# Embedded Systems - #9

## Symbolic Control

Maria Domenica Di Benedetto
Giordano Pola
Alessandro Borri

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

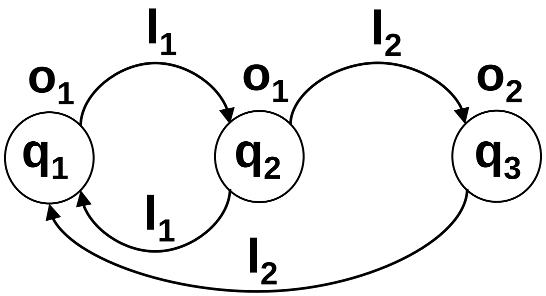
# Review: Transition Systems

**Definition** A transition 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 $\longrightarrow \subseteq Q \times L \times Q$
- an output set $O$
- an output function $H: Q \to O$




- $T$ is said countable if $Q$ and $L$ are countable sets
- $T$ is said symbolic/finite if $Q$ and $L$ are finite sets
- $T$ is non-blocking if for $q \in Q$ there exists $l \in L$ and $q' \in Q$ such that $(q,l,q') \in \longrightarrow$
- $T$ is deterministic if for any $q \in Q$ and any $l \in L$ there exists at most one state $q' \in Q$ s.t. $(q,l,q') \in \longrightarrow$

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

# Review: Transition Systems modeling Control Systems

A **nonlinear control system Σ**

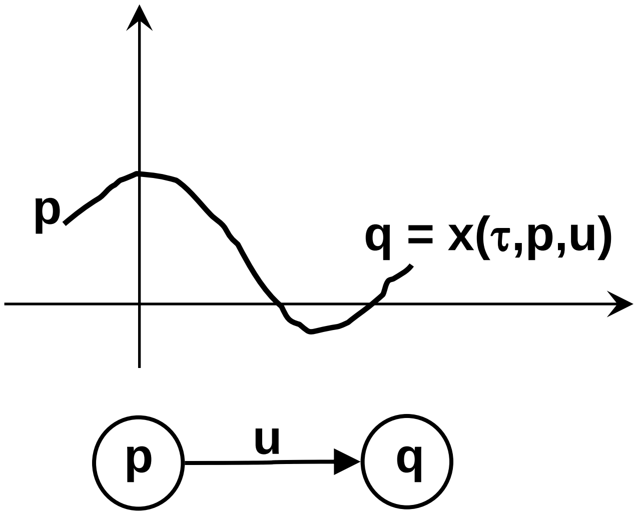
$$dx/dt = f(x,u),\ x \in X \subseteq R^n,\ u \in U \subseteq R^m$$

can be modeled by the transition system

$$T(\Sigma) = (X, X_0, \mathcal{U}, \longrightarrow, Y, H),$$

where:

- $X_0 = X$ is a set of initial states
- $\mathcal{U}$ is the collection of control signals $u : R \to U$
- $p \xrightarrow{u} q$, if $x(\tau,p,u) = q$ for some $\tau \geq 0$
- $Y = X$
- $H$ is the identity function




$T(\Sigma)$ captures information contained in $\Sigma$ but it is not a symbolic model because $X$ and $U$ are infinite sets!

# Review: Transition Systems modeling Control Systems

A **nonlinear control system Σ**

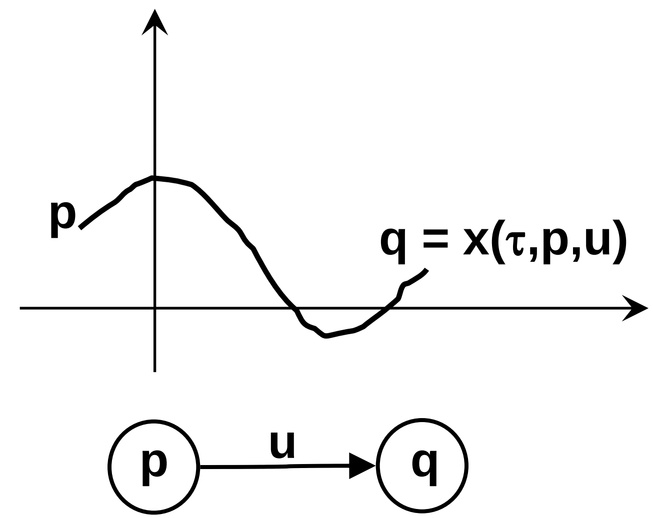
$$dx/dt = f(x,u),\ x \in X \subseteq R^n,\ u \in U \subseteq R^m$$

can be modeled by the transition system

$$T(\Sigma) = (X, X_0, \mathcal{U}, \longrightarrow, Y, H),$$

where:

- $X_0 = X$ is a set of initial states
- $\mathcal{U}$ is the collection of control signals $u : R \to U$
- $p \xrightarrow{u} q$, if $x(\tau,p,u) = q$ for some $\tau \geq 0$
- $Y = X$
- $H$ is the identity function




$T(\Sigma)$ captures information contained in $\Sigma$ but it is not a symbolic model because $X$ and $U$ are infinite sets!

**Software can be modelled by transition systems**

The states are all possible memory configurations and the transition relation describes how the memory contents are changed by the execution of instructions

# Review: Construction of symbolic models

We consider digital control systems, i.e. control systems where input signals are piecewise constant.

Consider a nonlinear digital control system

$$T(\Sigma) = (X, X_0, \mathcal{U}, \longrightarrow, Y, H),$$

and given some $\tau > 0$, define the transition system

$$T_\tau(\Sigma) = (X, X_0, \mathcal{U}_\tau, \longrightarrow_\tau, Y, H),$$

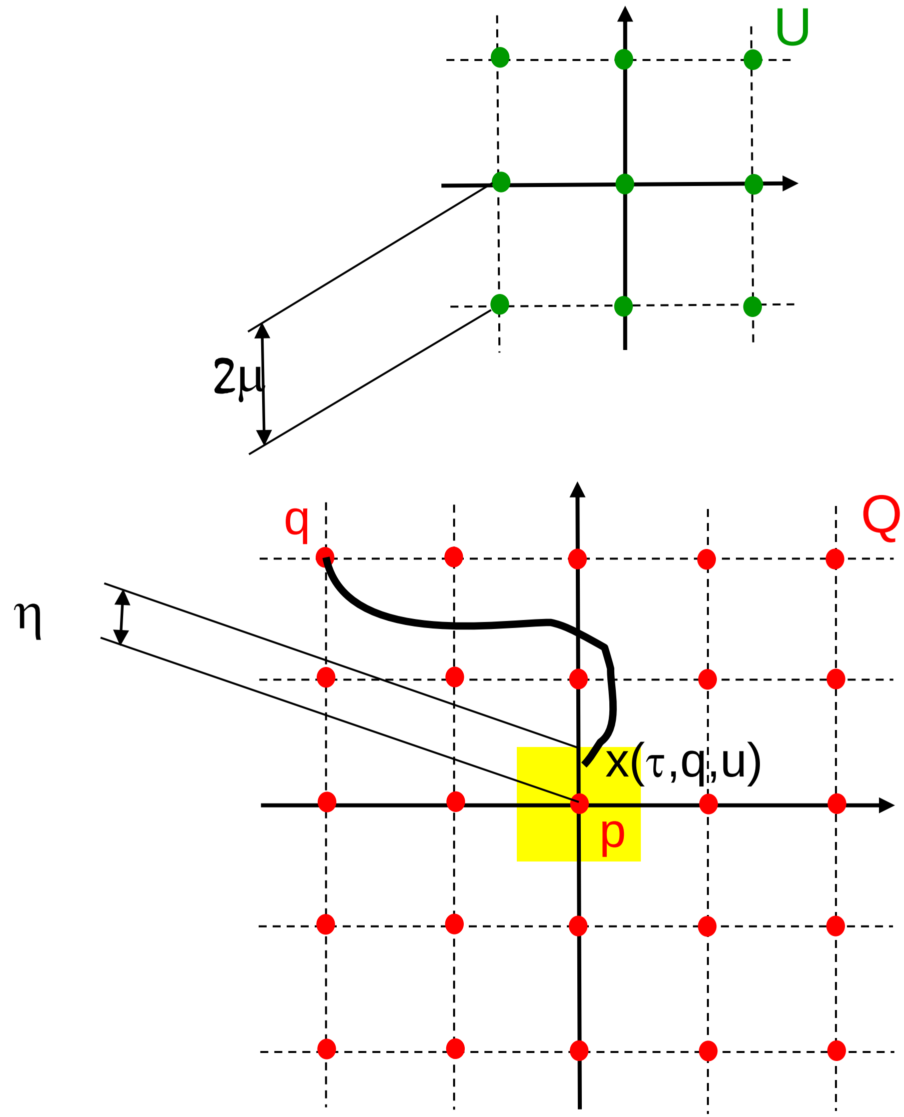
where:

- $\mathcal{U}_\tau$ is the collection of constant functions $u : [0,\tau] \rightarrow R^m$
- $p \xrightarrow{u}_\tau q$ if $x(\tau,p,u) = q$

# Review: Construction of symbolic models

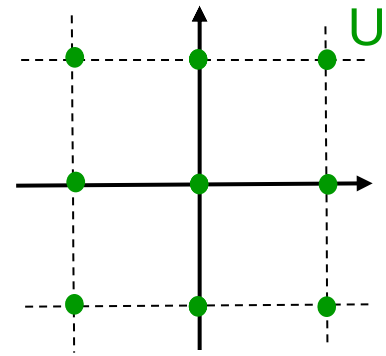
Consider the following parameters:

- $\tau > 0$ sampling time
- $\eta > 0$ state space quantization
- $\mu > 0$ input space quantization



# Review: Construction of symbolic models

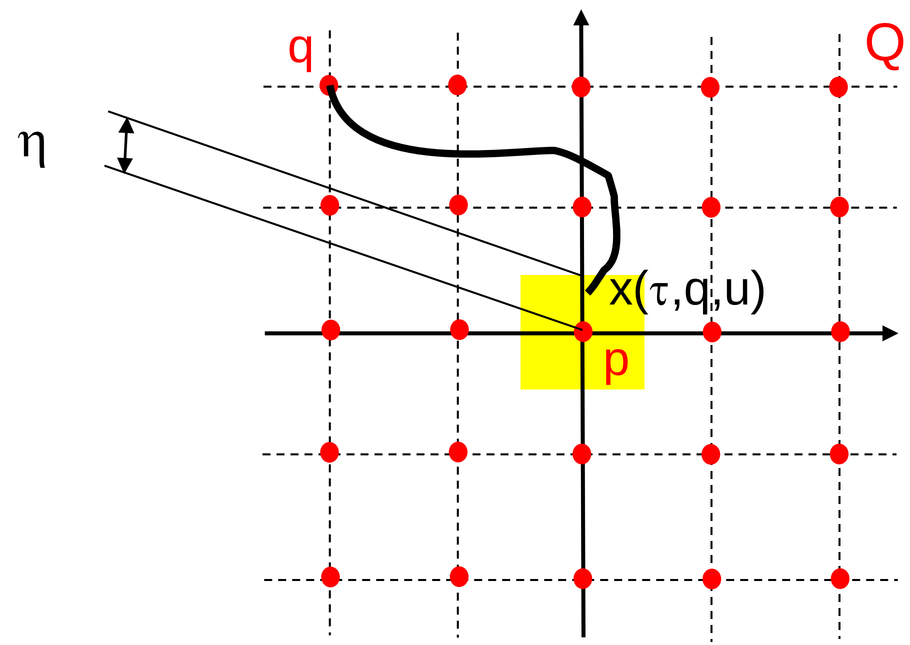
Consider the following parameters:




- $\tau > 0$ sampling time
- $\eta > 0$ state space quantization
- $\mu > 0$ input space quantization

and define $T_{\tau,\eta,\mu}(\Sigma) = (X_{\tau,\eta,\mu}, X_{0,\tau,\eta,\mu}, U_{\tau,\eta,\mu}, \longrightarrow_{\tau,\eta,\mu}, Y, H)$, where:

- $X_{\tau,\eta,\mu} = [X]_{2\eta}$
- $X_{0,\tau,\eta,\mu} = X_{\tau,\eta,\mu} \cap X_0$
- $U_{\tau,\eta,\mu} = [U]_{2\mu}$
- $p \xrightarrow{u}_{\tau,\eta,\mu} q$, if $\| x(\tau,p,u) - q \| \leq \eta$
- $Y = X$
- $H$ is the identity function



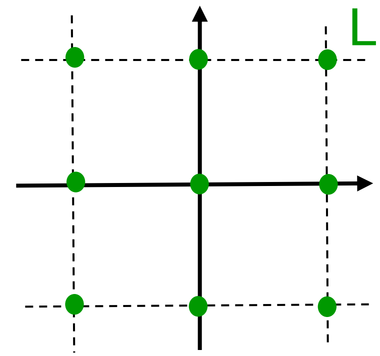

**Remark**

Transition system $T_{\tau,\eta,\mu}(\Sigma)$ is countable.
If state and input spaces of $\Sigma$ are bounded then $T_{\tau,\eta,\mu}(\Sigma)$ is symbolic!

