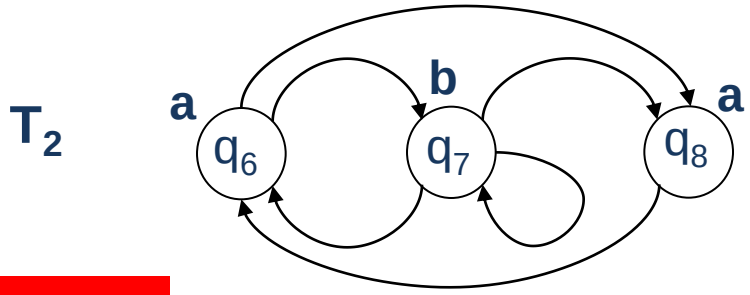
Abstraction and simulation



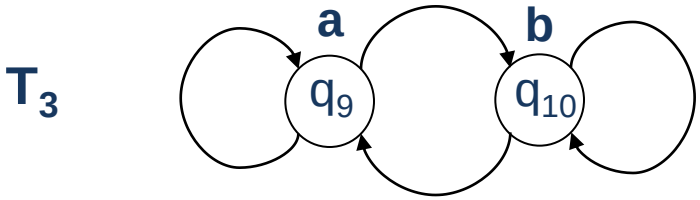
Abstraction and simulation



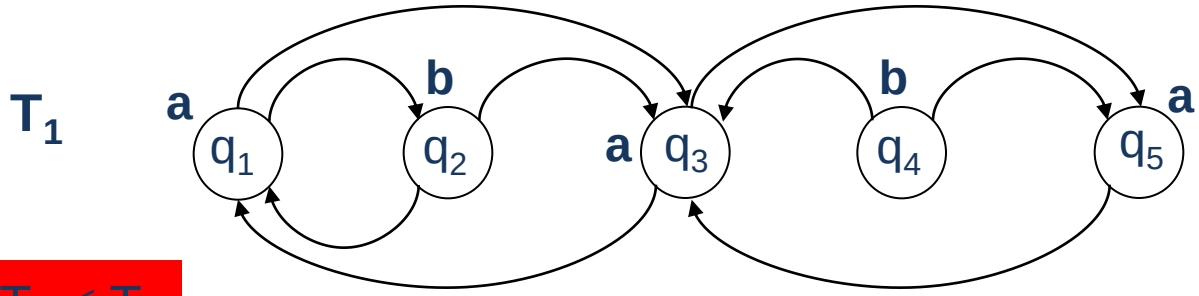
$T_1 \leq T_2$



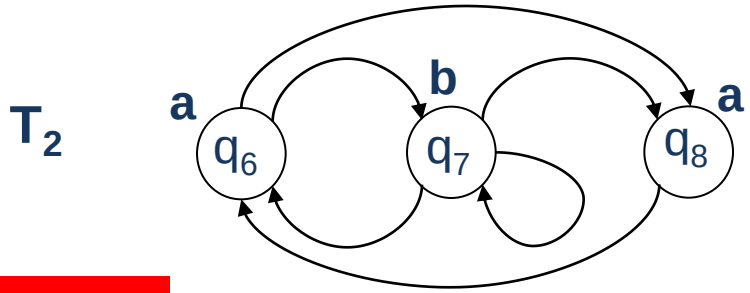
