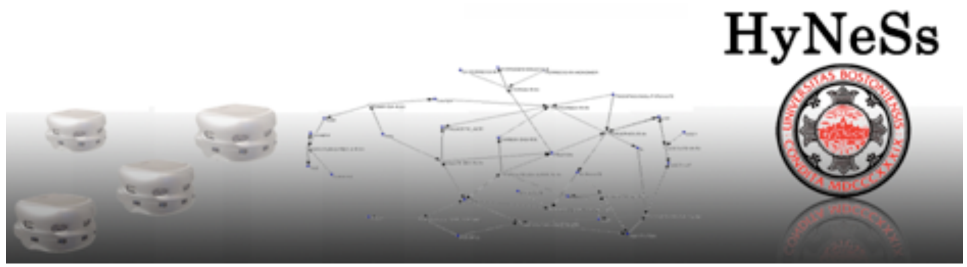
Hybrid and Networked Systems Lab




# Formal Verification and Synthesis of Piecewise Affine Systems with Applications to Gene Networks

**Calin Belta**
Mechanical Engineering, Systems Engineering, and Bioinformatics
Boston University

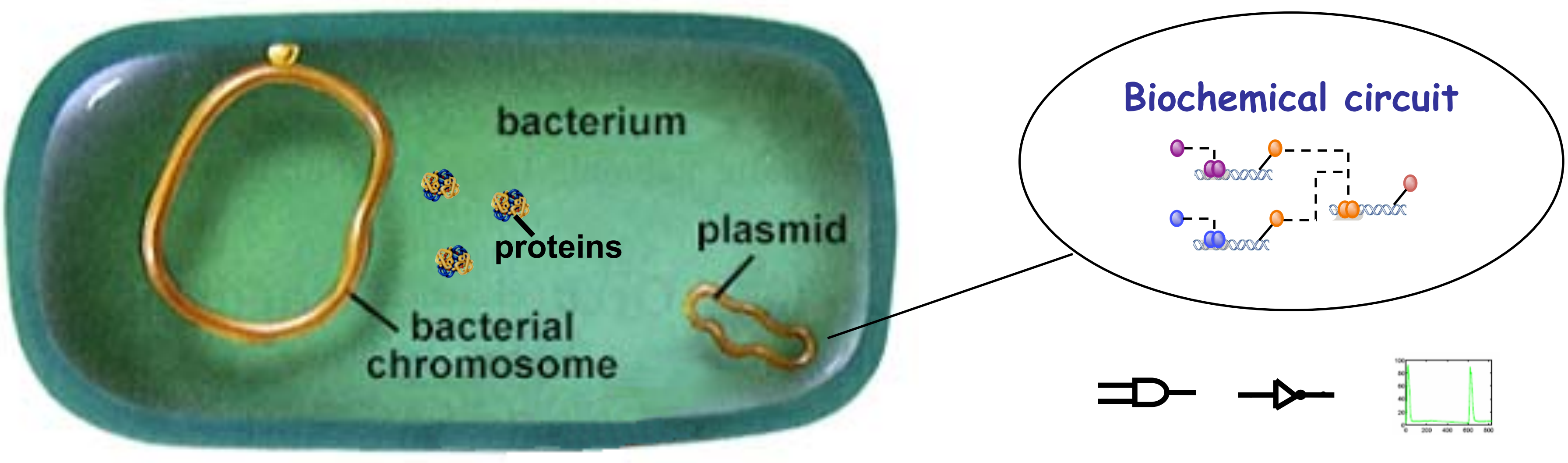
Joint work with **Boyan Yordanov**
Biomedical Engineering, Boston University
(→ Microsoft Research, Cambridge, UK)

# Motivation

**Synthetic Biology** is
A) the design and construction of new biological parts, devices, and systems, and
B) the re-design of existing, natural biological systems for useful purposes.

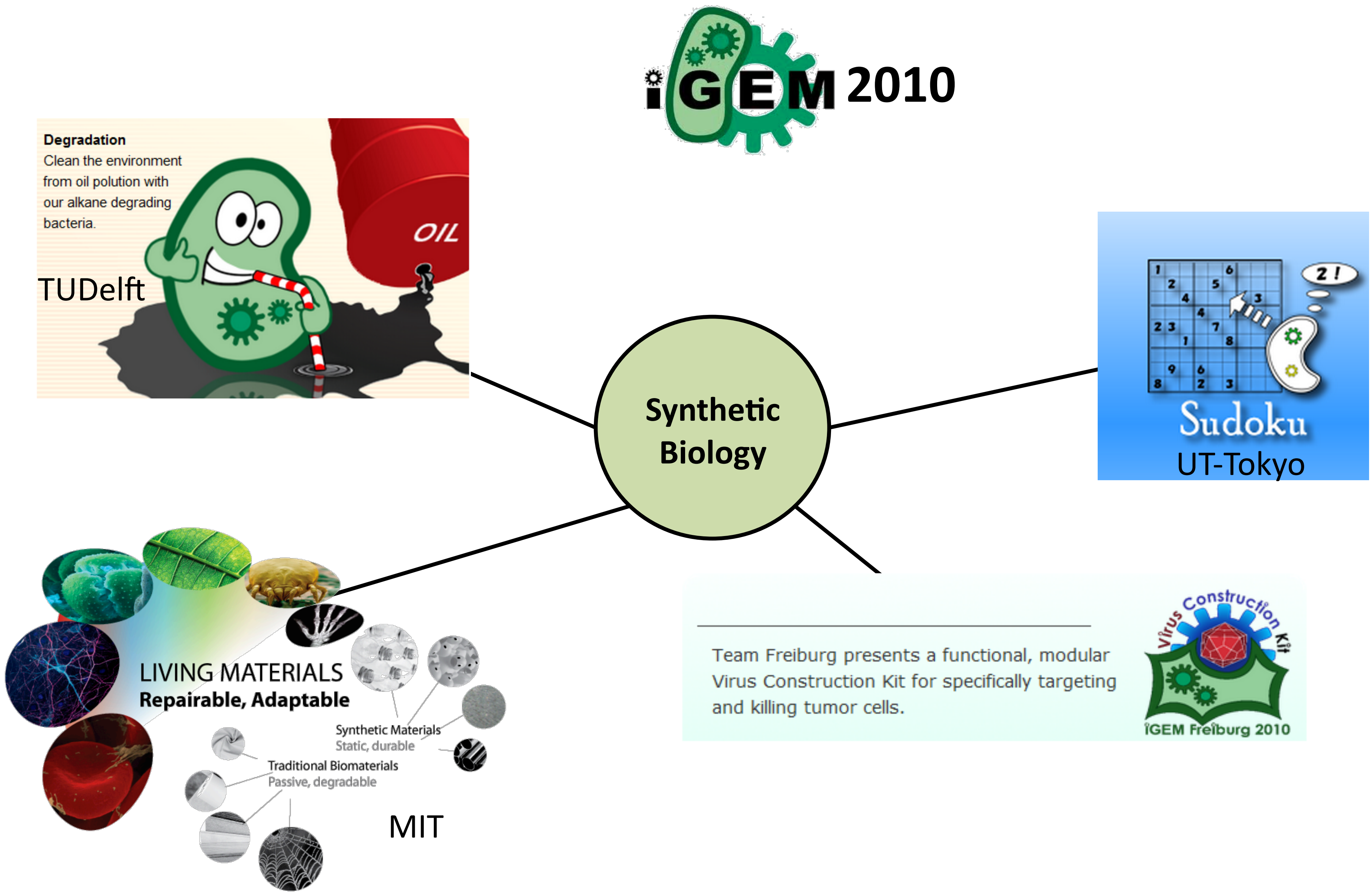
http://syntheticbiology.org/

- Bioremediation
- Biosensing
- Nanofabrication
- Therapeutics
- Biofabrication
- Biocomputing




Examples: toggle switch (Gardner 2000), oscillator (Elowitz 2000), logical gates (Weiss 2002), sensing and communication mechanisms (Weiss 2000), pulse generator (Basu 2004).