# Review: Construction of symbolic models

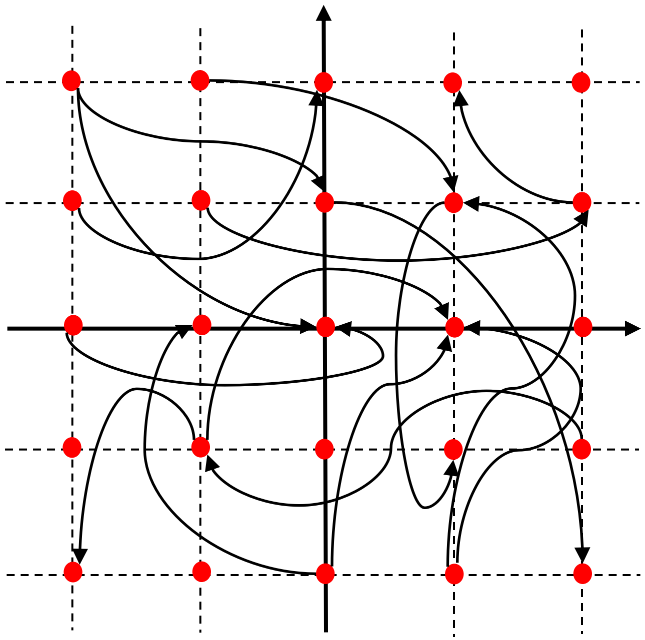
Consider the following parameters:

- $\tau > 0$ sampling time
- $\eta > 0$ state space quantization
- $\mu > 0$ input space quantization




and define $T_{\tau,\eta,\mu}(\Sigma) = (X_{\tau,\eta,\mu}, X_{0,\tau,\eta,\mu}, U_{\tau,\eta,\mu}, \longrightarrow_{\tau,\eta,\mu}, Y, H)$, where:

- $X_{\tau,\eta,\mu} = [X]_{2\eta}$
- $X_{0,\tau,\eta,\mu} = X_{\tau,\eta,\mu} \cap X_0$
- $U_{\tau,\eta,\mu} = [U]_{2\mu}$
- $p \xrightarrow{u}_{\tau,\eta,\mu} q$, if $\| x(\tau,p,u) - q \| \leq \eta$
- $Y = X$
- $H$ is the identity function



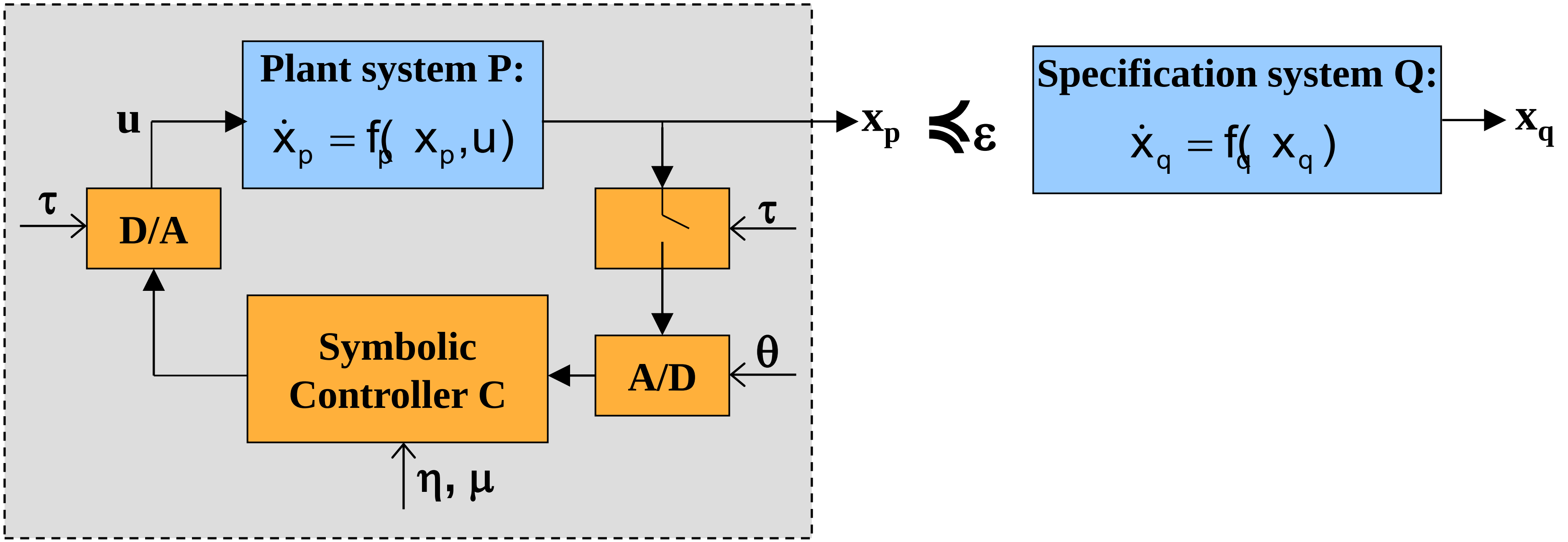
**Theorem** If $\Sigma$ is $\delta$-ISS, for any desired precision $\varepsilon > 0$ and for any $\tau, \eta, \mu > 0$ satisfying

$$\beta(\varepsilon,\tau) + \eta + \gamma(\mu) \leq \varepsilon$$

then $T_\tau(\Sigma)$ and $T_{\tau,\eta,\mu}(\Sigma)$ are $\varepsilon$-bisimilar

## Problem 1: Continuous Specifications [cf. Borri, Pola, Di Benedetto, CDC 2010]

Given a plant P, a specification Q and a desired precision $\varepsilon > 0$, find a symbolic controller that implements Q up to the precision $\varepsilon$ and that is non-blocking when interacting with P.



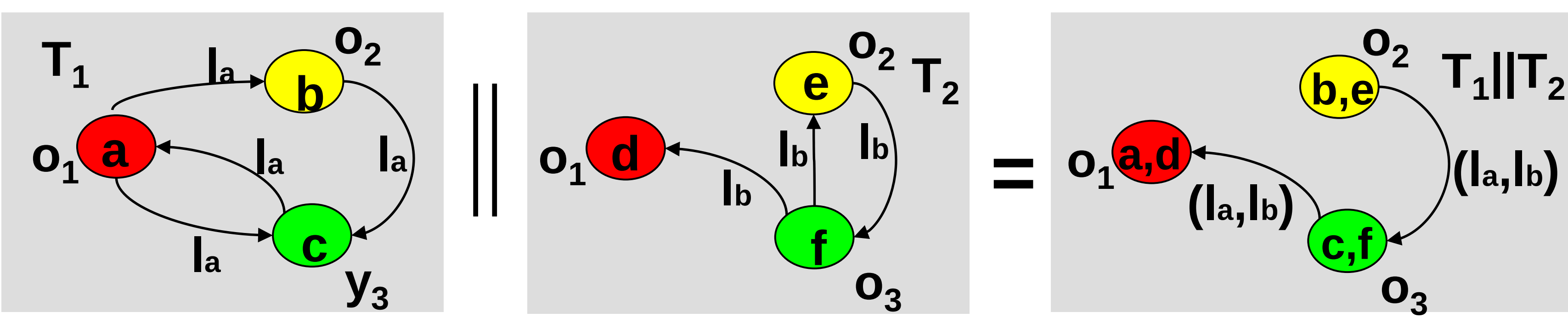
# Approximate composition [Tabuada IEEE TAC 08]

**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 $O_1 = O_2$, and a precision $\theta > 0$, the approximate composition of $T_1$ and $T_2$ is the system

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

where:

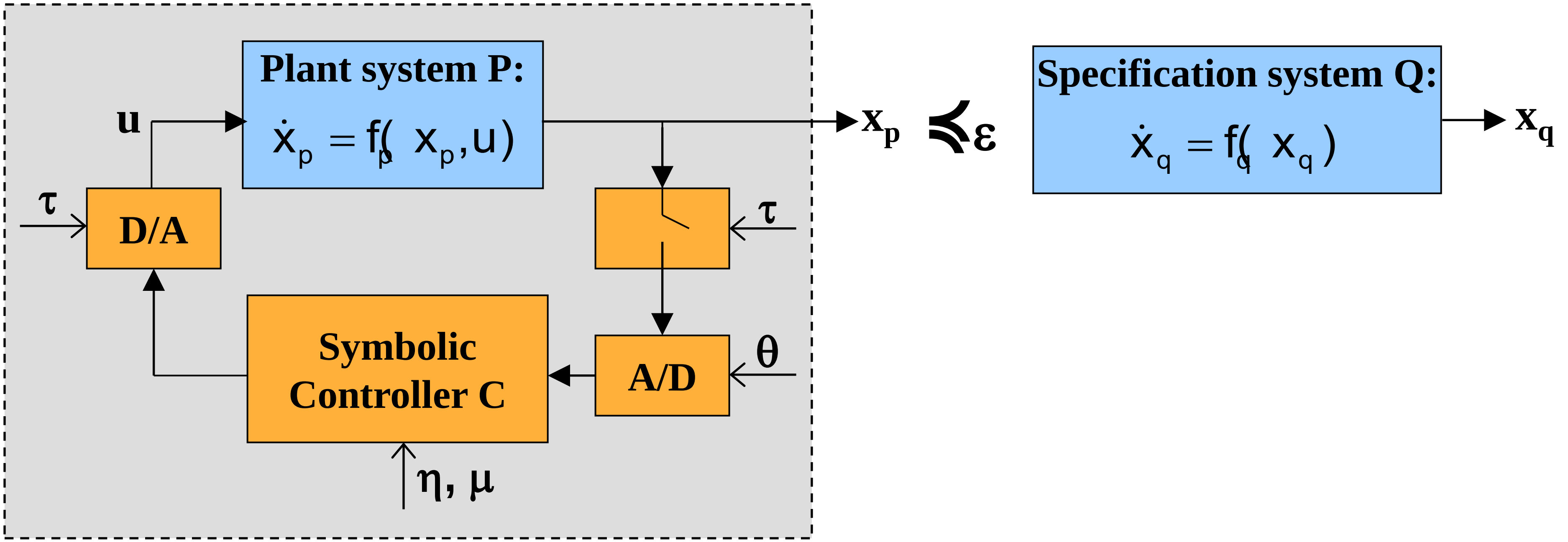
- $Q = \{(q_1, q_2) \in Q_1 \times Q_2 : d(H_1(q_1), H_2(q_2)) \leq \theta\}$
- $Q_0 = Q \cap (Q_{01} \times Q_{02})$
- $L = L_1 \times L_2$
- $(q_1,q_2) \xrightarrow{l_1,l_2} (p_1,p_2)$, if $q_1 \xrightarrow{l_1} p_1$ and $q_2 \xrightarrow{l_2} p_2$
- $O = O_1 = O_2$
- $H(q_1,q_2) = H_1(q_1)$