$T_2 \leq T_3$



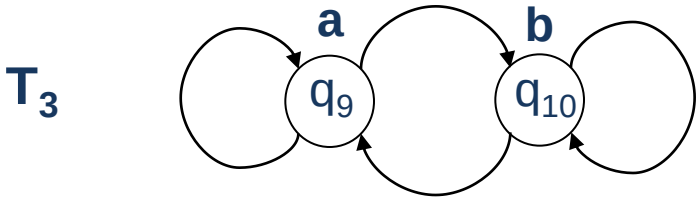
Abstraction and simulation



$T_1 \leq T_2$

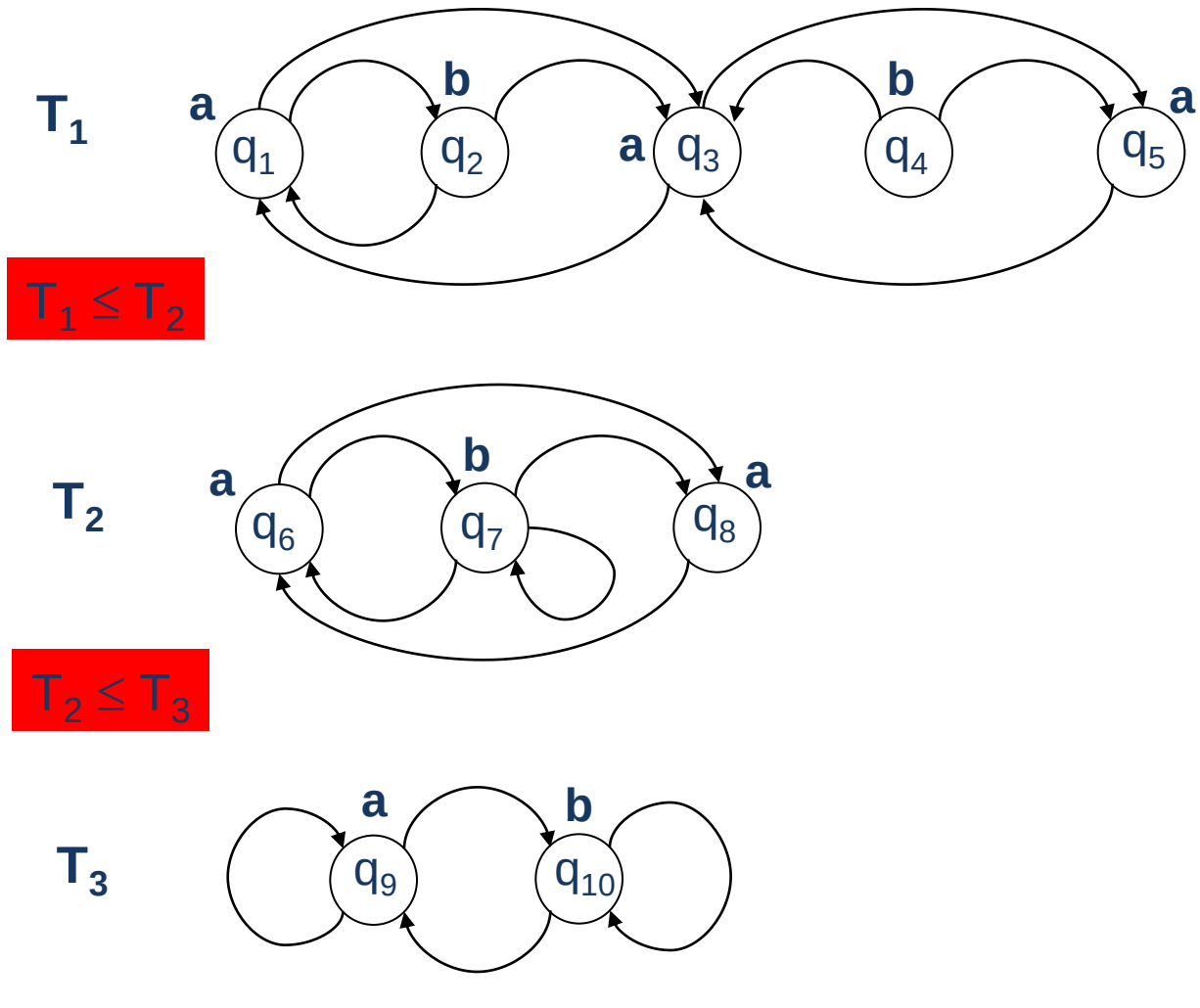


$T_2 \leq T_3$

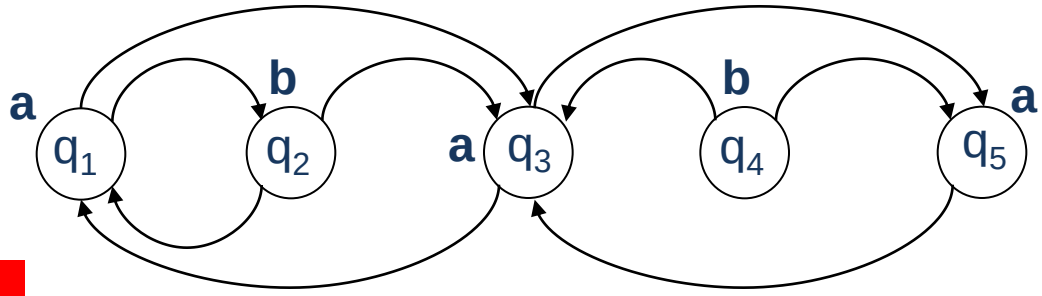


… what is R in the two cases?