# Motivation

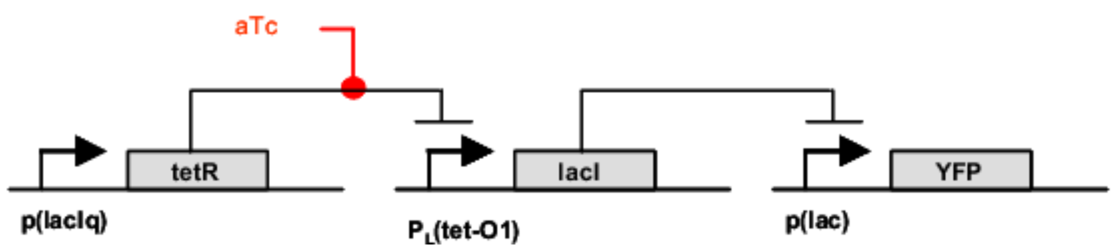


iGEM 2010
Degradation
Clean the environment from oil polution with our alkane degrading bacteria.
OIL
TUDelft
Synthetic Biology
Sudoku
UT-Tokyo
LIVING MATERIALS
Repairable, Adaptable
Synthetic Materials
Static, durable
Traditional Biomaterials
Passive, degradable
MIT
Team Freiburg presents a functional, modular Virus Construction Kit for specifically targeting and killing tumor cells.
Virus Construction Kit
iGEM Freiburg 2010

# Motivation

NSF CCF-0432070: "Collaborative Research: Rational Design of Synthetic Gene Networks using Formal Analysis of Hybrid Systems"

**Aim:** tune the parameters of a set of existing synthetic circuits such that **all possible behaviors** of the circuits satisfy a given specification



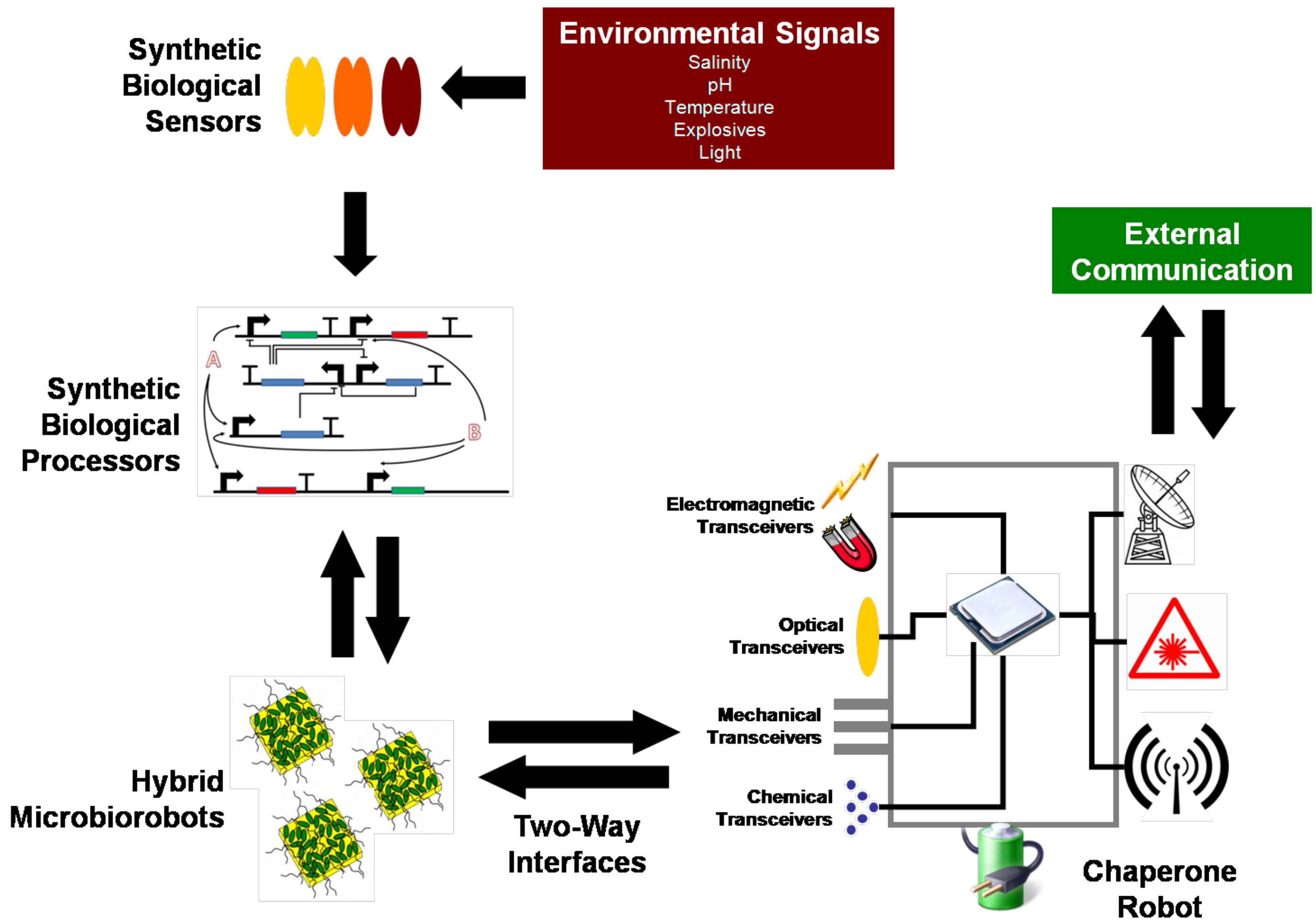

| aTc | YFP |
|---|---|
| < low | > high |
| > high | < low |

**Specification:**

**if aTc < low, then eventually always YFP > high, and if aTc > high, then eventually always YFP < low**

# Motivation

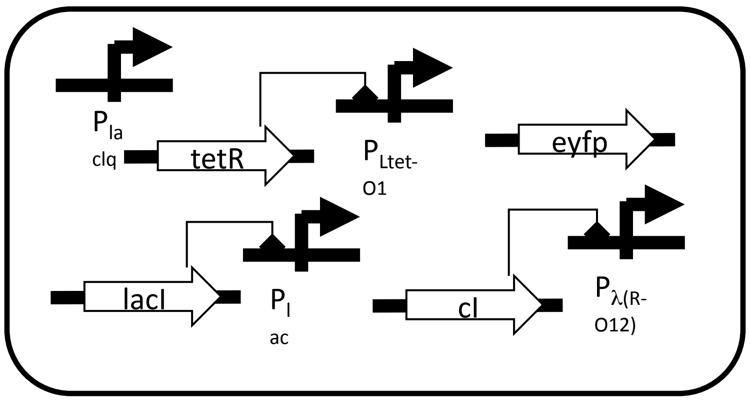
ONR MURI: Utilizing Synthetic Biology to Create Programmable Micro-Bio-Robots



# Motivation

ONR MURI: Utilizing Synthetic Biology to Create Programmable Micro-Bio-Robots

**One specific aim:** from a set of available parts, construct a circuit satisfying a given specification



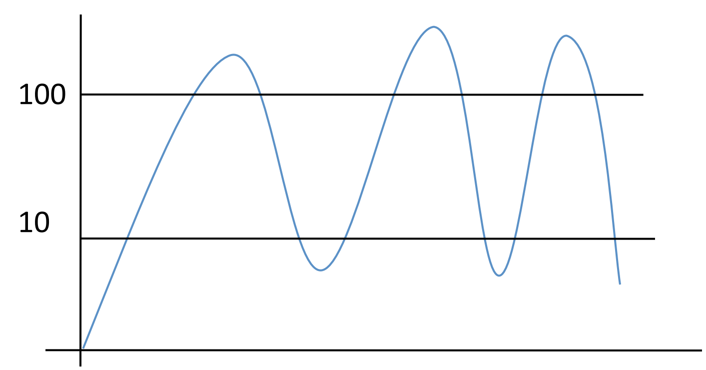

Registry of Standard Biological Parts

http://partsregistry.org/

**Specification:**

Eventually, the concentration of *eyfp* starts oscillating between values above 100 and below 10, i.e.,

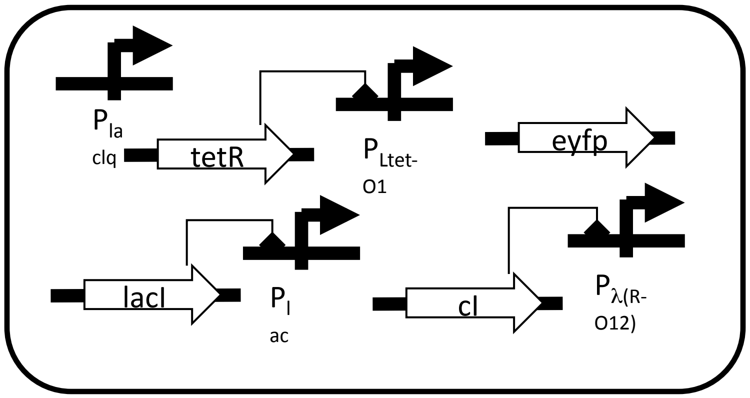
"Always eventually eyfp > 100 and always eventually eyfp < 10"