**Problem 1**

Given a plant P, a specification Q and a desired precision $\varepsilon > 0$, find a symbolic controller C such that

1. $T_{\tau,\eta,\mu}(P)\|_{\theta}C \preccurlyeq_{\varepsilon} T_{\tau,\eta,0}(Q)$
2. $T_{\tau,\eta,\mu}(P)\|_{\theta}C$ is non-blocking



# Solution of Problem 1

Synthesis through a four-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of Q
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P)||T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part Nb(C*) of C*

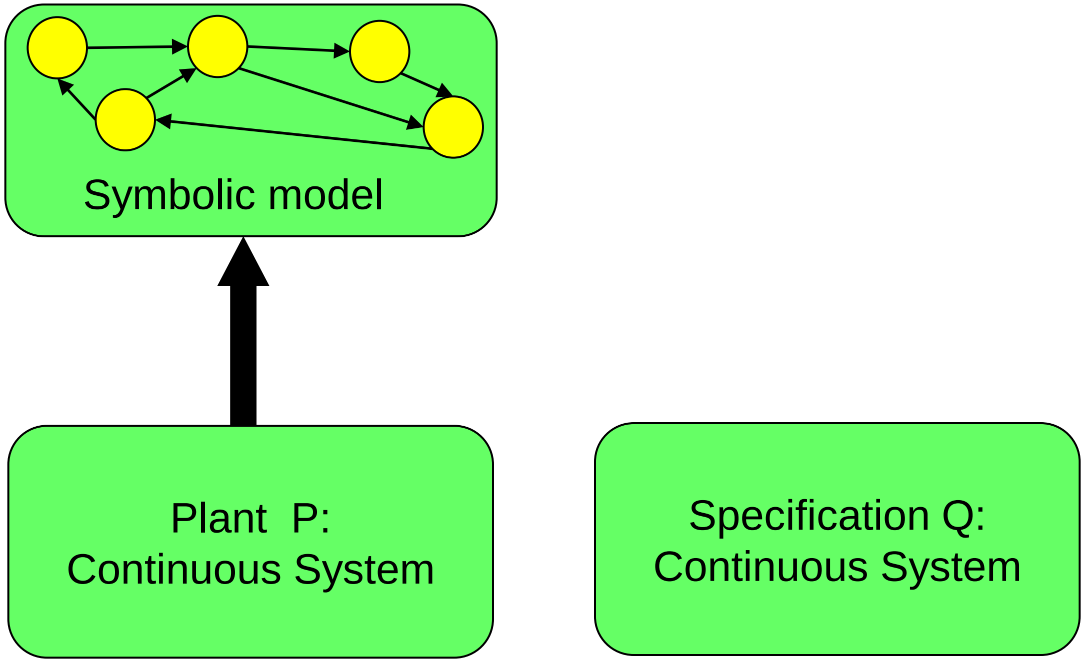
Plant P:
Continuous System

Specification Q:
Continuous System

# Solution of Problem 1

Synthesis through a four-step process:

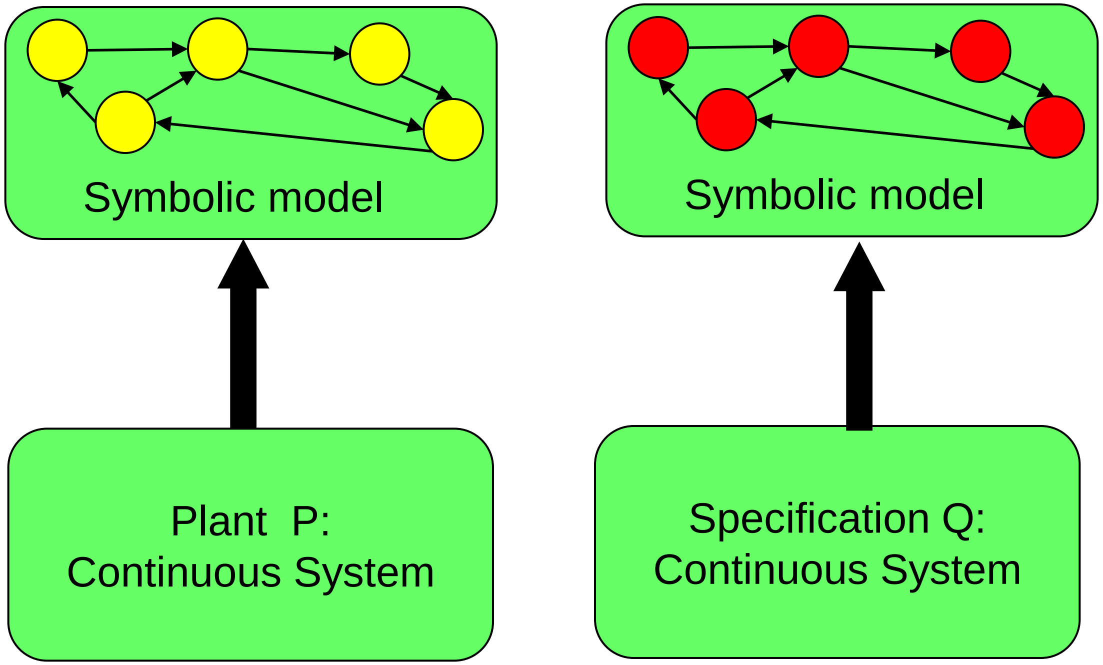
1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of Q
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P)||T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 1

Synthesis through a four-step process:

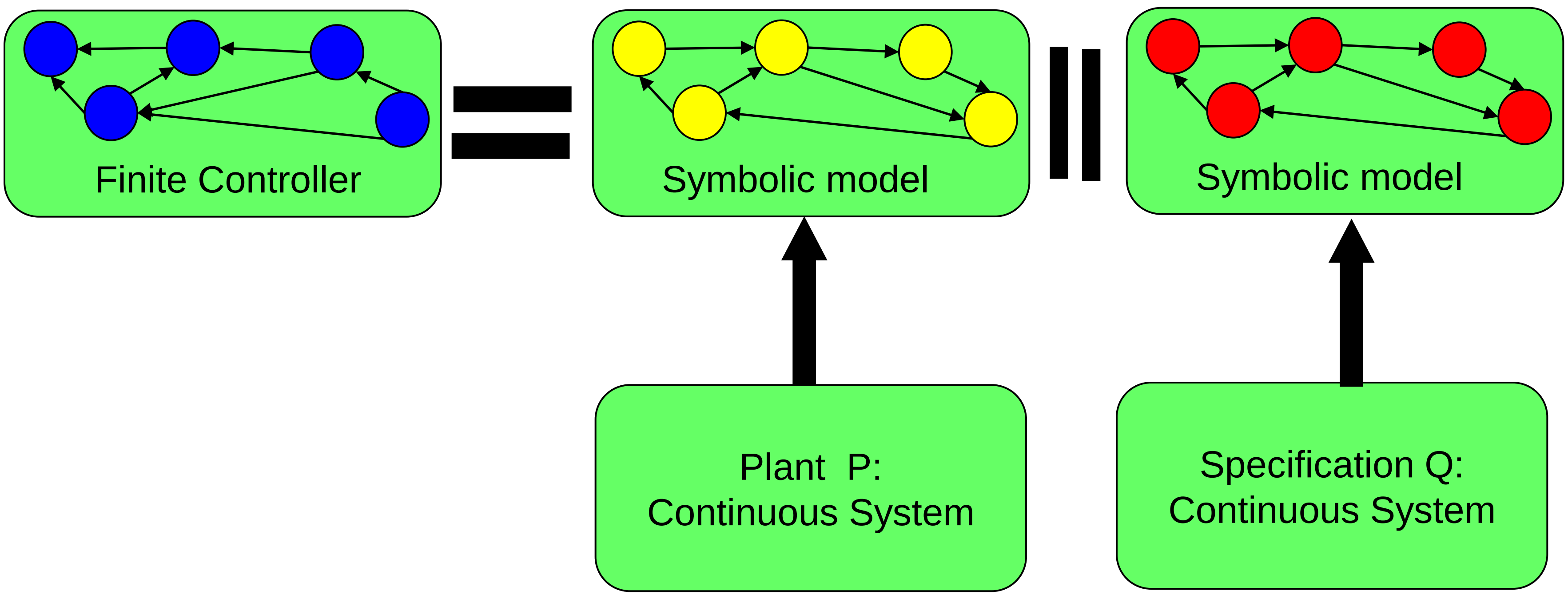
1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of Q
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P)||T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part $Nb(C^*)$ of $C^*$



# Solution of Problem 1

Synthesis through a four-step process:

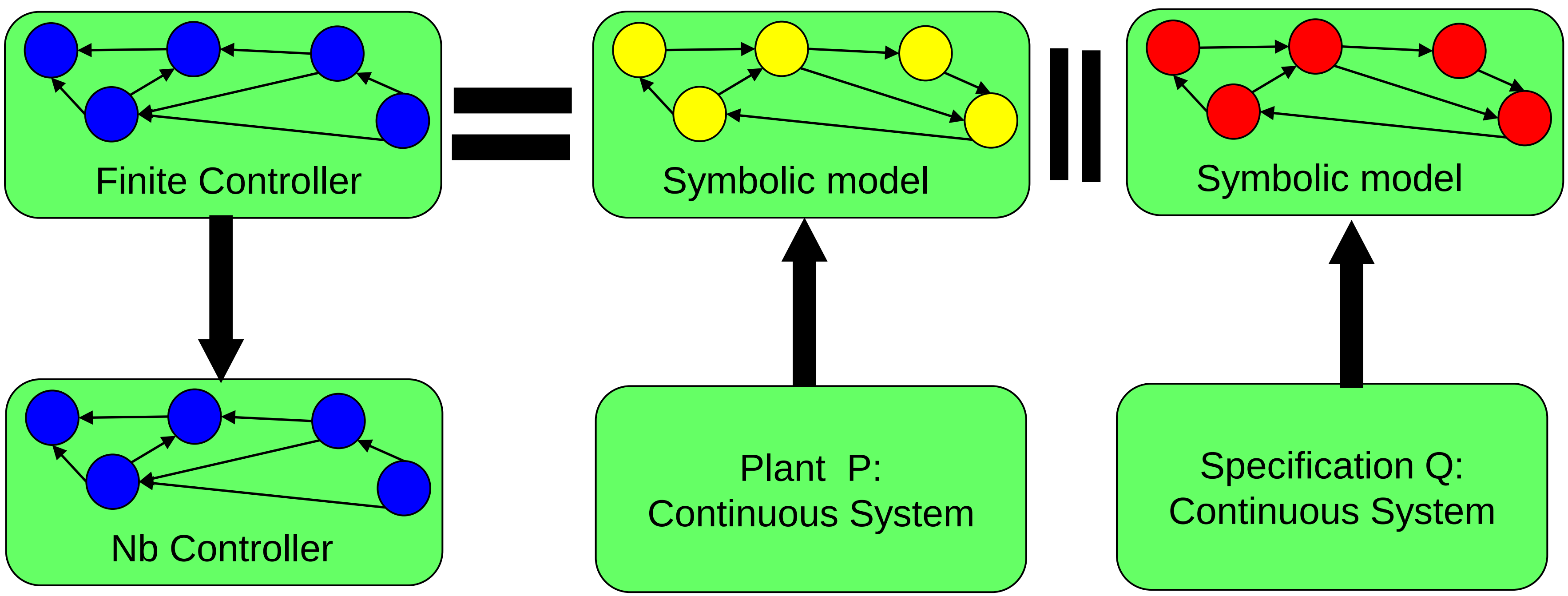
1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of Q
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) || T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 1

Synthesis through a four-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of $P$
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of $Q$
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) || T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 1

Synthesis through a four-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of $P$
2. Compute the symbolic model $T_{\tau,\eta,0}(Q)$ of $Q$
3. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) || T_{\tau,\eta,0}(Q)$
4. Compute the non-blocking part $Nb(C^*)$ of $C^*$