Abstraction and simulation



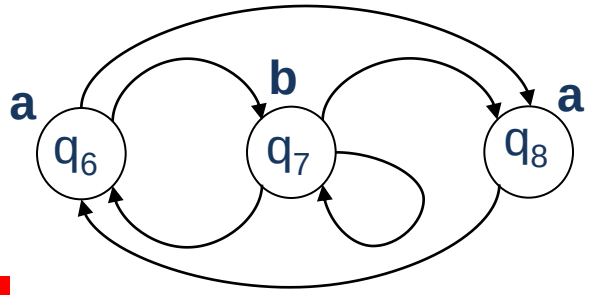
Abstraction and simulation



T_1

5 states

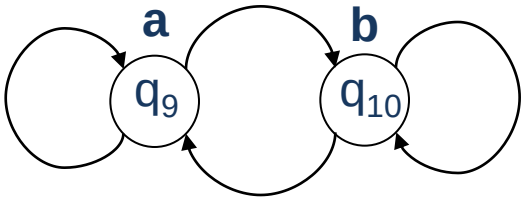
$T_1 \leq T_2$



T_2

3 states

$T_2 \leq T_3$



T_3

2 states

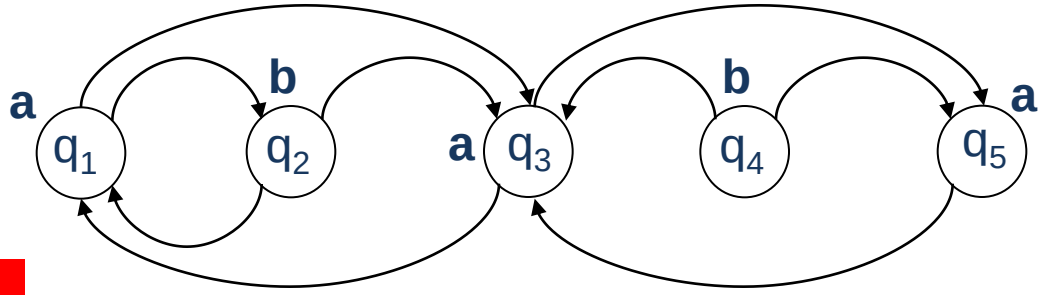
Asbstraction

Refinement

Abstraction and simulation

T_0 … infinite number of states…

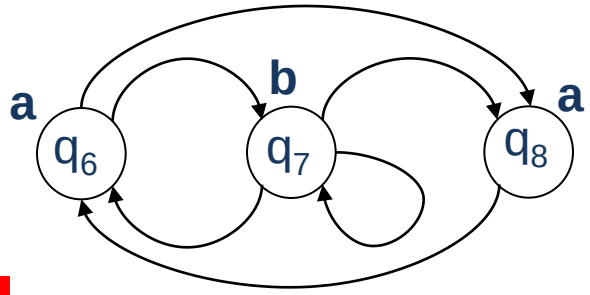
T_1



5 states

$T_1 \leq T_2$

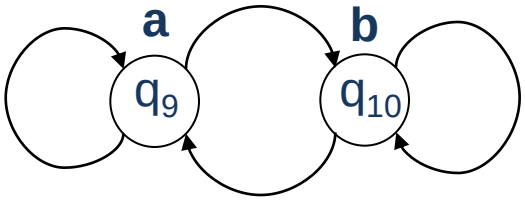
T_2



3 states

$T_2 \leq T_3$

T_3



2 states

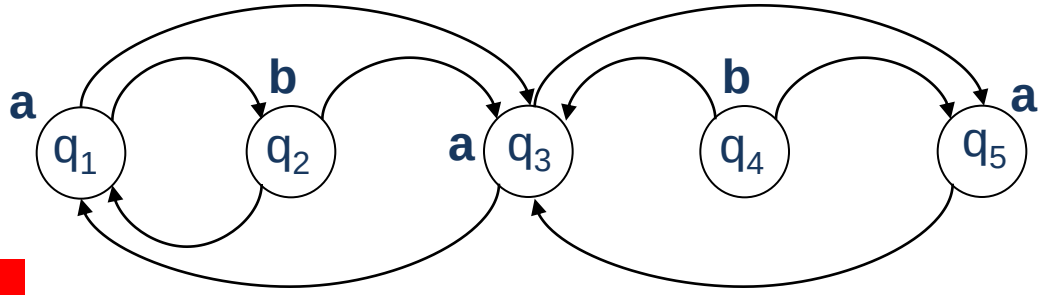
Asbstraction

Refinement

Abstraction and simulation

T_0 … infinite number of states…

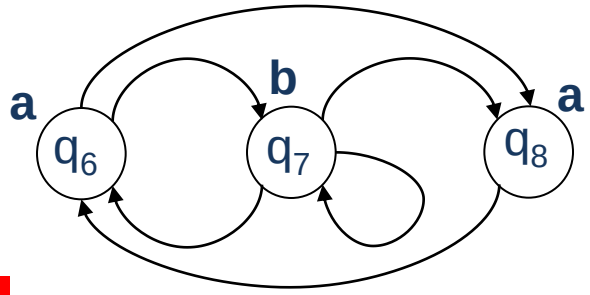
T_1



5 states

$T_1 \leq T_2$

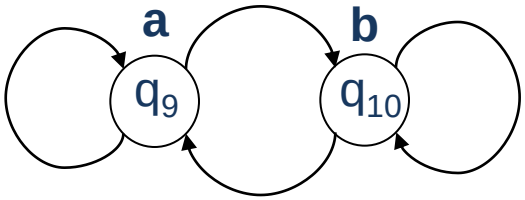
T_2



3 states

$T_2 \leq T_3$

T_3

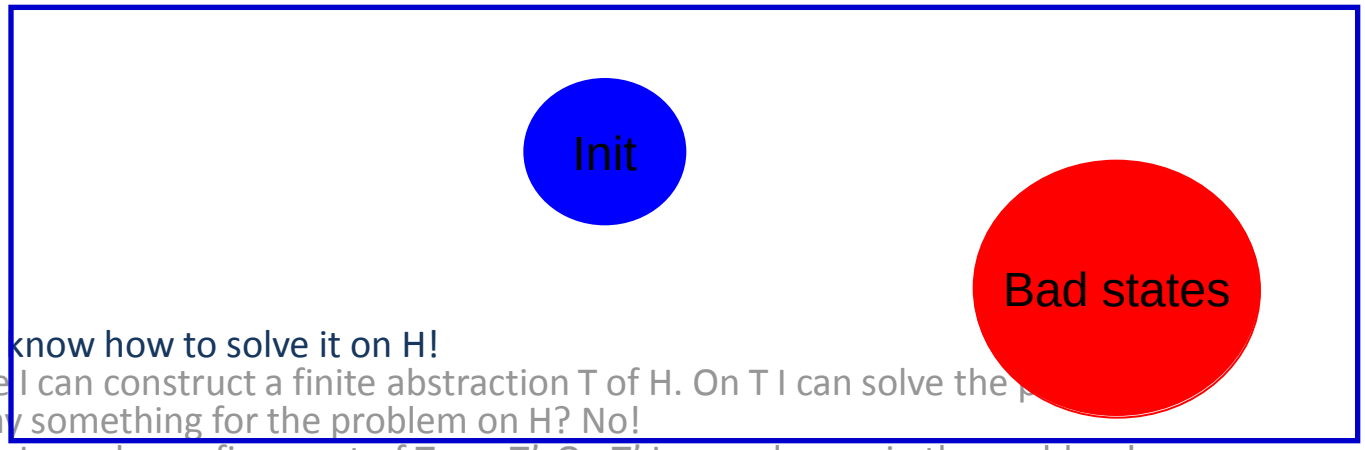


2 states

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?