# Motivation

ONR MURI: Utilizing Synthetic Biology to Create Programmable Micro-Bio-Robots

## 1. *In silico* construction of all biologically feasible circuits




**Registry of Standard Biological Parts**

http://partsregistry.org/

**BioBricks**

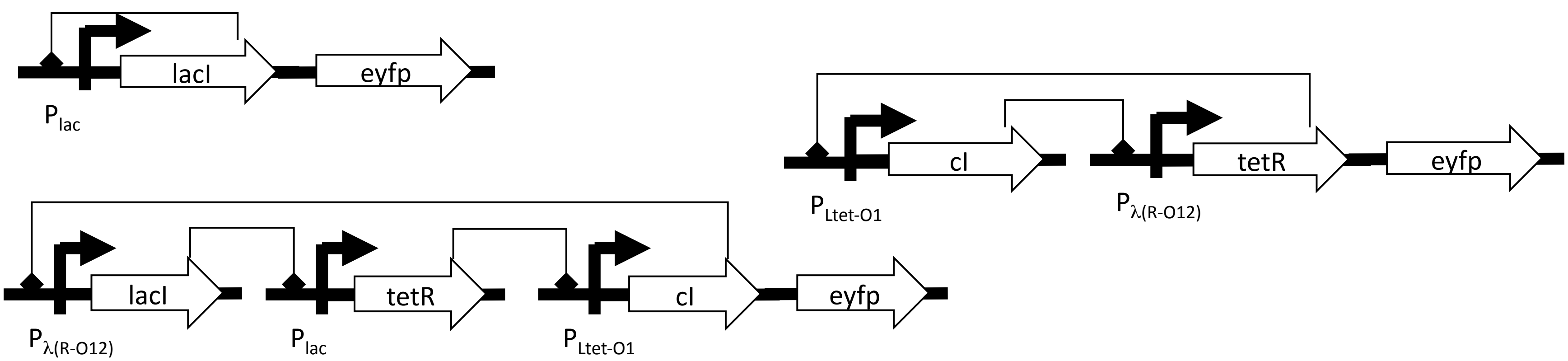
Knight, 2003

http://biobricks.org/

**Clotho**

Densmore et al., 2009

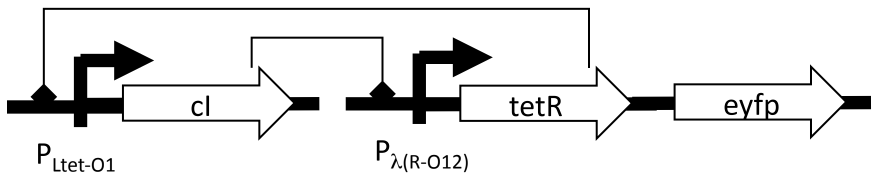
http://www.clothocad.org/

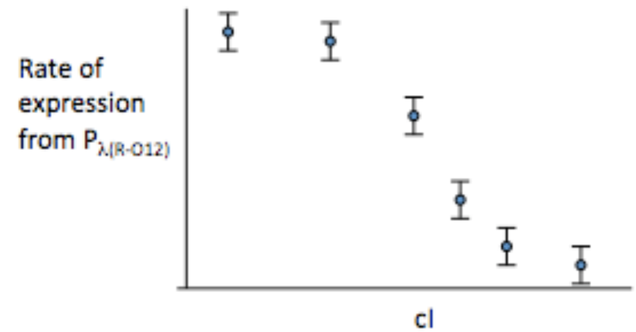


# Motivation

ONR MURI: Utilizing Synthetic Biology to Create Programmable Micro-Bio-Robots

**2. For each circuit, using the available information on the kinetic parameters and/or experimental data, check the satisfaction of the specification for a mathematical model of the circuit**




"Always eventually eyfp > 100 and always eventually eyfp < 10"

Rbs calculator
http://www.voigtlab.ucsf.edu/software/

Protein decay rates ExPASy
http://ca.expasy.org/

**Mathematical model**

**Verification**

# Approach

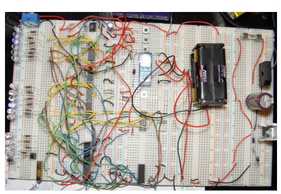
## Draw inspiration from formal analysis (verification)

**Specification**

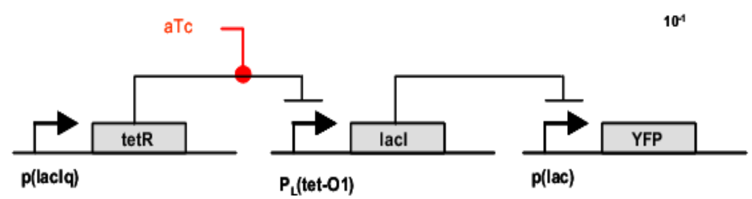
"Is deadlock ever possible?"
"If a request is received, make sure it is eventually granted."

if aTc < low, then eventually always YFP > high, and if aTc > high, then eventually always YFP < low

**Process**



```
#include<time.h>

main()
{
        clock_t time,deltime;
        long junk,i;
        float secs;
LOOP:
        printf("input loop count:  ");
        scanf("%ld",&junk);
        time = clock();
        for(i=0;i<junk;i++)
        deltime = clock() - time;
        secs = (float) deltime/CLOCKS_PER_SEC;
        printf("for %ld loops, #tics = %ld, time
        goto LOOP;
        return 0;
}
```



# Approach

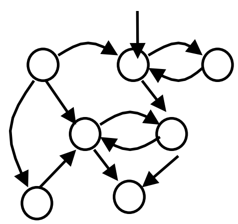
## Draw inspiration from formal analysis (verification)

**Specification**

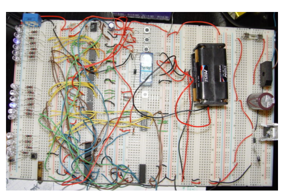
"Is deadlock ever possible?"
"If a request is received, make sure it is eventually granted."

**Model checking (SPIN, NuSMV)**

**Model**

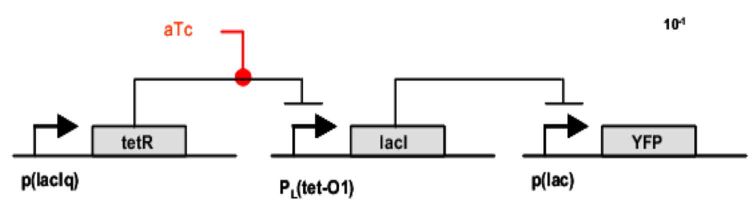


**Process**



```
#include<time.h>

main()
{
        clock_t time,deltime;
        long junk,i;
        float secs;
LOOP:
        printf("input loop count:  ");
        scanf("%ld",&junk);
        time = clock();
        for(i=0;i<junk;i++)
        deltime = clock() - time;
        secs = (float) deltime/CLOCKS_PER_SEC;
        printf("for %ld loops, #tics = %ld, time
        goto LOOP;
        return 0;
}
```

if aTc < low, then eventually always YFP > high, and if aTc > high, then eventually always YFP < low



# Approach

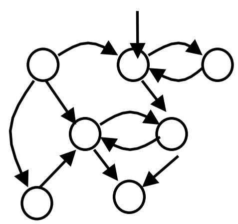
Draw inspiration from formal analysis (verification)

**Specification**

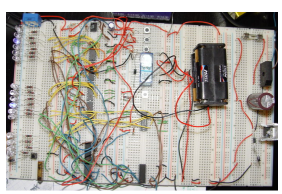
"Is deadlock ever possible?"
"If a request is received, make sure it is eventually granted."

**Model checking (SPIN, NuSMV)**

**Model**



**Process**



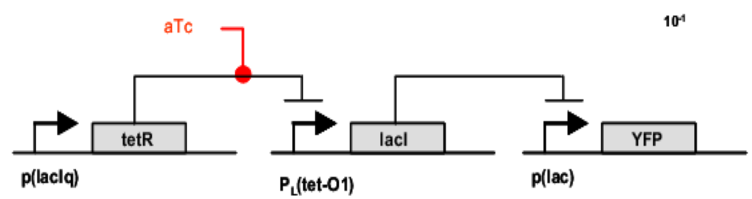
```
#include<time.h>

main()
{
        clock_t time,deltime;
        long junk,i;
        float secs;
LOOP:
        printf("input loop count:  ");
        scanf("%ld",&junk);
        time = clock();
        for(i=0;i<junk;i++)
        deltime = clock() - time;
        secs = (float) deltime/CLOCKS_PER_SEC;
        printf("for %ld loops, #tics = %ld, time
        goto LOOP;
        return 0;
}
```

if aTc < low, then eventually always YFP > high, and if aTc > high, then eventually always YFP < low