**Theorem** Suppose that $P$ and $Q$ are $\delta$–ISS and choose parameters $\varepsilon_p, \varepsilon_q > 0$ so that

(1) $$\varepsilon_p + \varepsilon_q \leq \varepsilon$$

Choose parameters $\tau, \eta, \mu > 0$ satisfying

(2) $$\beta_p(\varepsilon_p,\tau) + \eta + \gamma_p(\mu) \leq \varepsilon_p$$

(3) $$\beta_q(\varepsilon_q,\tau) + \eta \leq \varepsilon_q$$

The symbolic controller $Nb(C^*)$ solves Problem 1 with $\theta = \varepsilon_p$

# Design of symbolic controllers

**Drawbacks**

- It considers the whole sets of states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$
- For any source state x and target state y, it includes all transitions $x \xrightarrow{u} y$ with any control input u by which state x reaches state y
- It first constructs $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$, then C*, to finally eliminate blocking states from C*

To cope with space and time complexity, instead of computing separately

(1) Discrete abstraction $T_{\tau,\eta,\mu}(P)$ of P
(2) Discrete abstraction $T_{\tau,\eta,0}(Q)$ of Q
(3) Symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \| T_{\tau,\eta,0}(Q)$
(4) Non-blocking part Nb(C) of C*

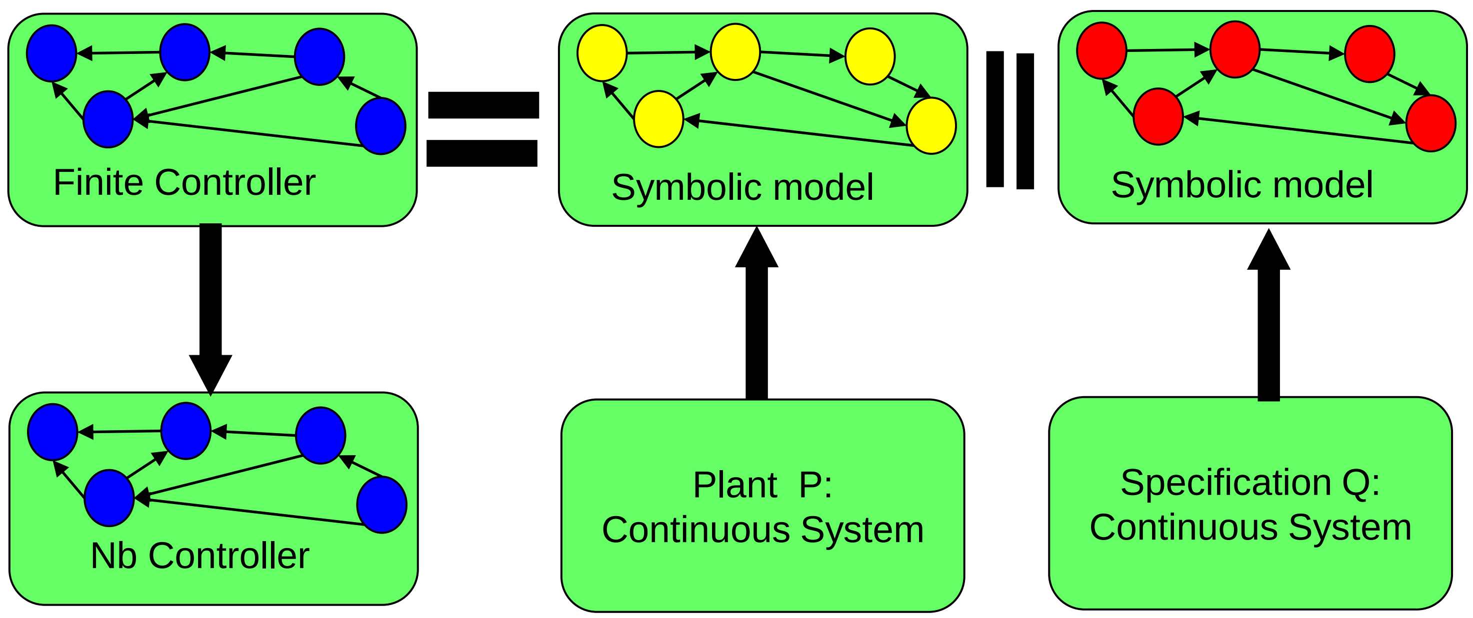
**Integrated Approach: Compute (1) + (2) + (3) + (4) at once!**

Space/time complexity analysis of the proposed algorithm formally quantifies the gain of the integrated approach

# Integrated Algorithm for Problem 1

**Basic ideas**

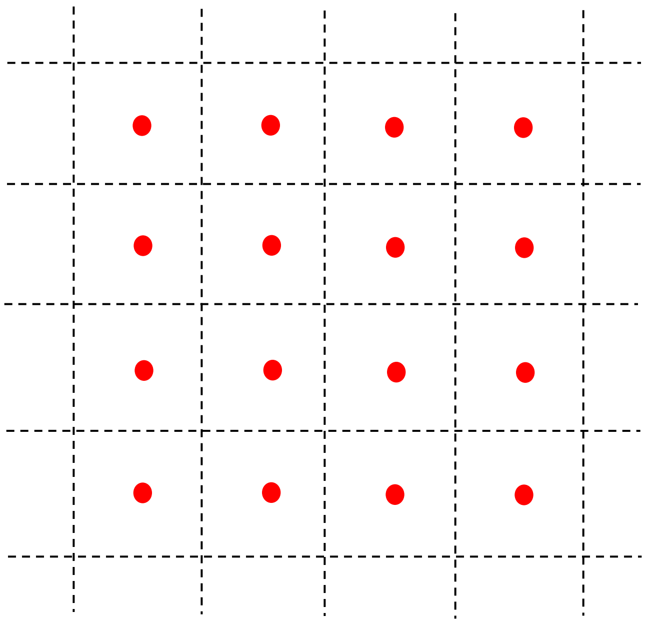
1. It only considers the intersection of the accessible parts of P and Q
2. For any given source state x and target state y, it considers only one transition (x,u,y)
3. It eliminates blocking states as soon as show up



# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

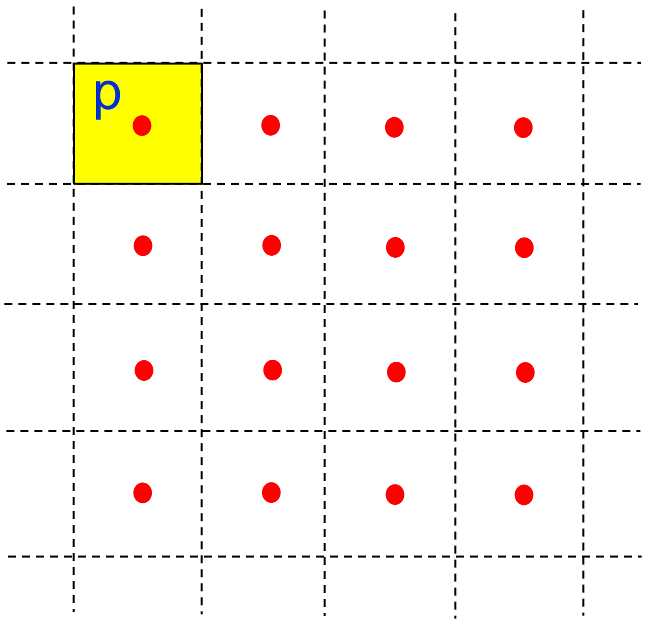


# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state p from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

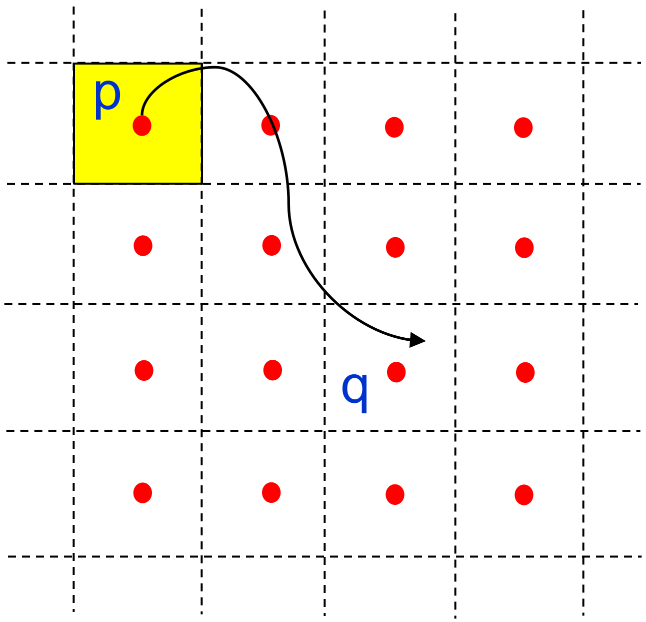


# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state p from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

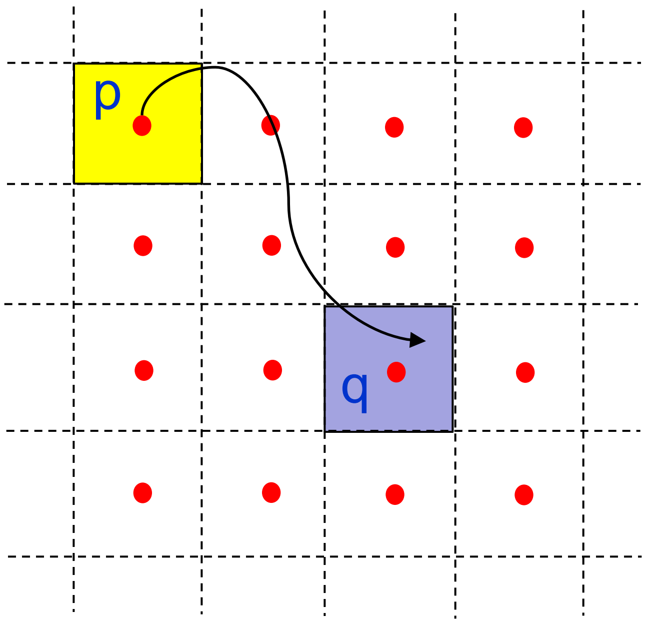


# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state p from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation



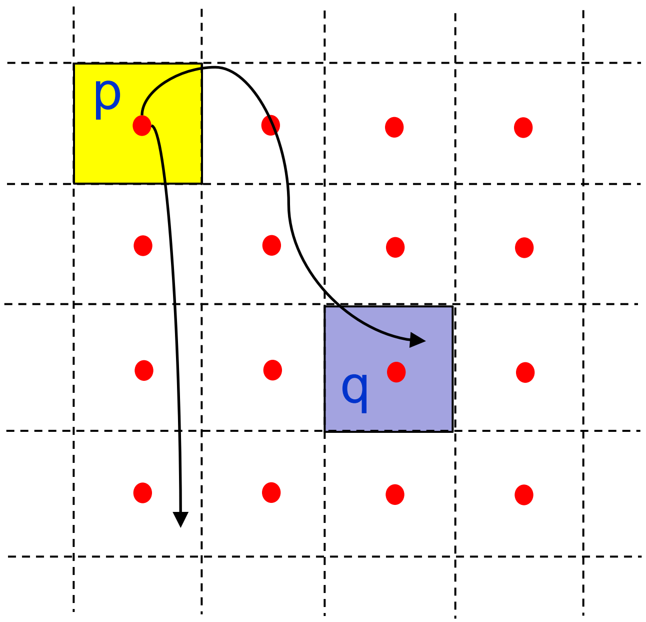
# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state $p$ from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some $u$.