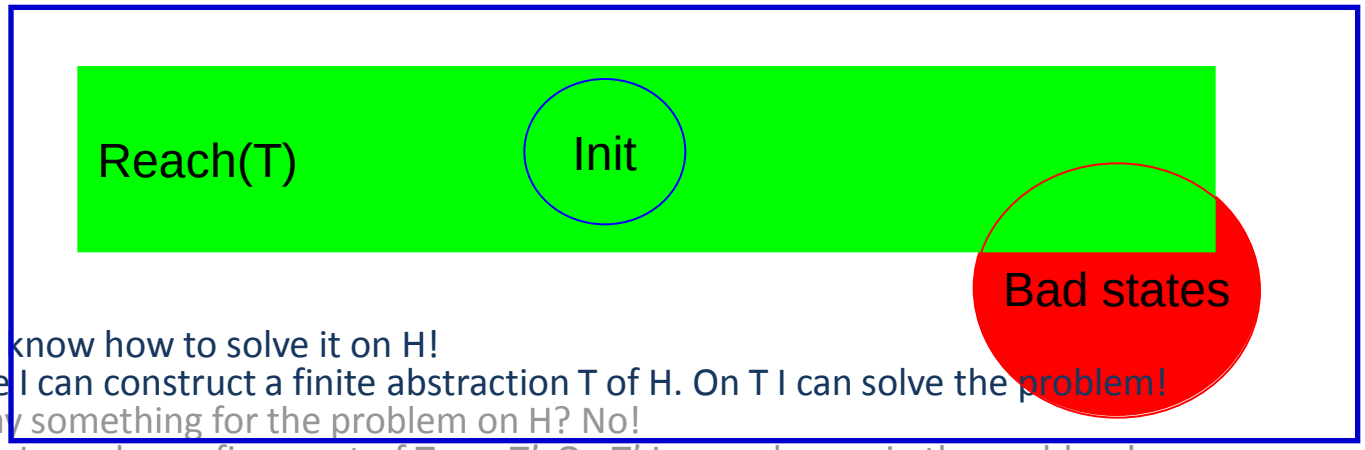


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the p
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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?

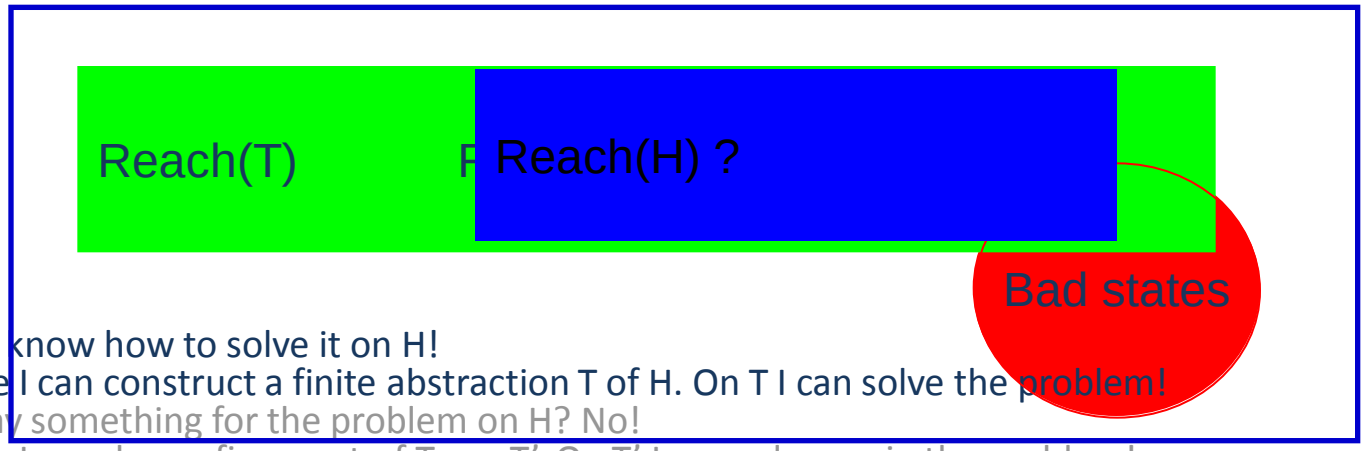


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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?

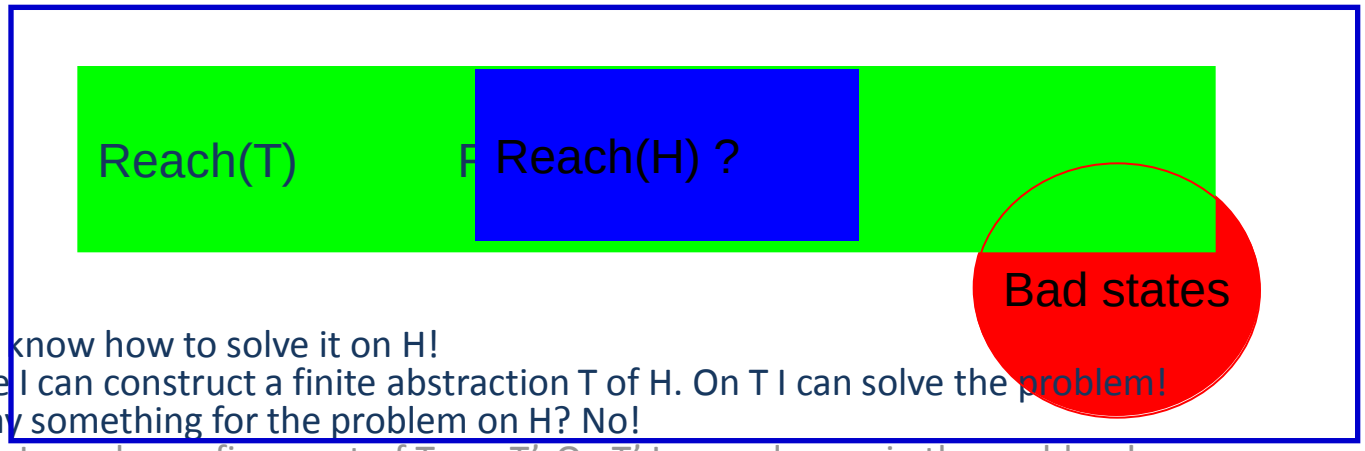


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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?