**?**

- Analysis / control

$$\dot{x} = f(x, u)$$



# Outline

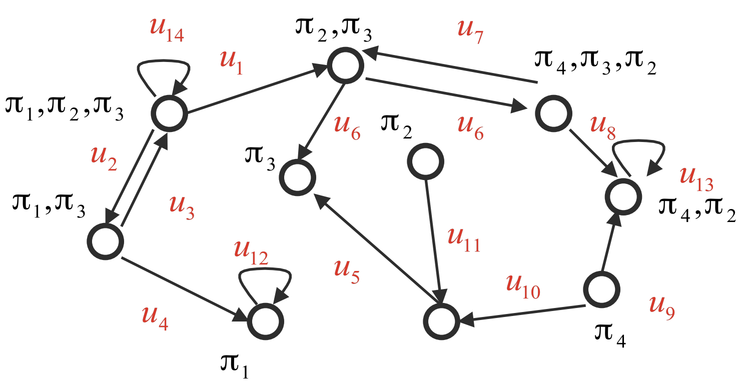
1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

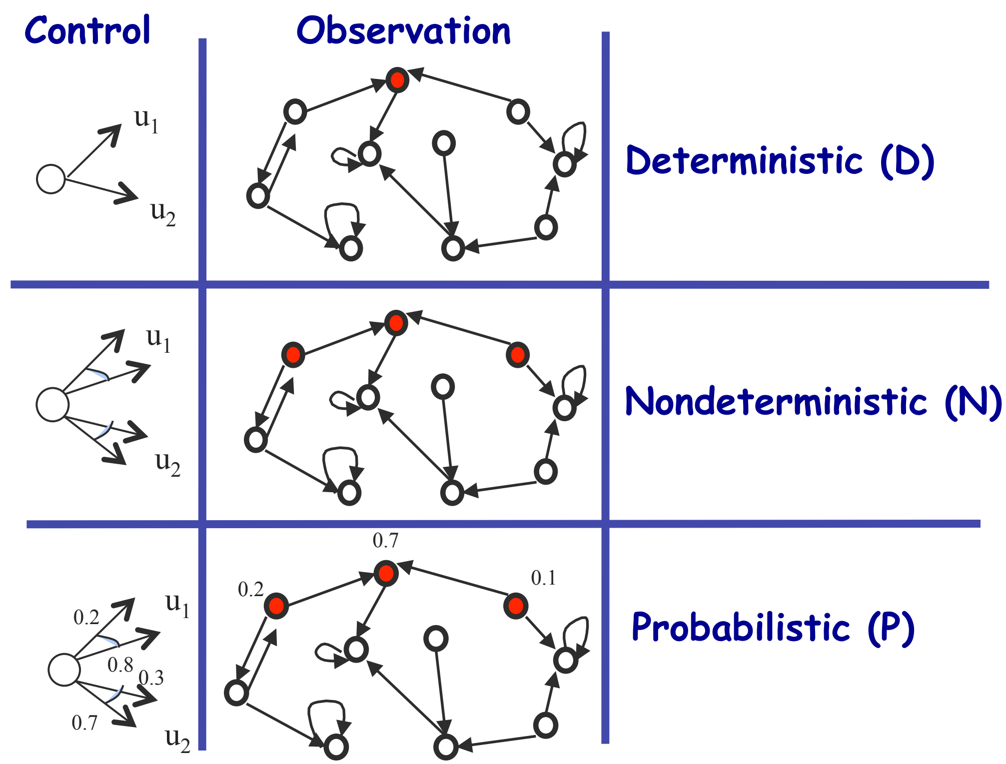
# Outline

1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

# LTL Verification and Control for Finite Systems

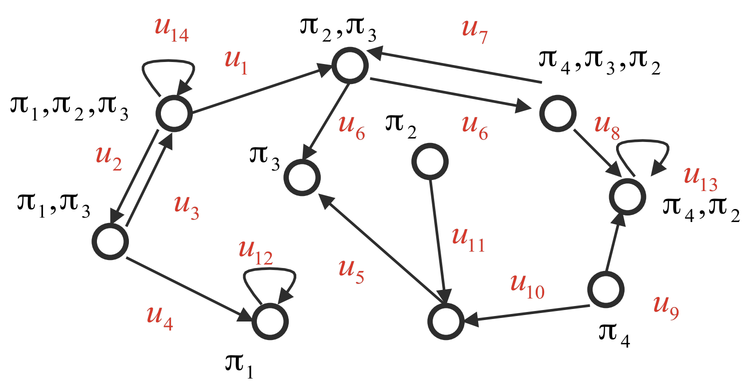
## Transition systems with finitely many states and actions