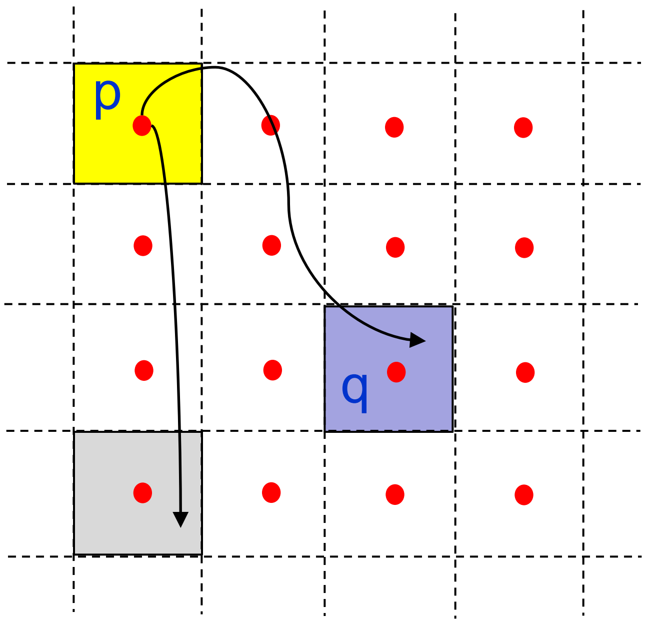
# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state $p$ from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some $u$.




No matching! Try another input!

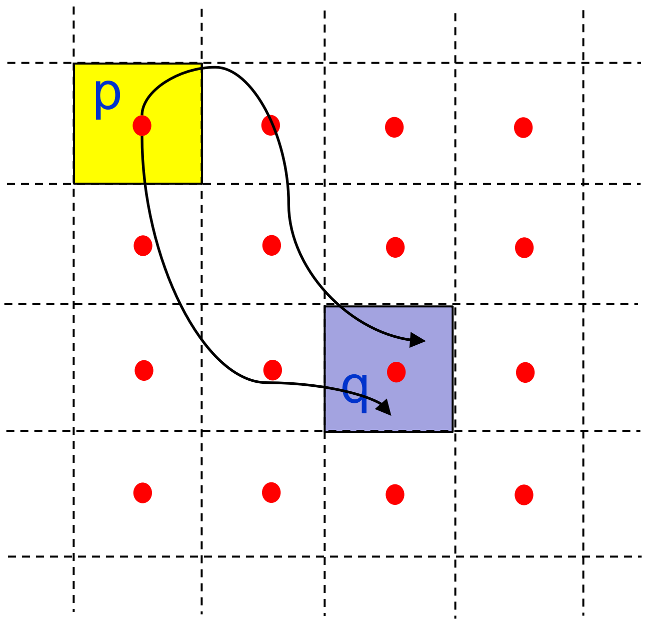
# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state $p$ from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some $u$.

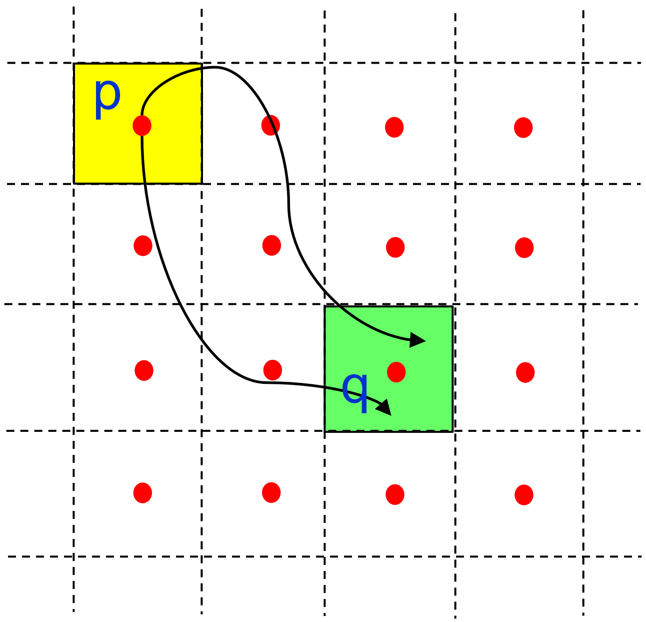


**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state $p$ from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some $u$.




Matching found!!

# Integrated Algorithm for Problem 1

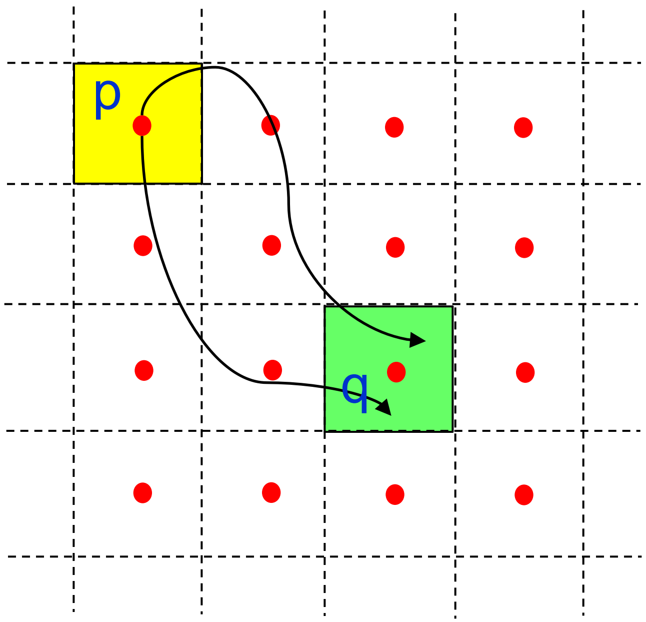
**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state p from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some u.

Add the transition (p,u,q) to the controller




Matching found!!

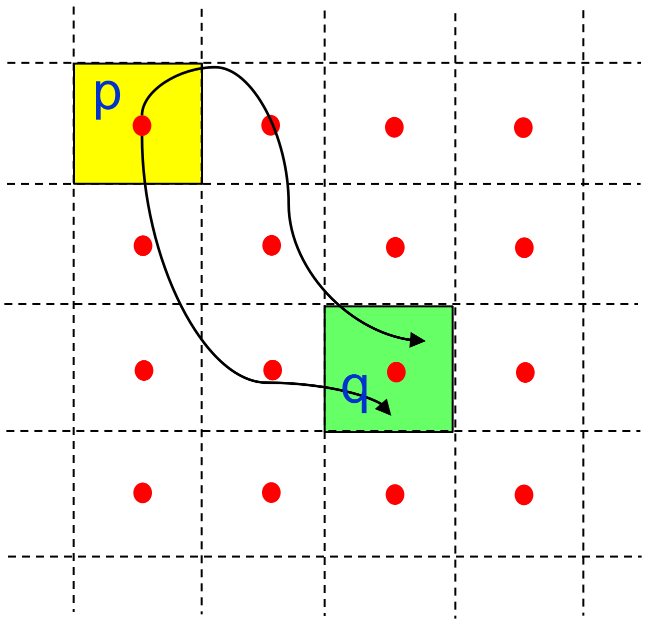
# Integrated Algorithm for Problem 1

**How does it work?**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $T_{\tau,\eta,0}(Q)$.

Pick a "symbolic" state p from the target space and compute $[x_q(\tau,p)]_{2\eta}=q$ by integrating the specification differential equation

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x_p(\tau,p,u)]_{2\eta}$ for some u.




Add the transition (p,u,q) to the controller

If any "good" input does not exist, then p is blocking!
A backwards procedure is executed to eliminate p and all its ingoing transitions from the controller, until a controller is found which is non-blocking.

# Integrated Algorithm for Problem 1

**Properties**

Let C** be the outcome of the integrated procedure:

1. The integrated algorithm terminates in a finite number of steps
2. C** and Nb(C*) are exactly bisimilar $\Rightarrow$ C** solves Problem 1
3. C** is the minimal 0-bisimilar system of Nb(C*)
4. C** is accessible
5. space/time complexity of the integrated procedure is not larger than the one of the classical procedure

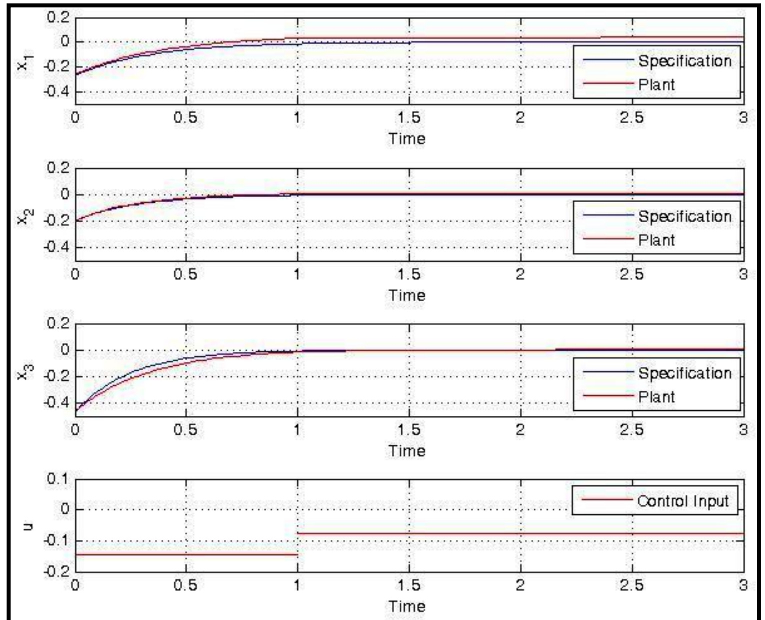
## Example 1

**Plant and Specification systems:**

$$P : \begin{cases} \dot{x}_1 = -2x_1 + x_3^2 - u \\ \dot{x}_2 = 2x_1 - 7e^{x_2} + 7 \\ \dot{x}_3 = -3x_3 + \frac{3}{4}u^2, \end{cases}$$

$$Q : \begin{cases} \dot{x}_1 = -3x_1 + x_3^3 \\ \dot{x}_2 = x_1 - 5\sin x_2 \\ \dot{x}_3 = -x_2^2 - 4x_3. \end{cases}$$

**Precision $\varepsilon = 0.2$**

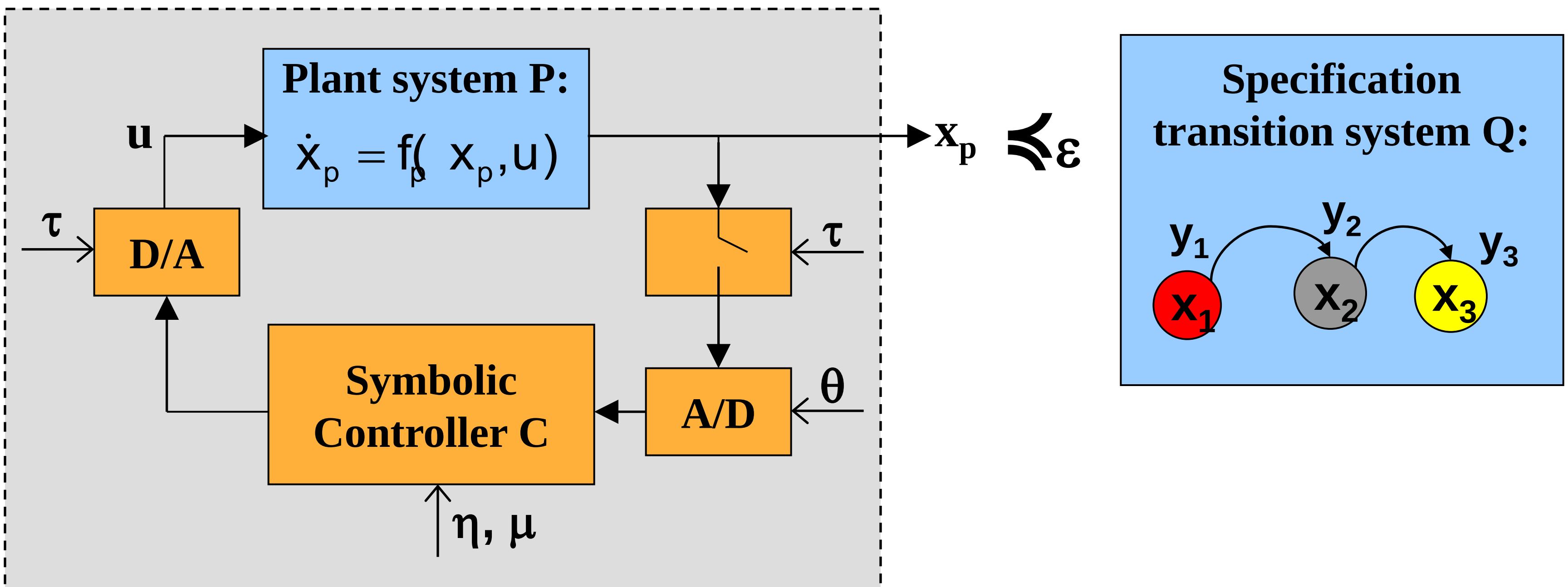



| Comparison between Nb(C*) and C** | Nb(C*) | C** | Ratio |
|---|---|---|---|
| States | 21,894 | 3,152 | 0.14 |
| Transitions | 12,652 | 3,152 | $2.5 \cdot 10^{-3}$ |
| Max memory occupation | 93,347,397 | 10,400 | $1.11 \cdot 10^{-4}$ |
| Time | 147,487 | 11,144 | 0.08 |

# Design of symbolic controllers

**Problem 2: Specifications given as deterministic transition systems [cf. Pola, Borri, Di Benedetto, IEEE TAC 12, to appear]**

Given a plant P, a deterministic specification Q and a desired precision $\varepsilon > 0$, find a symbolic controller that implements Q up to the precision $\varepsilon$ and that is non-blocking when interacting with P.