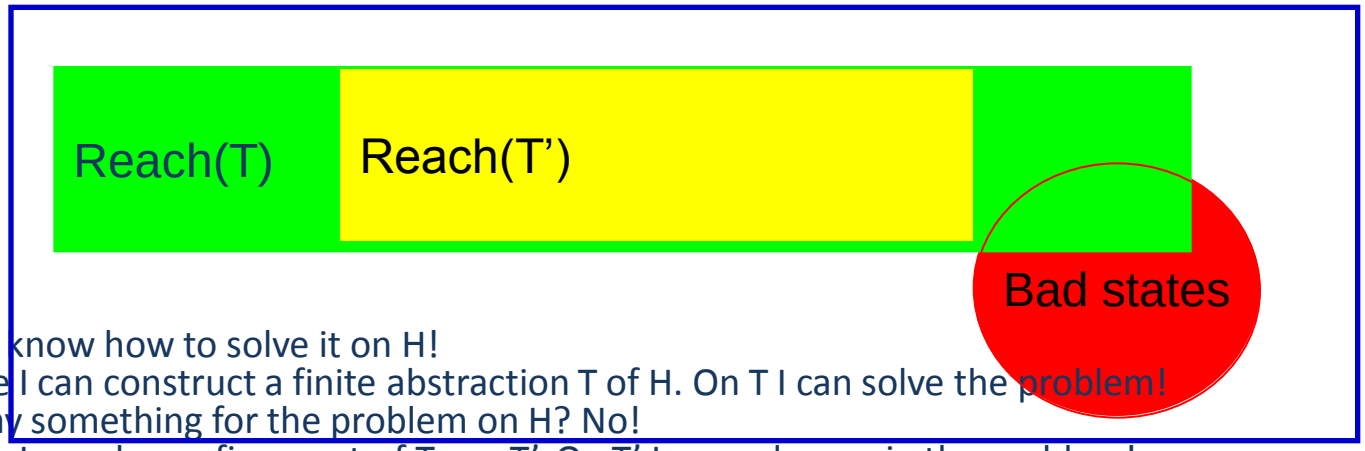


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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?

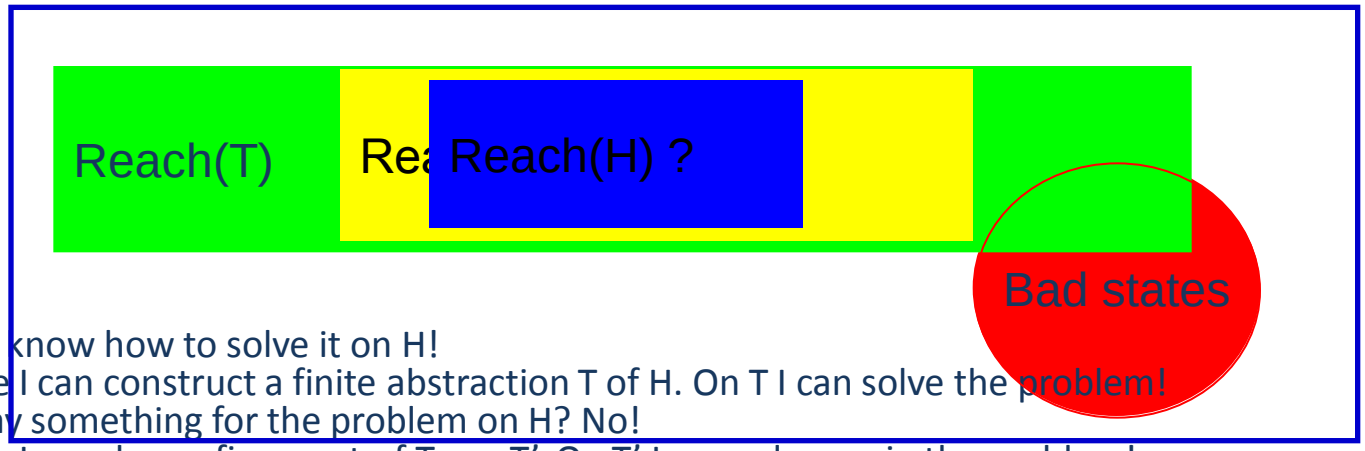


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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?