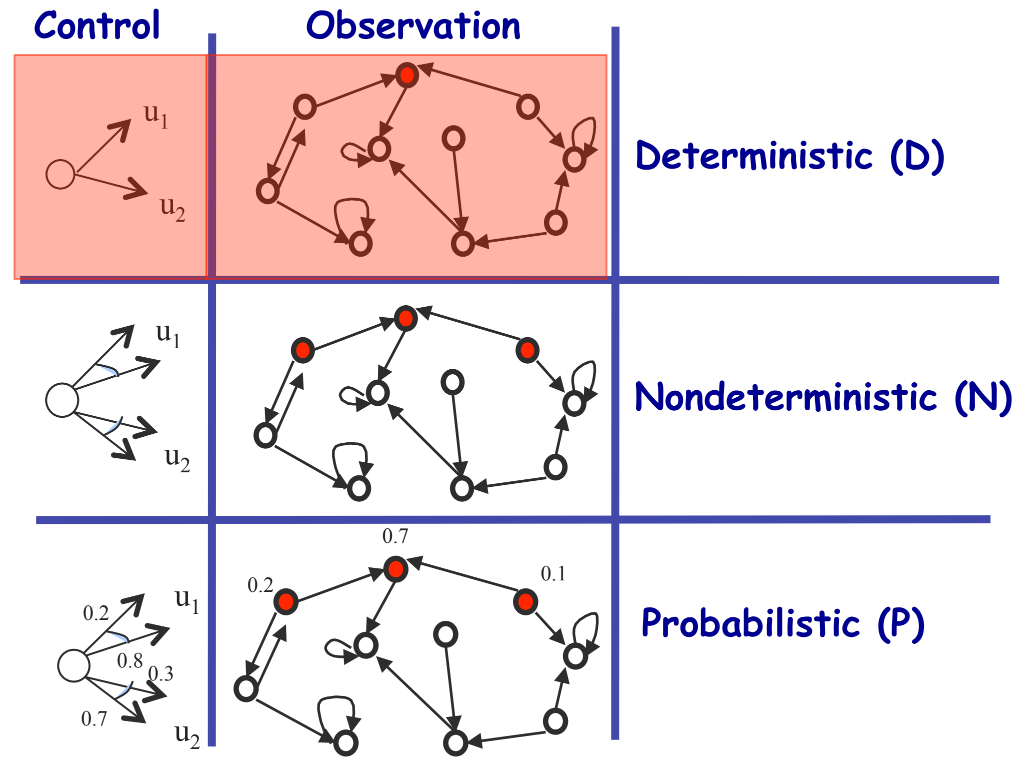


# LTL Verification and Control for Finite Systems

**Transition systems with finitely many states and actions**

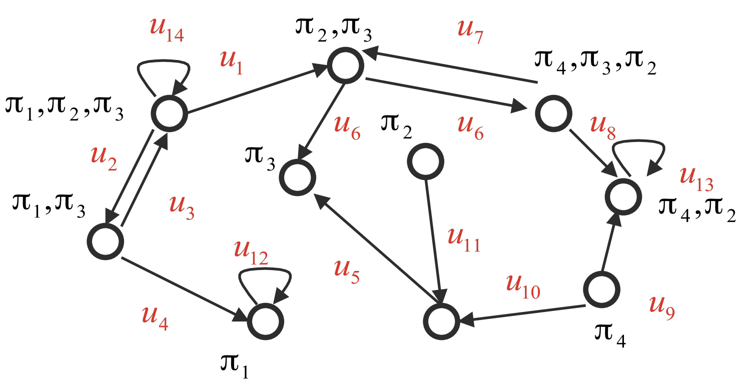


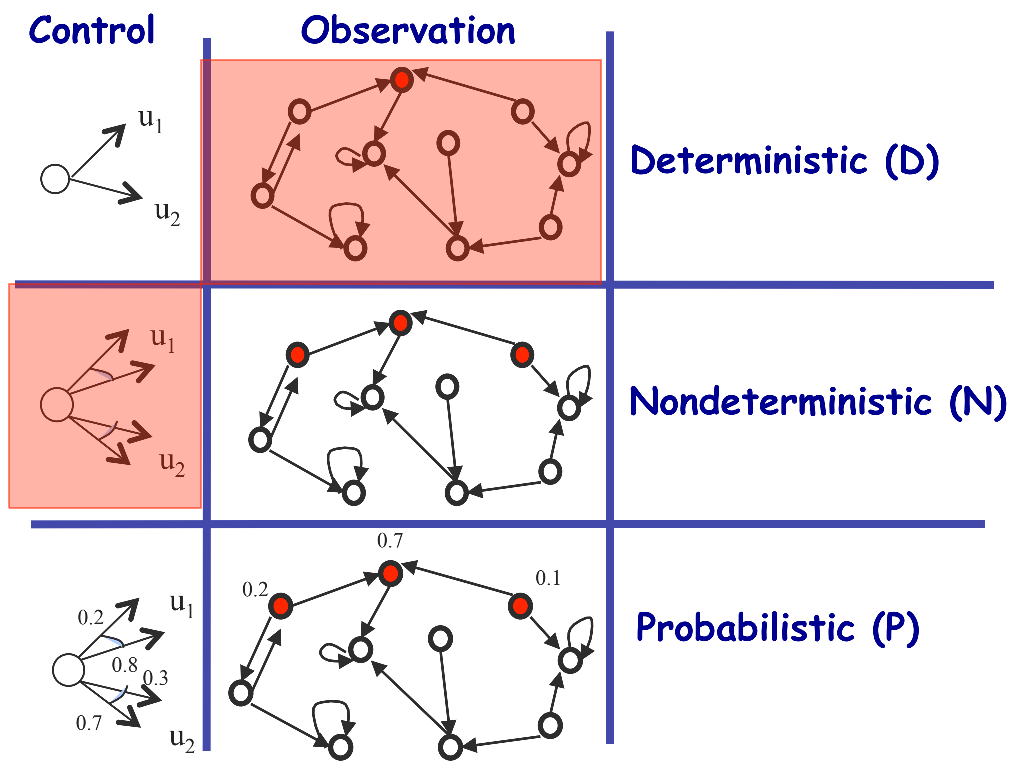




**D-D**: deterministic fully observable transition system

# LTL Verification and Control for Finite Systems

## Transition systems with finitely many states and actions

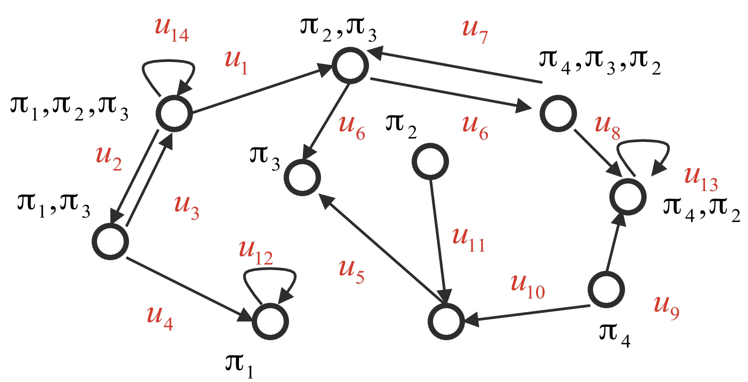


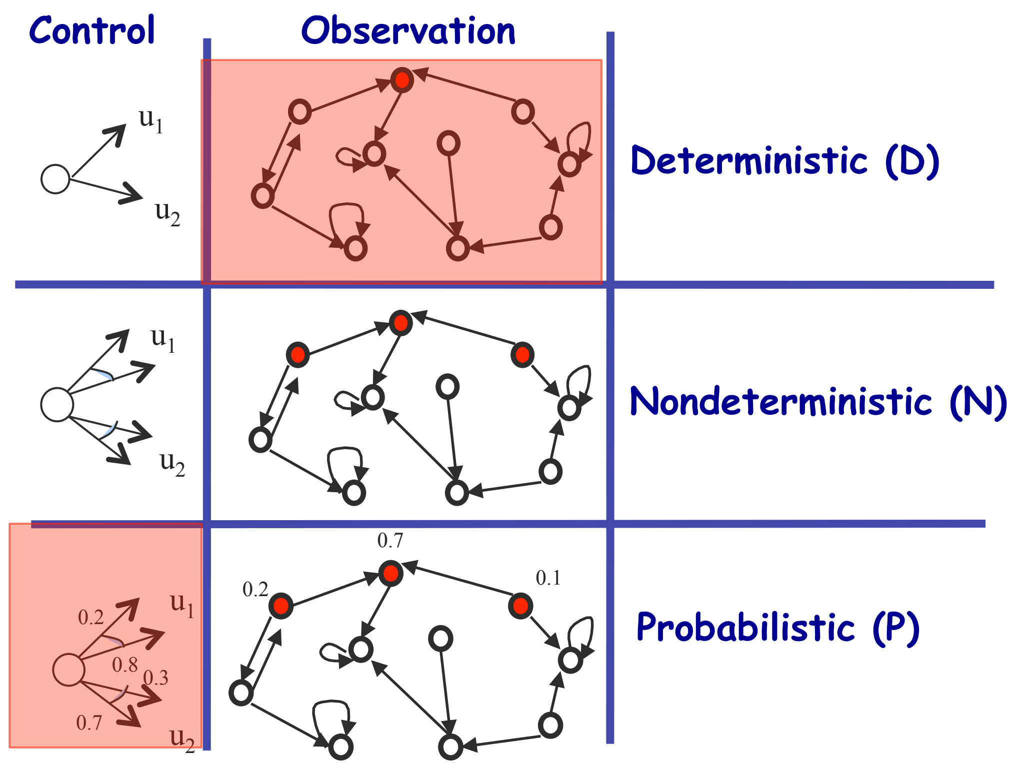




**N-D**: nondeterministic fully observable transition system

# LTL Verification and Control for Finite Systems

## Transition systems with finitely many states and actions

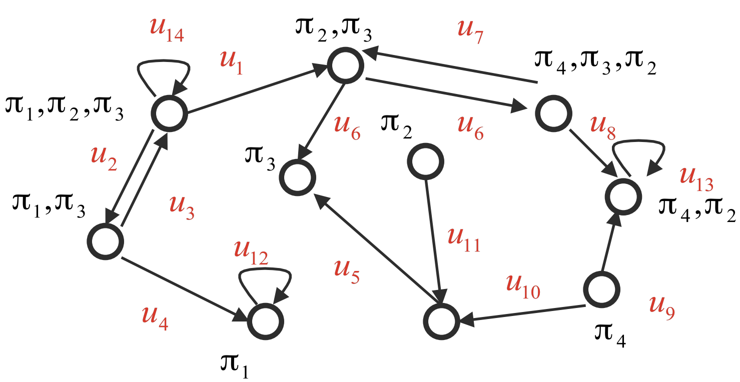


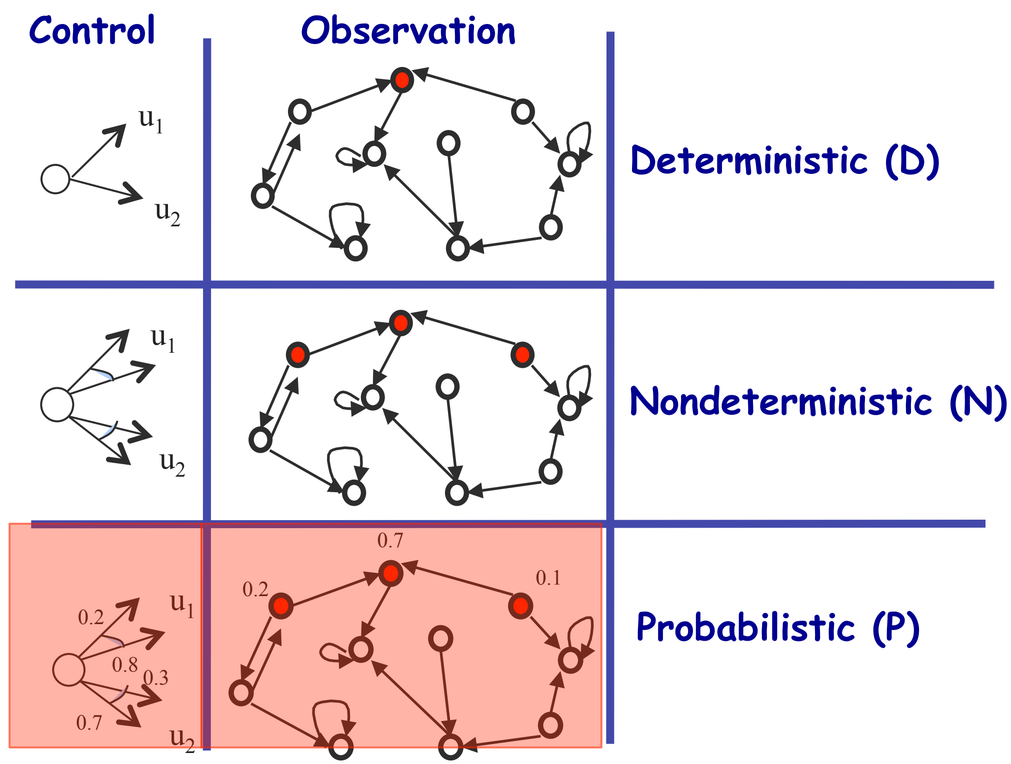




**P-D**: Markov Decision Process (MDP)

# LTL Verification and Control for Finite Systems

## Transition systems with finitely many states and actions

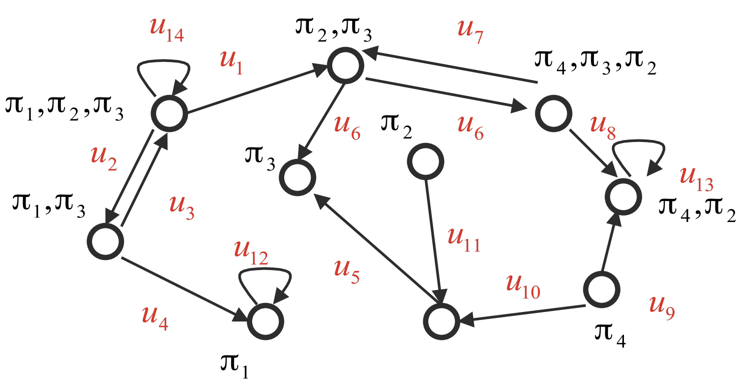


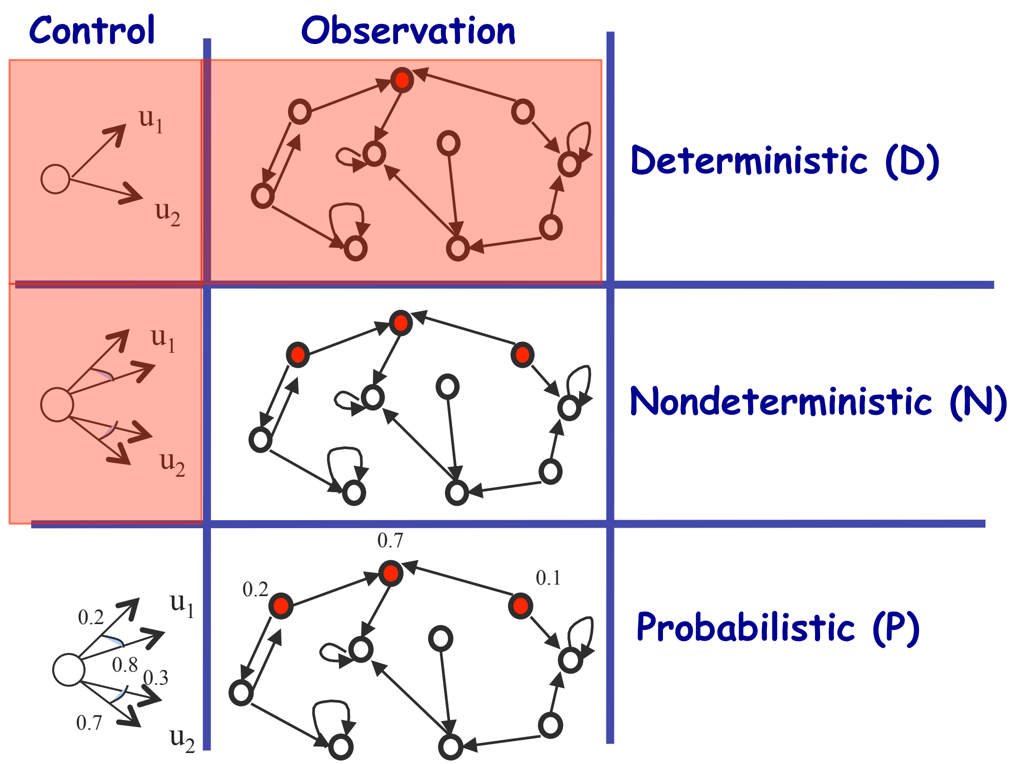