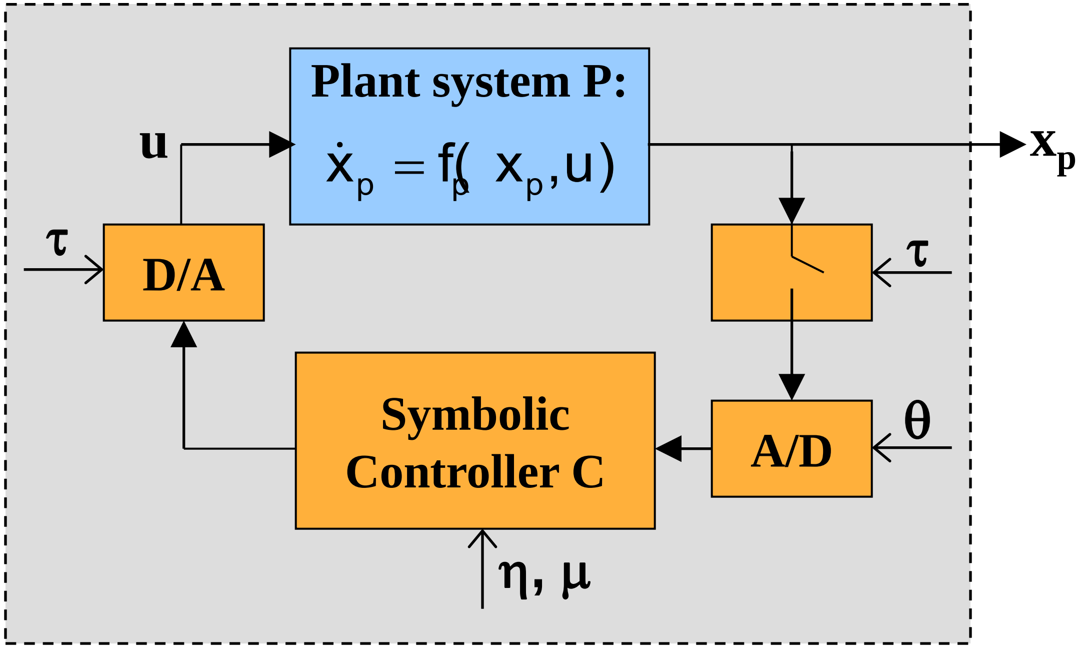


# Design of symbolic controllers

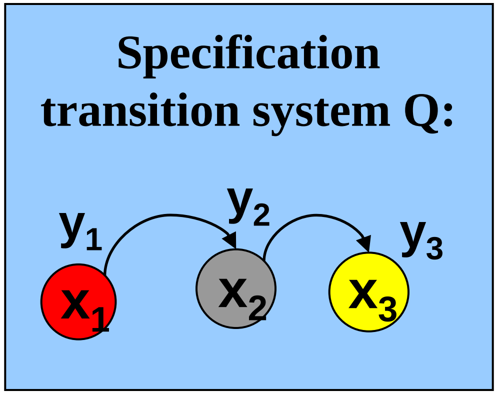
**Problem 2**

Given a plant P, a deterministic specification Q and a desired precision $\varepsilon > 0$, find a symbolic controller C such that

1. $T_{\tau,\eta,\mu}(P)||_{\theta}C \preccurlyeq_{\varepsilon} Q$
2. $T_{\tau,\eta,\mu}(P)||_{\theta}C$ is non-blocking



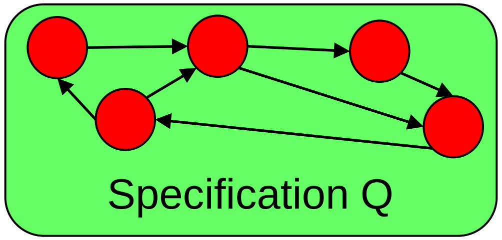

$\preccurlyeq_{\varepsilon}$



# Solution of Problem 2

Synthesis through a three-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P)||_{\eta}\ Q$
3. Compute the non-blocking part Nb(C*) of C*

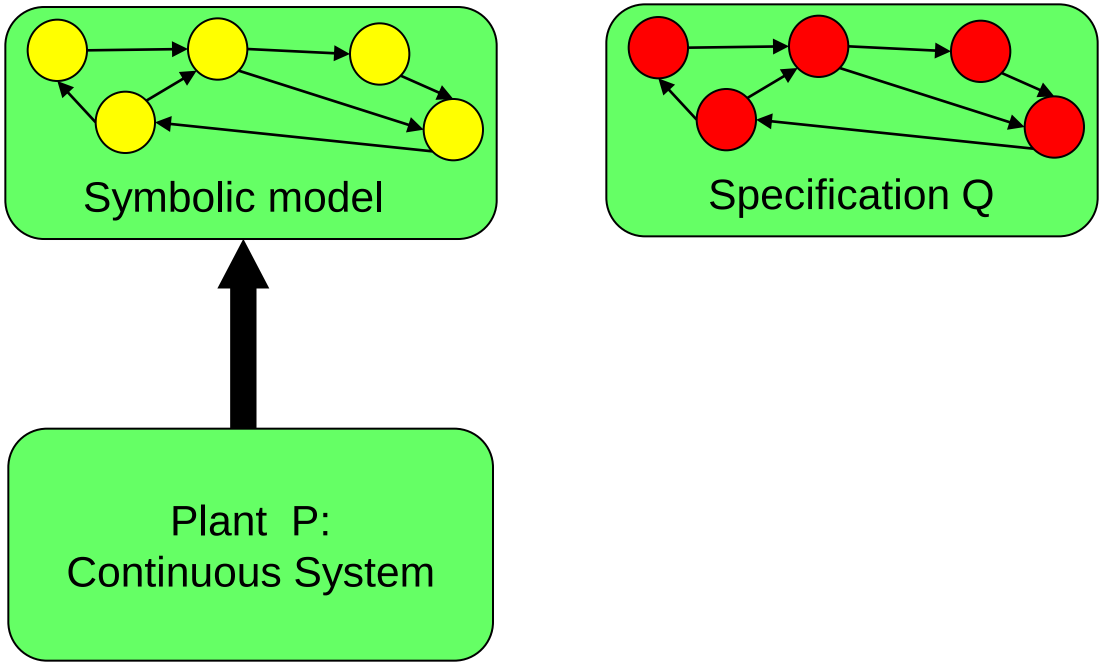



Plant P:
Continuous System

# Solution of Problem 2

Synthesis through a three-step process:

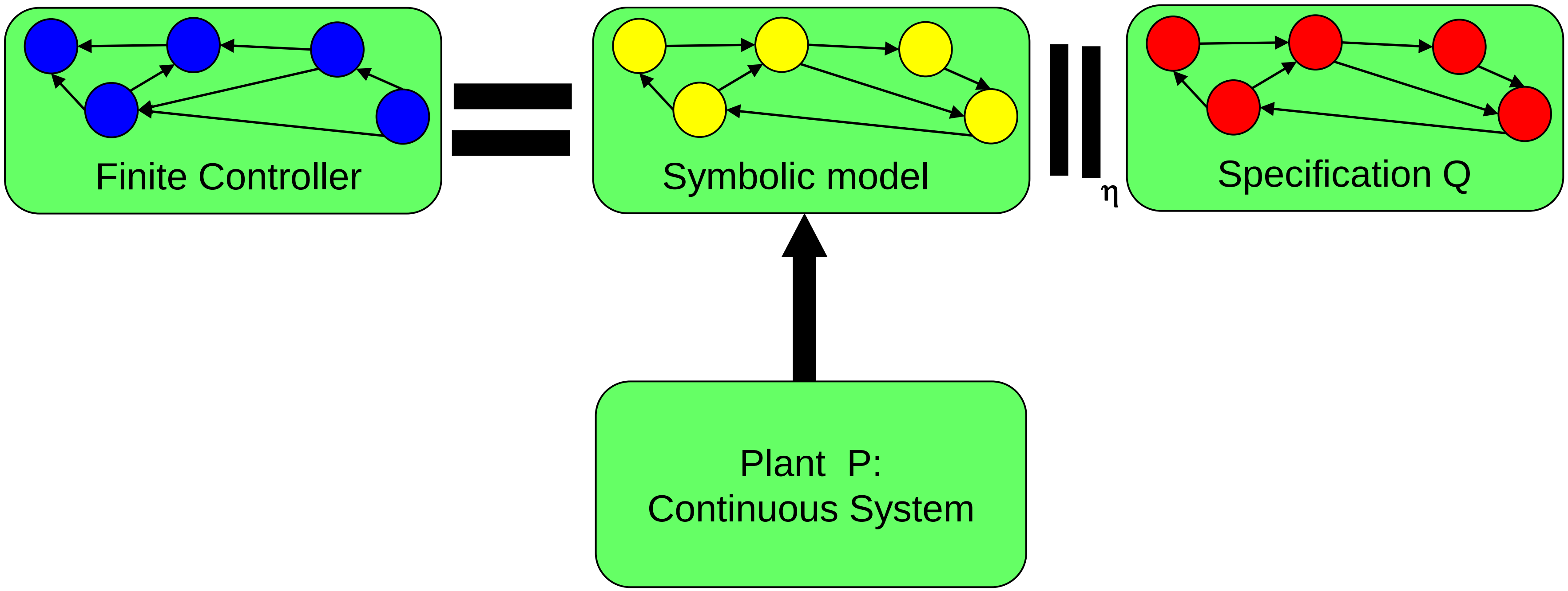
1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \|_{\eta} Q$
3. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 2

Synthesis through a three-step process:

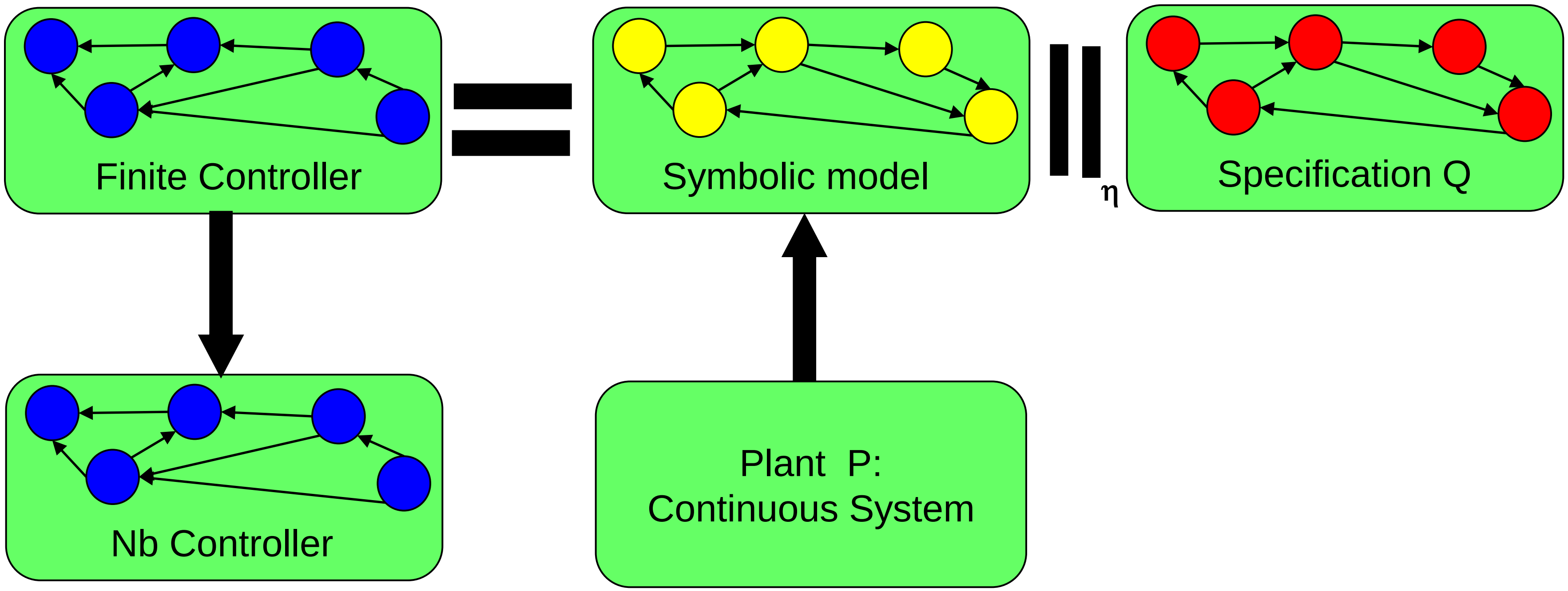
1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of P
2. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \|_{\eta} Q$
3. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 2

Synthesis through a three-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of $P$
2. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \|_{\eta} Q$
3. Compute the non-blocking part Nb(C*) of C*



# Solution of Problem 2

Synthesis through a three-step process:

1. Compute the symbolic model $T_{\tau,\eta,\mu}(P)$ of $P$
2. Compute the symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \|_{\eta} Q$
3. Compute the non-blocking part $Nb(C^*)$ of $C^*$

**Theorem** Suppose that $P$ is $\delta$–ISS and choose parameters $\tau, \eta, \mu, \theta > 0$ satisfying:

$$\beta(\theta,\tau) + \gamma(\mu) + 2\eta \leq \theta + \eta \leq \varepsilon$$

The symbolic controller $Nb(C^*)$ solves Problem 2.

# Design of symbolic controllers

**Drawbacks**

- It considers the whole sets of states of $T_{\tau,\eta,\mu}(P)$ and $Q$
- For any source state $x$ and target state $y$, it includes all transitions $x \xrightarrow{u} y$ with any control input $u$ by which state $x$ reaches state $y$
- It first constructs $T_{\tau,\eta,\mu}(P)$ and $Q$, then $C^*$, to finally eliminate blocking states from $C^*$

To cope with space and time complexity, instead of computing separately

(1) Discrete abstraction $T_{\tau,\eta,\mu}(P)$ of $P$
(2) Symbolic controller $C^* = T_{\tau,\eta,\mu}(P) \|_{\eta} Q$
(3) Non-blocking part $Nb(C)$ of $C^*$

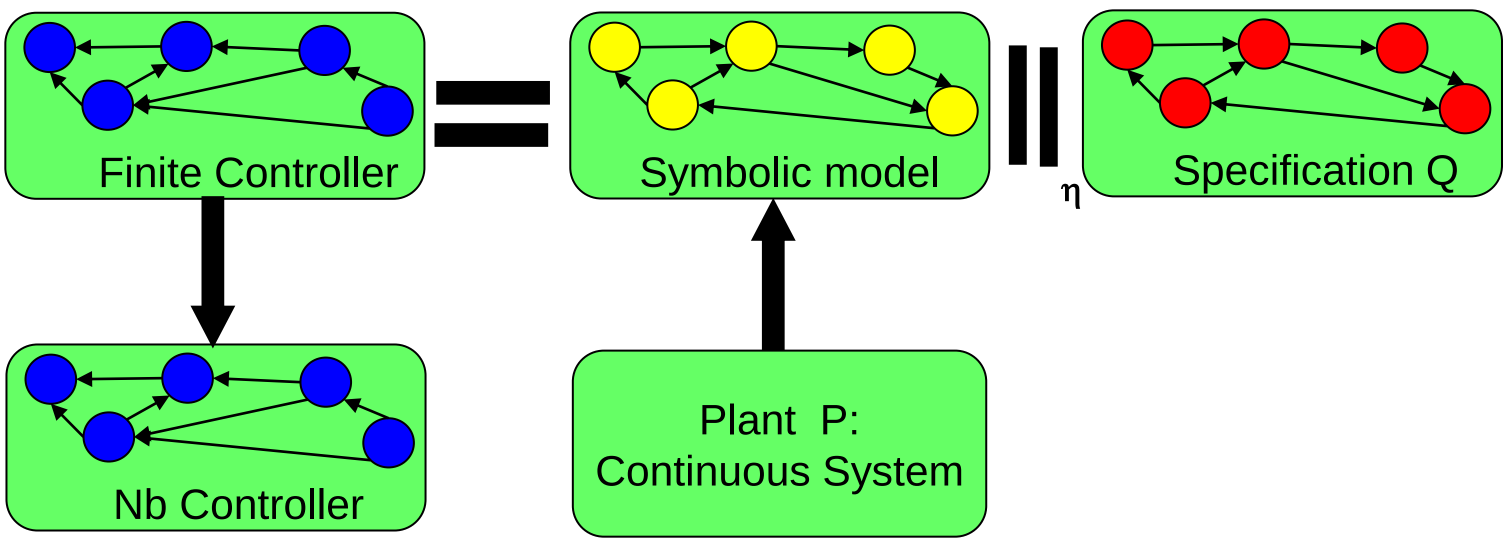
Integrated Approach: Compute (1) + (2) + (3) at once!

Space/time complexity analysis of the proposed algorithm formally quantifies the gain of the integrated approach

# Integrated Algorithm for Problem 2

**Basic ideas**

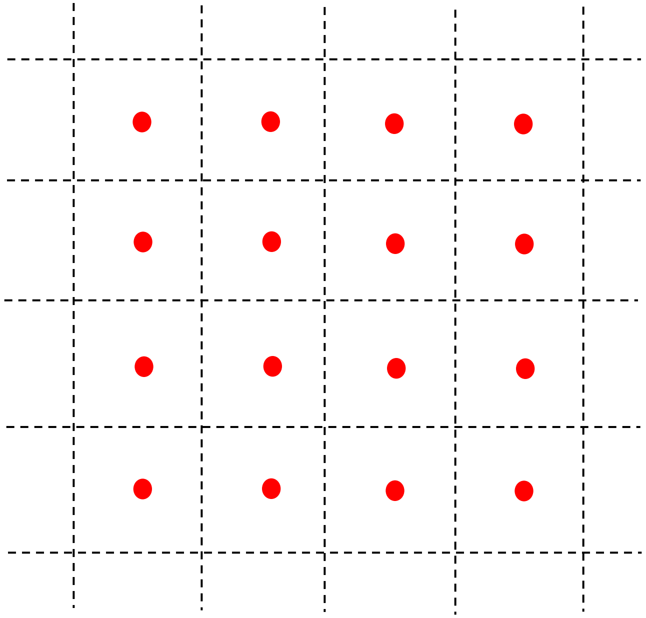
1. It only considers the intersection of the accessible parts of P and Q
2. For any given source state x and target state y, it considers only one transition (x,u,y)
3. It eliminates blocking states as soon as show up



# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and Q.

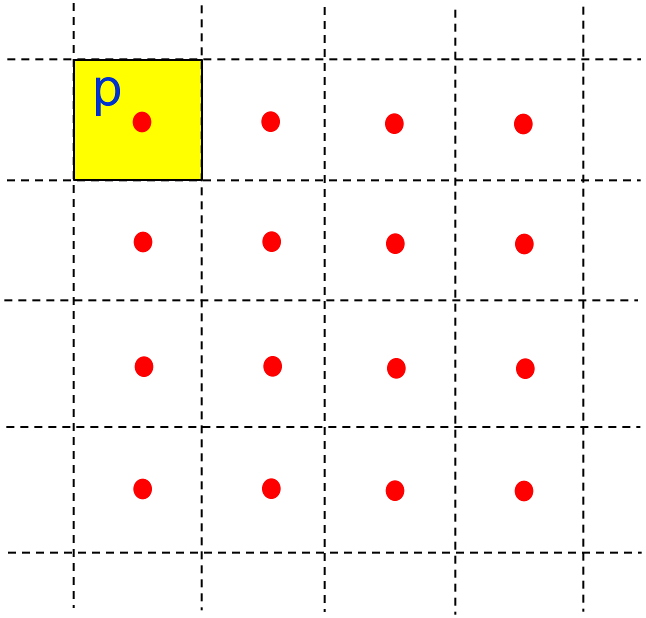


# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

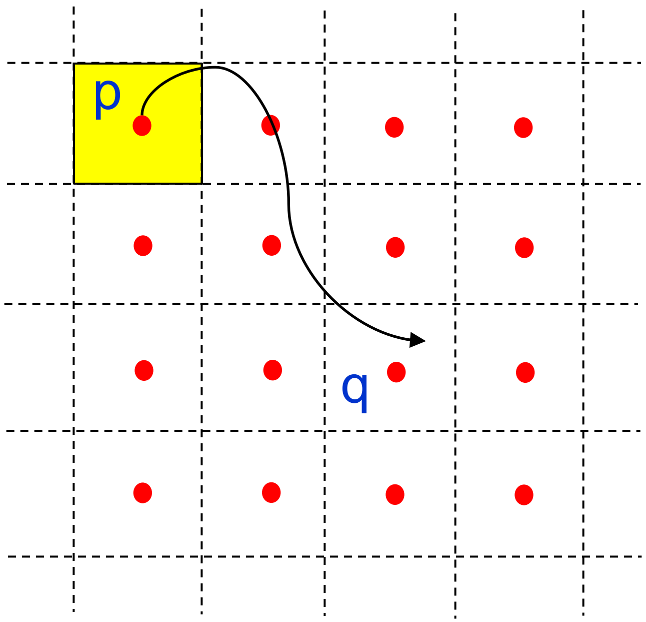


# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

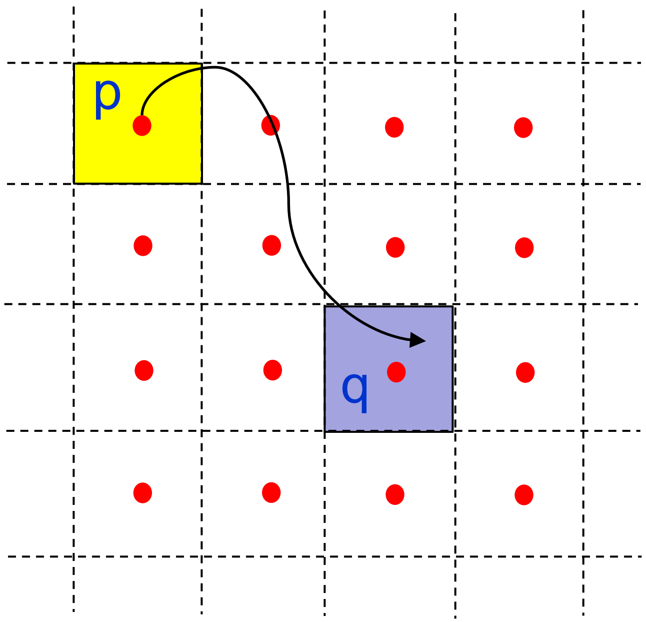


# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.