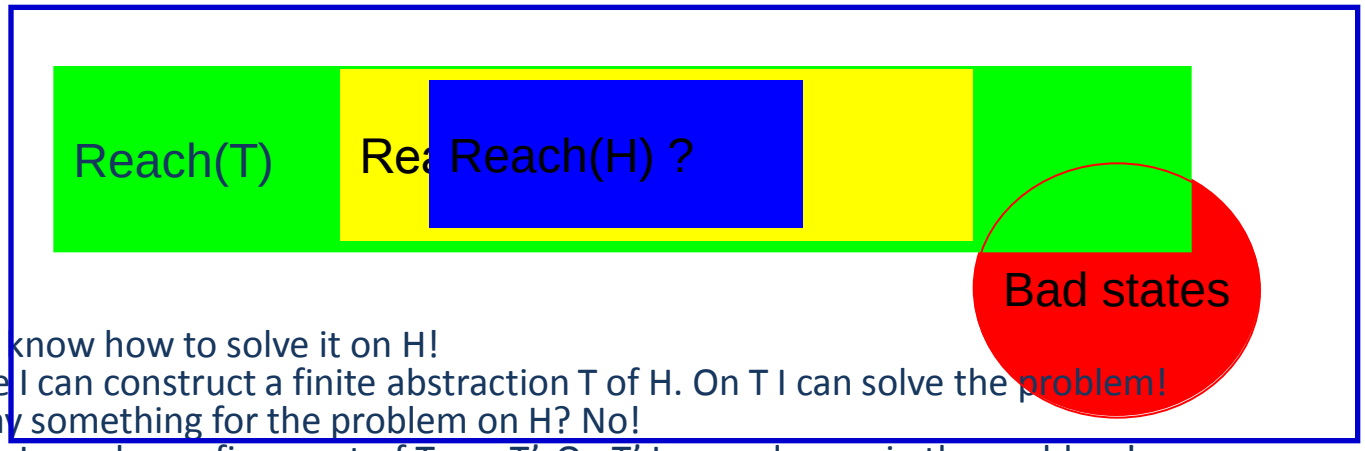


- I do not know how to solve it on H!
- Suppose I can construct a finite abstraction T of H. On T I can solve the problem!
- May I say something for the problem on H? No!
- Suppose I can do a refinement of T, say T'. On T' I can solve again the problem!
- May I say something for the problem on H? Yes!

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!

Why abstraction?

Dealing with non-determinism: alternating bisimulation

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 such 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 such that $H_1(p_1) = H_2(p_2)$

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

- Bisimulation equivalence preserves most of the dynamical properties of interest!
- However, in the presence of non-determinism, this notion can be shown not to capture correctly the robustness requirement in control problems!

Dealing with non-determinism: alternating bisimulation

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

- $H_1(q_1) = H_2(q_2)$
- For any $(q_1, q_2) \in R$ and for any $l_1 \in L_1$, there exists $l_2 \in L_2$ s.t. the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies the existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 with $(p_1, p_2) \in R$
- For any $(q_1, q_2) \in R$ and for any $l_2 \in L_2$, there exists $l_1 \in L_1$ s.t. the existence of $x_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 with $(p_1, p_2) \in R$

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

From [Alur et al., 1998] strategies designed for T_1 can be appropriately transferred to T_2 if the LTSs are alternatingly bisimilar.

Dealing with non-determinism: alternating bisimulation

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

- $H_1(q_1) = H_2(q_2)$
- For any $(q_1, q_2) \in R$ and for any $l_1 \in L_1$, there exists $l_2 \in L_2$ s.t. the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies the existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 with $(p_1, p_2) \in R$
- For any $(q_1, q_2) \in R$ and for any $l_2 \in L_2$, there exists $l_1 \in L_1$ s.t. the existence of $x_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 with $(p_1, p_2) \in R$

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

From [Alur et al., 1998] strategies designed for T_1 can be appropriately transferred to T_2 if the LTSs are alternatingly bisimilar.

Dealing with non-determinism: alternating ~~bi~~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 an ***alternating ~~bisimulation~~ relation*** ~~between~~ T_1 ~~and~~ T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- For any $(q_1, q_2) \in R$ and for any $l_1 \in L_1$, there exists $l_2 \in L_2$ s.t. the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies the existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 with $(p_1, p_2) \in R$
- ~~For any $(q_1, q_2) \in R$ and for any $l_2 \in L_2$, there exists $l_1 \in L_1$ s.t. the existence of $x_1 \xrightarrow{l_1}_1 p_1$ in T_1 implies the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 with $(p_1, p_2) \in R$~~

is ***alternatingly simulated*** by T_2 $\quad T_1 \preccurlyeq^{alt} T_2$

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

Dealing with non-determinism: alternating 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 an ***alternating simulation relation*** from T_1 to T_2 if for all $(q_1, q_2) \in R$

- $H_1(q_1) = H_2(q_2)$
- For any $(q_1, q_2) \in R$ and for any $l_1 \in L_1$, there exists $l_2 \in L_2$ s.t. the existence of $q_2 \xrightarrow{l_2}_2 p_2$ in T_2 implies the existence of $q_1 \xrightarrow{l_1}_1 p_1$ in T_1 with $(p_1, p_2) \in R$

LTS T_1 is alternatingly simulated by T_2, denoted $T_1 \preccurlyeq^{alt} T_2$, if $\pi|_{Q_1}(R) = Q_1$

Dealing with non-determinism: alternating simulation

Given $T_1 = (Q_1, A_1 \times B_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, A_2 \times B_2, \longrightarrow_2, O_2, H_2)$ with $O_1 = O_2$, a relation

$$R \subseteq Q_1 \times Q_2$$

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

- $H_1(q_1) = H_2(q_2)$
- for any $a_1 \in A_1$ there exists $a_2 \in A_2$ s.t. for any $b_2 \in B_2$ there exists $b_1 \in B_1$ s.t.

  $q_1 \xrightarrow{(a_1,b_1)}_1 p_1$ in T_1 and $q_2 \xrightarrow{(a_2,b_2)}_2 p_2$ in T_2 with $(p_1, p_2) \in R$.

LTS T_1 is alternatingly simulated by T_2, denoted $T_1 \preccurlyeq^{alt} T_2$, if $\pi|_{Q_1}(R) = Q_1$

In the previous definition, the non-determinism is parametrized by disturbance labels.