**P-P**: Partially Observable Markov Decision Process (POMDP)

# LTL Verification and Control for Finite Systems

**Transition systems with finitely many states and actions**




**In this talk:**

**D-D**: deterministic fully observable transition system

**N-D**: nondeterministic fully observable transition system

# LTL Verification and Control for Finite Systems

## Linear Temporal Logic (LTL)

### Syntax

$\Diamond \pi_2$ (eventually) $\quad \Diamond \Box (\pi_3 \wedge \pi_4)$ (always) $\quad (\pi_1 \vee \pi_2) U \pi_4$ (until)

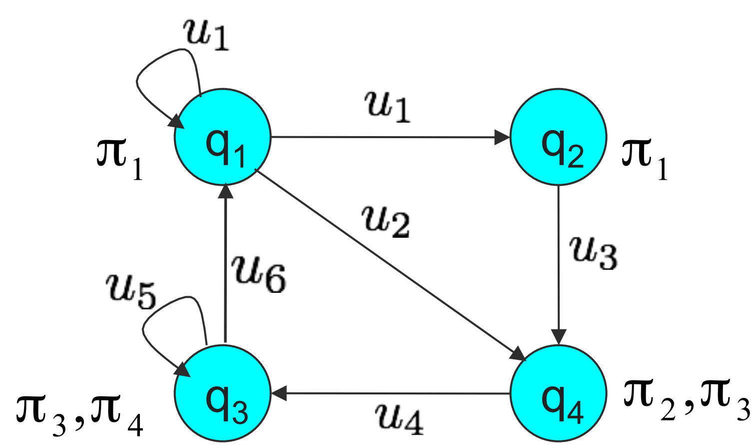
### Semantics

Run (trajectory): $q_1, q_4, q_3, q_3, \ldots$

Word: $\pi_1 \quad \pi_2 \; \pi_3 \quad \pi_3 \quad \ldots$

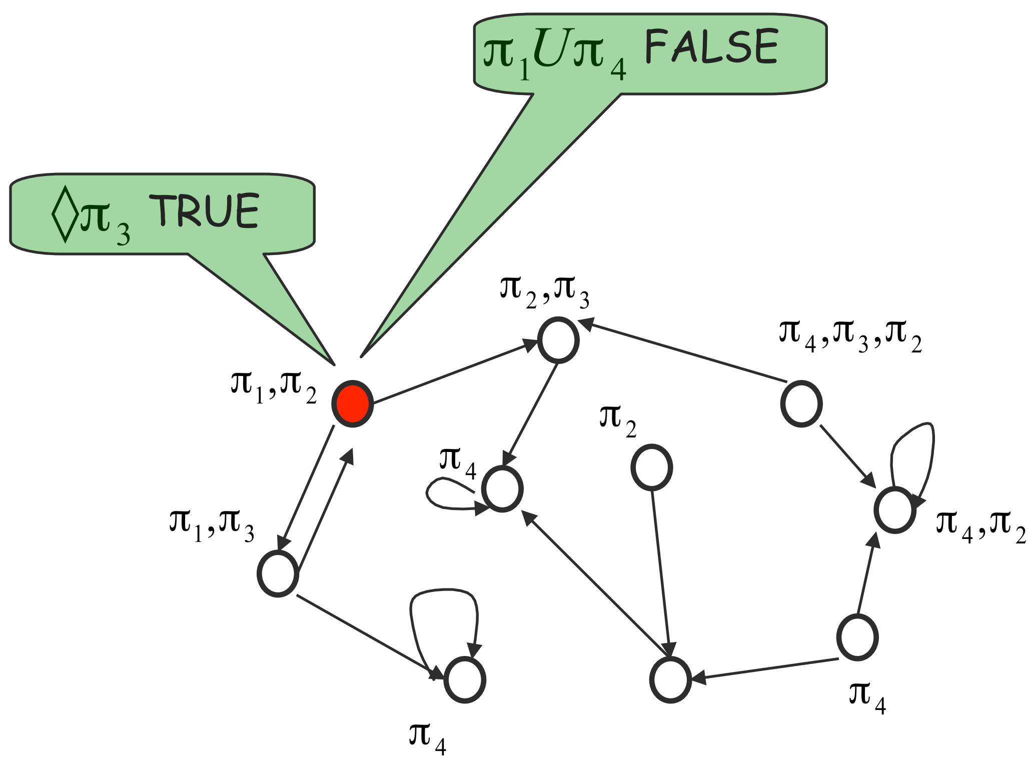
$\quad\quad\quad\; \pi_3 \; \pi_4 \; \pi_4$

Language: the set of all words



# LTL Verification and Control for Finite Systems

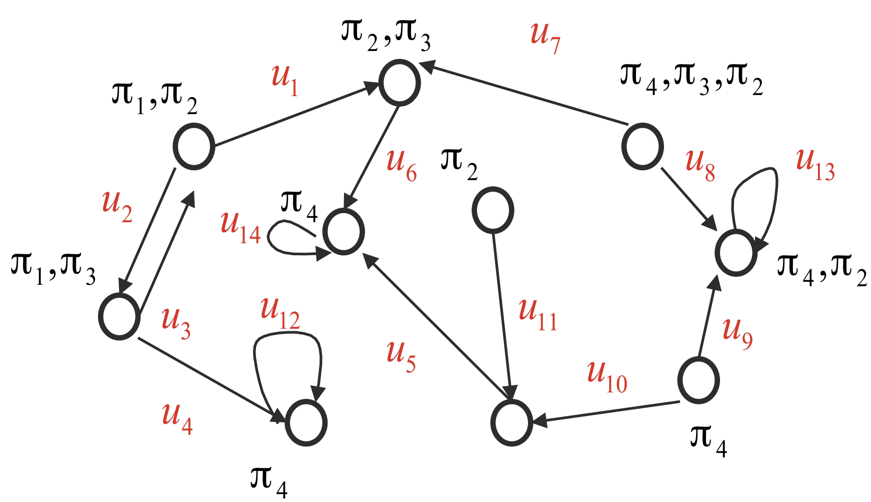
Given a transition system and an LTL formula over its set of propositions, check if the language of the transition system starting from all initial states satisfies the formula.