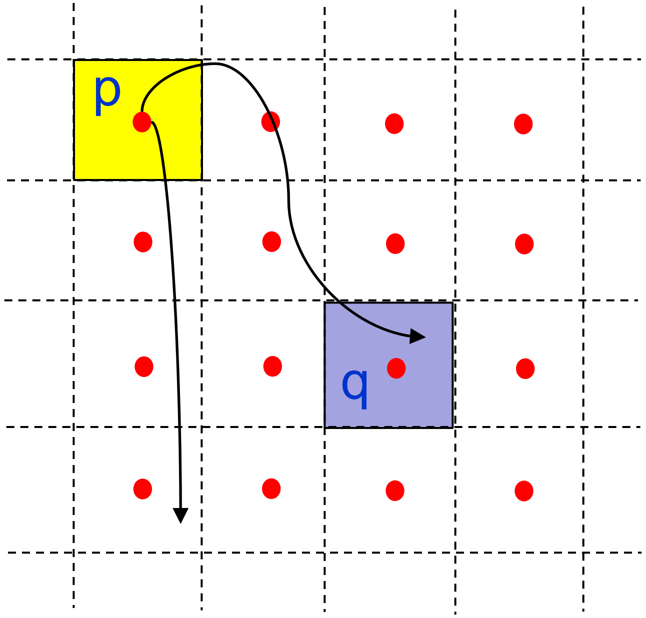
# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.



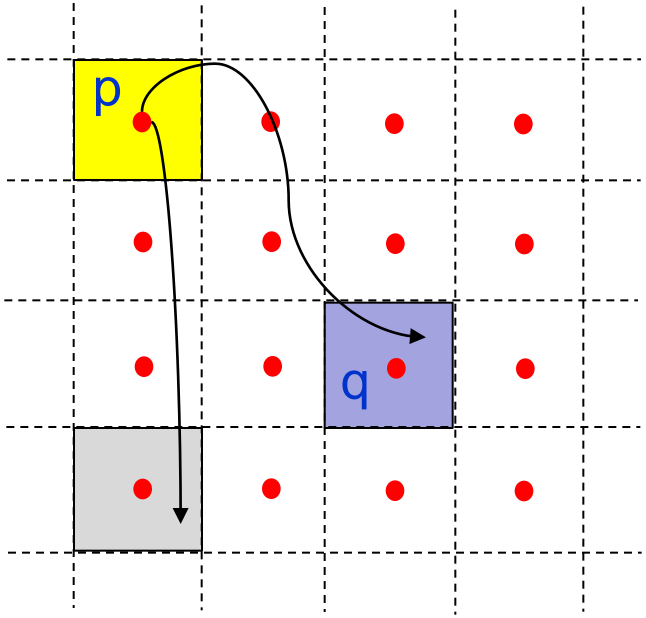
# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.




No matching! Try another input!

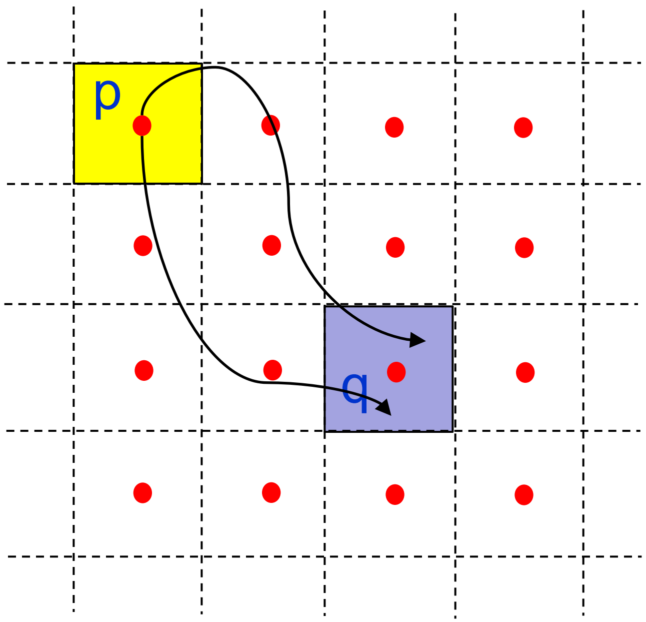
# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a “symbolic” state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.



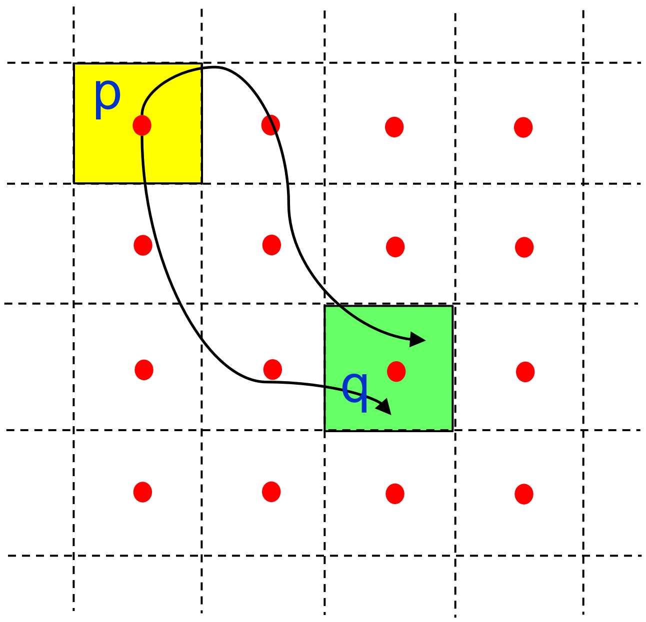
# Integrated Algorithm for Problem 2

**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.




Matching found!!

# Integrated Algorithm for Problem 2

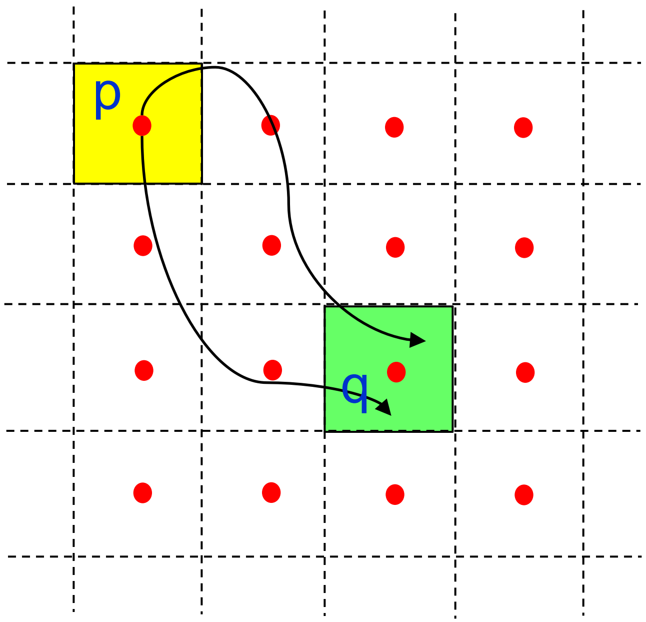
**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.

Add the transition $(p,u,q)$ to the controller




Matching found!!

# Integrated Algorithm for Problem 2

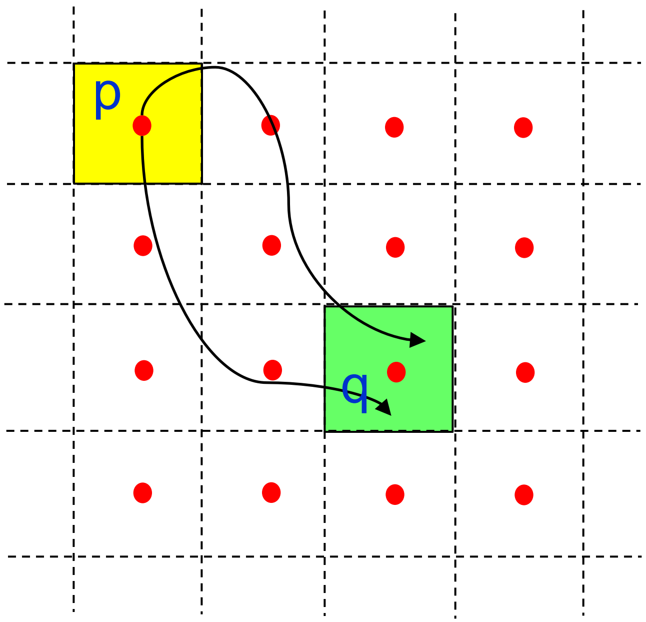
**How does it work? It is similar to the one for continuous specifications**

First, we consider the target space as the intersection of the sets of initial states of $T_{\tau,\eta,\mu}(P)$ and $Q$.

Pick a "symbolic" state $p$ from the target space and compute the unique state $q$ such that the transition $p \longrightarrow q$ is in $Q$.

Pick control inputs in $[U]_{2\mu}$ and integrate the plant differential equation until $q=[x(\tau,p,u)]_{2\eta}$ for some $u$.

Add the transition $(p,u,q)$ to the controller




If any "good" input does not exist, then $p$ is blocking!
A backwards procedure is executed to eliminate $p$ and all its ingoing transitions from the controller, until a controller is found which is non-blocking.

# Integrated Algorithm for Problem 2

## Properties

Let C** be the outcome of the integrated procedure:

1. The integrated algorithm terminates in a finite number of steps
2. C** and Nb(C*) are exactly bisimilar ⇒ C** solves Problem 2
3. C** is the minimal 0-bisimilar system of Nb(C*)
4. C** is accessible
5. space/time complexity of the integrated procedure is not larger than the one of the classical procedure

## Example 2

**Plant**

$$P : \begin{cases} \dot{x}_1 = -4x_1 + x_2^2 - u \\ \dot{x}_2 = 2x_1 - 7\sin x_2 \end{cases}$$

**Specification**

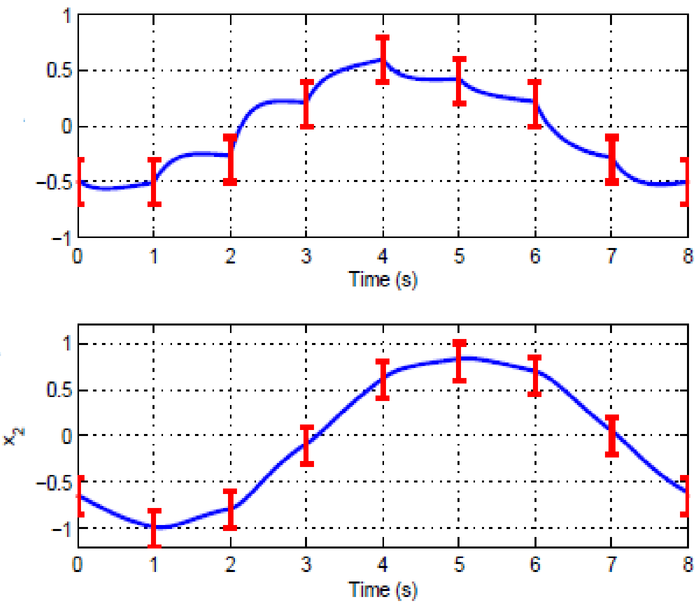
$(-0.5, -0.65) \longrightarrow (-0.5, -1) \longrightarrow$

$(-0.3, -0.8) \longrightarrow (0.2, -0.1) \longrightarrow$

$(0.6, 0.6) \longrightarrow (0.4, 0.8) \longrightarrow$

$(0.2, 0.65) \longrightarrow (-0.3, 0) \longrightarrow$

$(-0.5, -0.65)$




**Precision ε = 0.2**

| Comparison between Nb(C*) and C** | Nb(C*) | C** | Ratio |
|---|---|---|---|
| Max memory occupation | 2,759,580 | 48 | $1.74 \cdot 10^{-5}$ |
| Time | 5,442 | 13 | 0.002 |