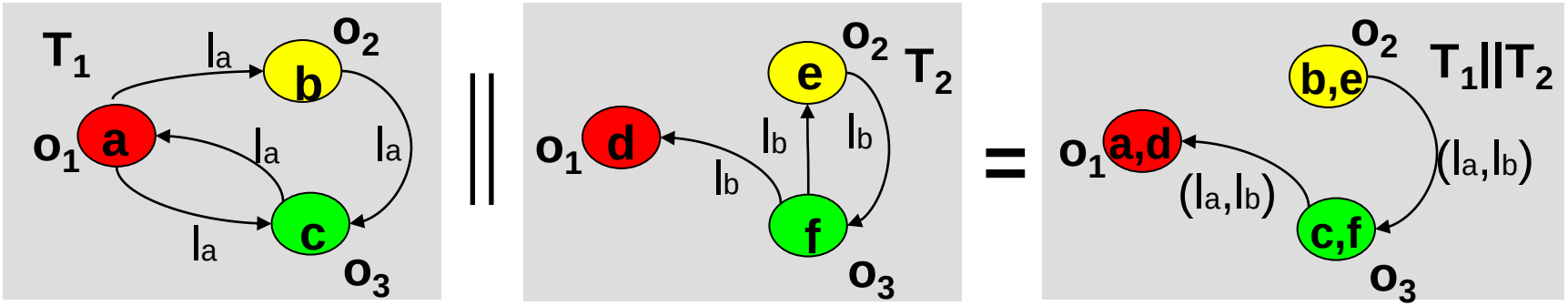
Discrete control synthesis: parallel composition

Definition Given $T_1 = (Q_1, Q_{01}, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, Q_{02}, L_2, \longrightarrow_2, O_2, H_2)$, with $O_1 = O_2$, the composition of T_1 and T_2 is the LTS

$$T = T_1 || T_2 = (Q, Q_0, L, \longrightarrow, O, H)$$

where:

- $Q = \{(q_1, q_2) \in Q_1 \times Q_2 : H_1(q_1) = H_2(q_2)\}$
- $Q_0 = Q \cap (Q_{01} \times Q_{02})$
- $L = L_1 \times L_2$
- $(q_1, q_2) \xrightarrow{(l_1, l_2)} (p_1, p_2)$, if $q_1 \xrightarrow{l_1} p_1$ and $q_2 \xrightarrow{l_2} p_2$
- $O = O_1 = O_2$
- $H(q_1, q_2) = H_1(q_1)$



Discrete control synthesis: safety games

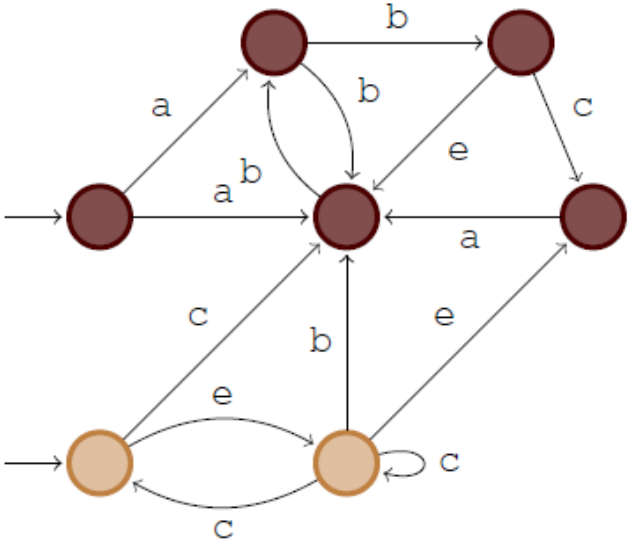
Safety game

Given a transition system $T=(Q,Q_0,L,\longrightarrow,O,H)$ with $O=Q$ and a set of safe outputs $W \subseteq Q$, find a transition system C (controller) such that

1. $L(T||C) \subseteq W^{\omega}$;
2. $T||C$ is non-blocking.

W^{ω} is the set of all the infinite words that are generated from W.

Several concrete requirements can be formulated as safety specifications: avoid buffer overflows, avoid collision with obstacles, avoid leaving operational regions, etc.



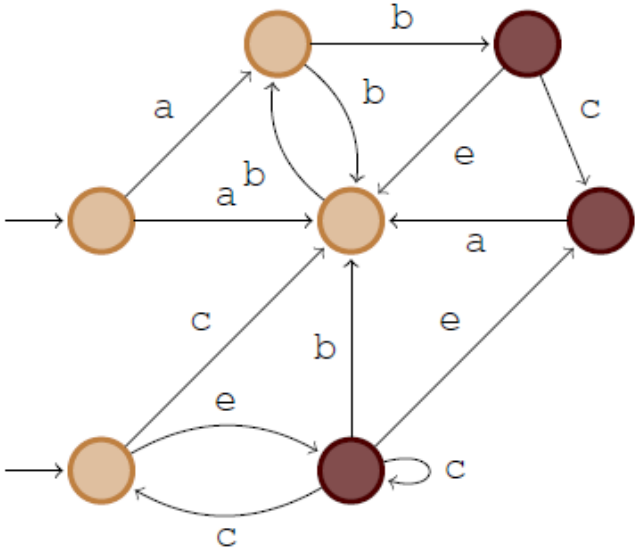
Discrete control synthesis: reachability games

Reachability game

Given a transition system $T=(Q,Q_0,L,\longrightarrow,O,H)$ with $O=Q$ and a target set $W\subseteq Q$, find a transition system C (controller) such that for every maximal output run $y\in L^{\omega}(T)$ there exists a time k such that $y(k)\in W$.

A *maximal output run* is an output trajectory of T that cannot be further extended.

Note that a maximal output run *can have a finite length* because the LTS can be blocking.