SPIN, NuSMV, …

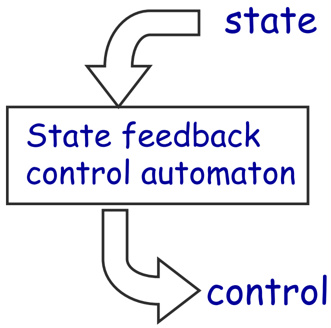
# LTL Verification and Control for Finite Systems

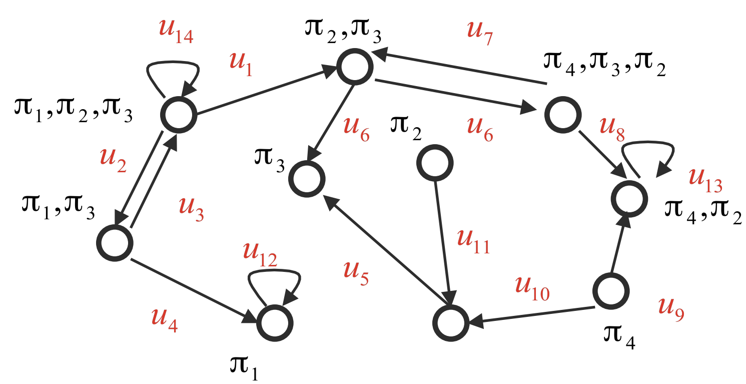
Given a transition system and an LTL formula over its set of propositions, find a set of initial states and a control strategy for all initial states such that the produced language of the transition system satisfies the formula.



# LTL Verification and Control for Finite Systems

Given a transition system and an LTL formula over its set of propositions, find a set of initial states and a control strategy for all initial states such that the produced language of the transition system satisfies the formula.




• **for deterministic systems** the solution is a simple adaptation of LTL model checking algorithms

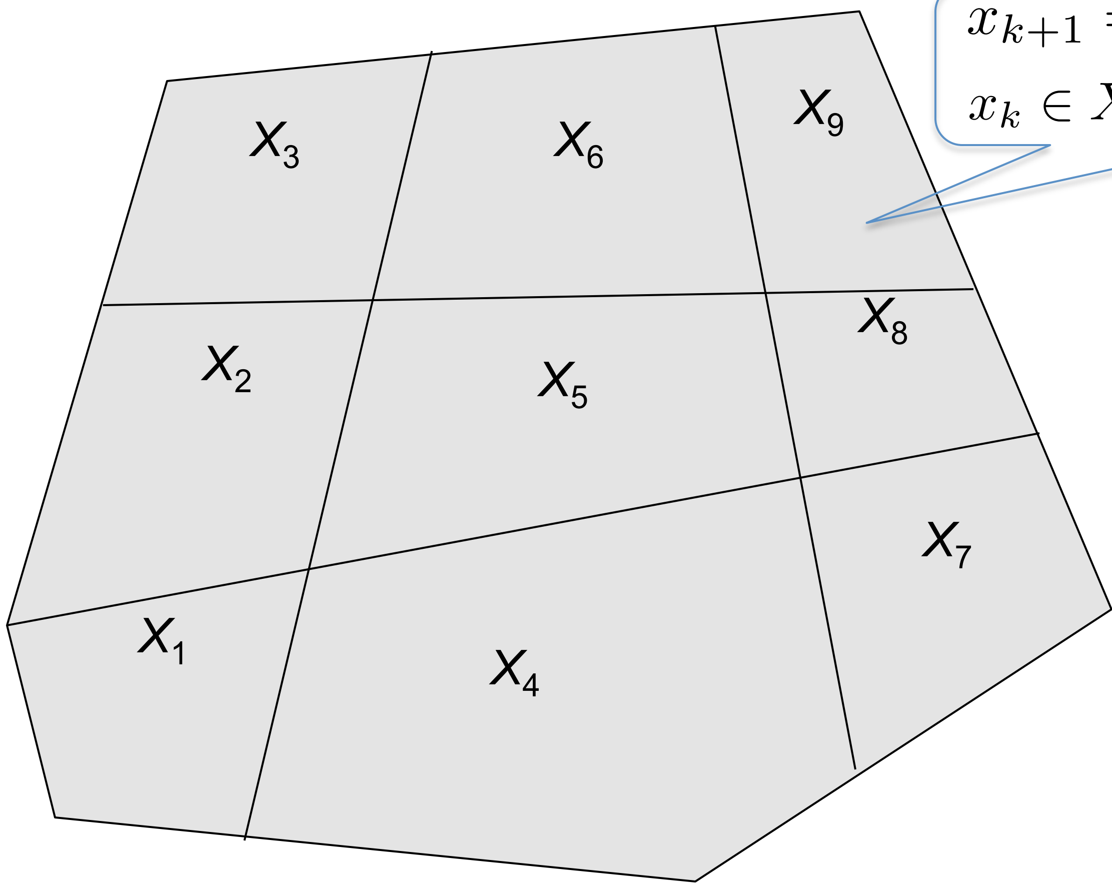
• **for nondeterministic systems** the solution is based on Buchi and Rabin games