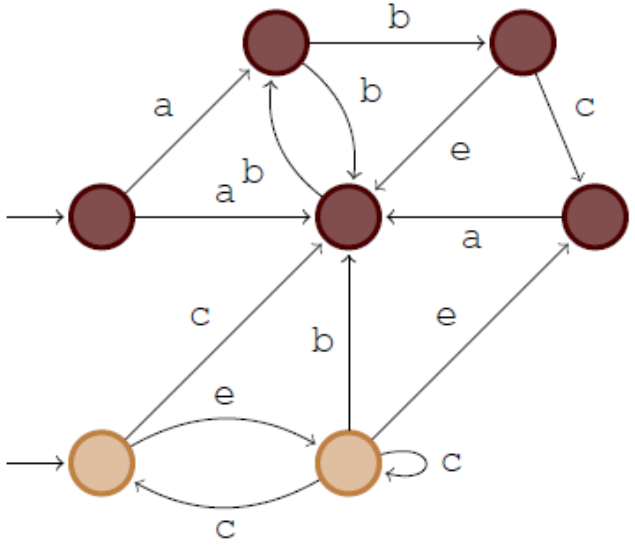


Discrete control synthesis: similarity games

Similarity game

Given a transition system P (plant) and a transition system Q (specificaton), find a transition system C (controller) such that

1.P||C ≼Q;
2.P||C is non-blocking.



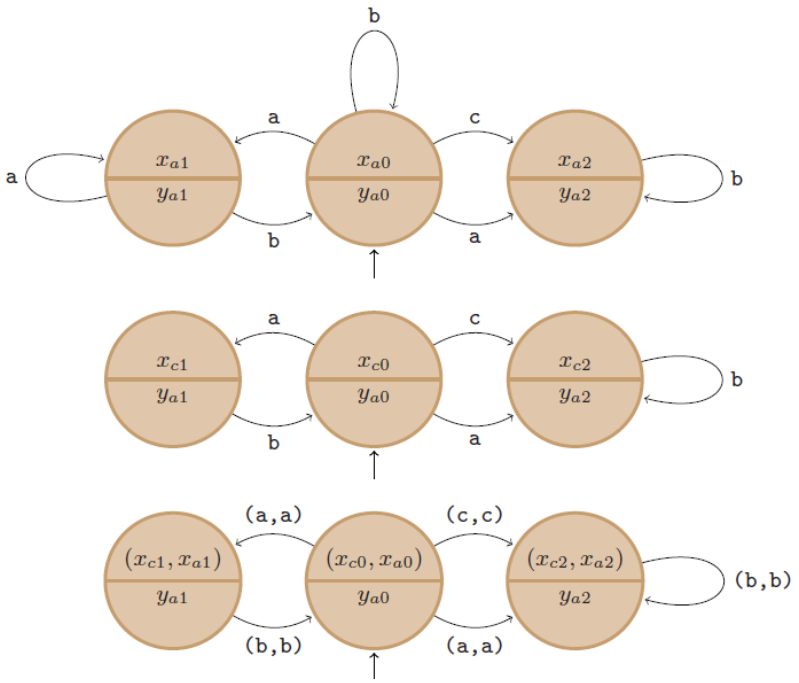
Robust control synthesis: feedback composition

Definition Given $T_1 = (Q_1, Q_{01}, L_1, \longrightarrow_1, O_1, H_1)$ and $T_2 = (Q_2, Q_{02}, L_2, \longrightarrow_2, O_2, H_2)$, with $O_1 = O_2$, let R be an ***alternating simulation relation*** from T_2 to T_1. The ***feedback composition*** of T_1 and T_2 is the LTS

$$T = T_1 ||^R T_2 = (Q, Q_0, L, \longrightarrow, O, H)$$

where:

- $Q = R^{-1}$
- $Q_0 = Q \cap (Q_{01} \times Q_{02})$
- $L = L_1 \times L_2$
- $(q_1, q_2) \xrightarrow{(l_1,l_2)} (p_1, p_2)$, if $q_1 \xrightarrow{l_1} p_1$ and $q_2 \xrightarrow{l_2} p_2$
- $O = O_1 = O_2$
- $H(q_1, q_2) = H_1(q_1)$



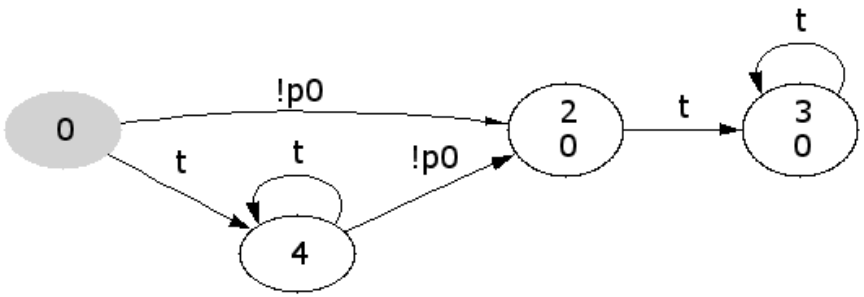
Safety and similarity games are rewritten and solved robustly in the presence of non-determinism by replacing the parallel composition || with the feedback composition $||^R$.

Linear Temporal Logic (LTL)

- **Linear (or Linear-time) Temporal Logic (LTL)** is a widely used formalism for specifying and verifying properties of reactive systems.

- A property is expressed by an **LTL formula** describing the set of all the infinite state/output sequences fulfilling the temporal/logic property.

- An LTL formula can be encoded into a **Buchi Automaton**, which is a more general model than a transition system. The previous concepts of relations, compositions, verification and control problems can be properly extended to Buchi Automata.



Linear Temporal Logic (LTL)

An LTL formula is built up from :

- a finite set of Atomic Propositions (e.g.: the state labels of a transition system)
- Logical Operators:

  ¬ (negation) true

  ∧ (conjunction) false

  ∨ (disjunction) → (implication)

  ↔ (equivalence)
- Temporal Operators:

  O (next)

  U (until)

  G (always)

  F (eventually)

By combining formulas by means of temporal/logical operators, you get other formulas.