# Outline

1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

# Piecewise Affine (PWA) Systems

## Syntax




$$x_{k+1} = A_l x_k + B_l u_k + b_l$$

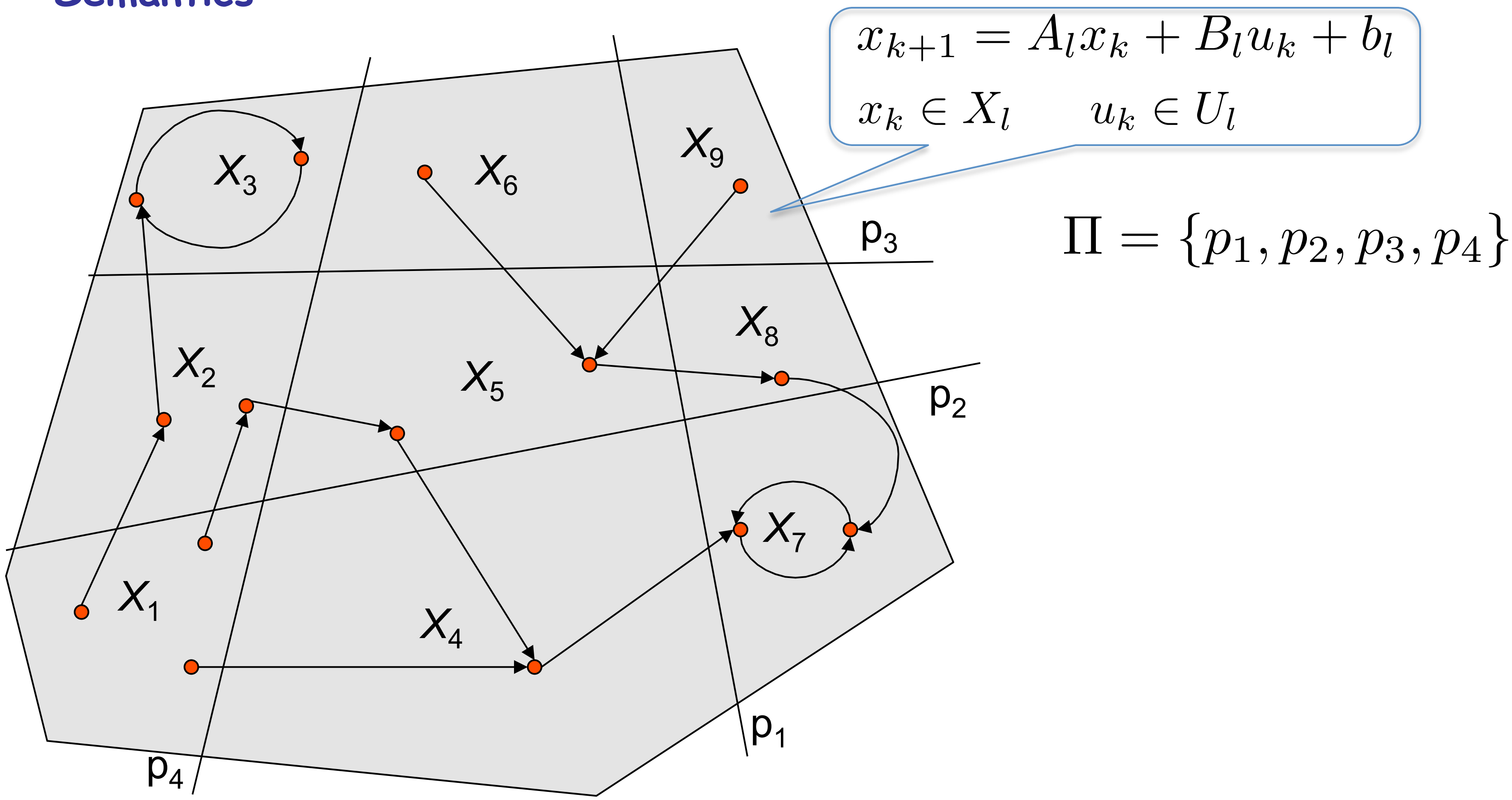
$$x_k \in X_l \qquad u_k \in U_l$$

$$A_l \in P_l^A \qquad B_l \in P_l^B \qquad b_l \in P_l^b \qquad l \in L$$

All the sets are polyhedral subsets of Euclidean spaces of appropriate dimensions.

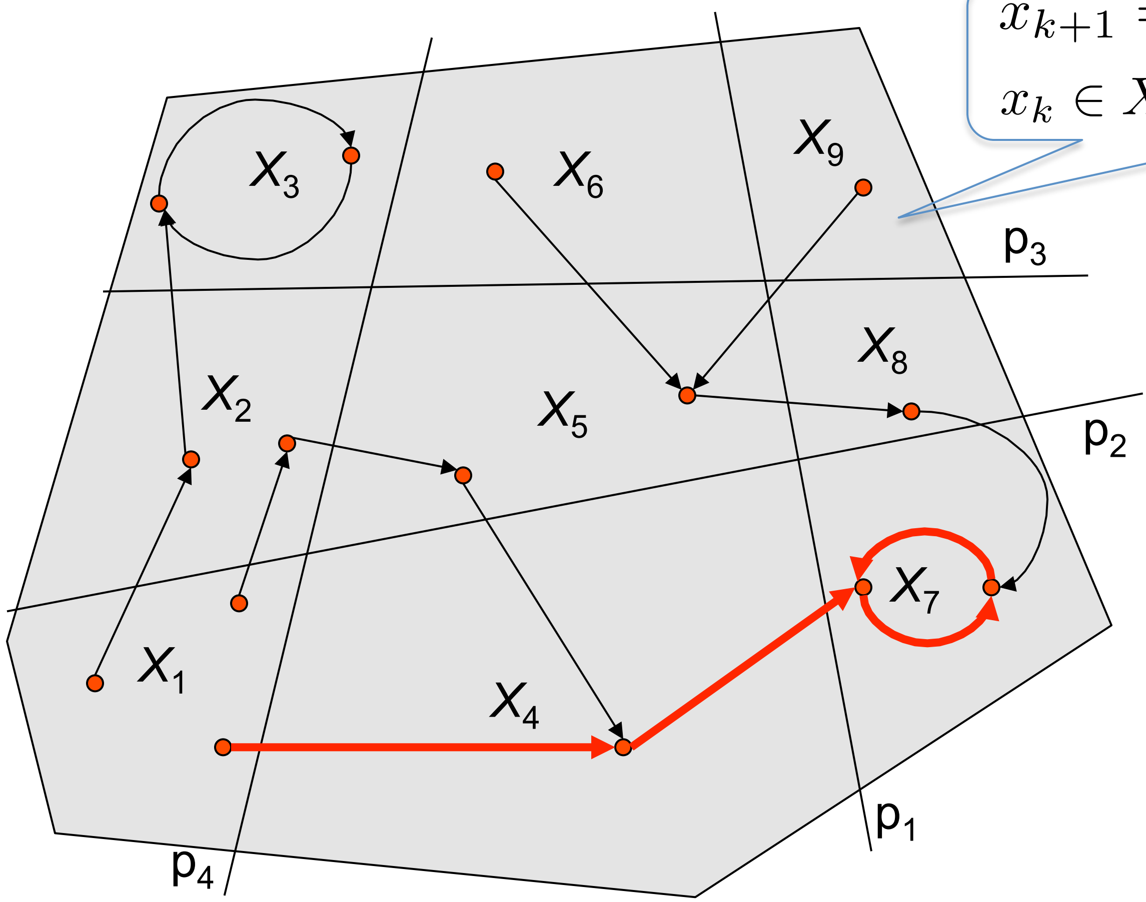
# Piecewise Affine (PWA) Systems

## Semantics



# Piecewise Affine (PWA) Systems

## Semantics




$$x_{k+1} = A_l x_k + B_l u_k + b_l$$

$$x_k \in X_l \qquad u_k \in U_l$$

$$\Pi = \{p_1, p_2, p_3, p_4\}$$

Word:

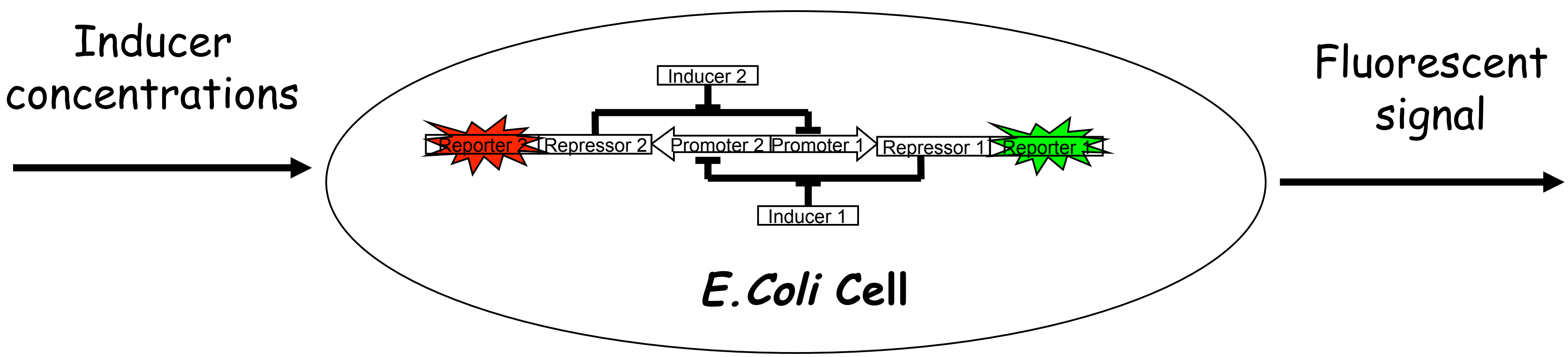
| $p_4$ | $p_2$ | $p_1$ | $p_1$ | $p_1$ | ... |
|---|---|---|---|---|---|
| $p_2$ | | $p_2$ | $p_2$ | $p_2$ | |

Can be checked against the satisfaction of LTL formulae over $\Pi$

# Piecewise Affine (PWA) Systems

## Why PWA systems?

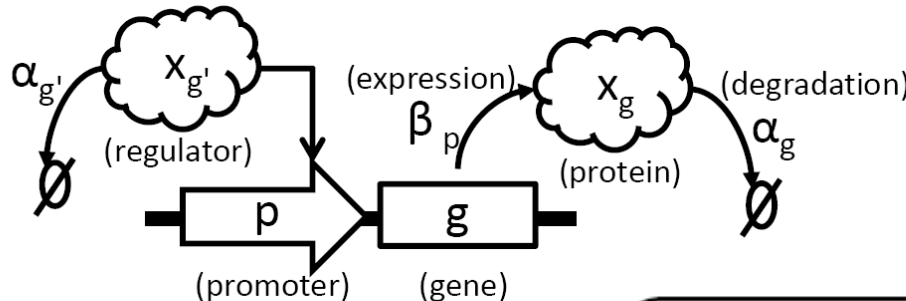
- PWA systems can approximate nonlinear systems with arbitrary accuracy [Lin and Unbehauen, 1992].
- Under mild assumptions, PWA systems are equivalent with several other classes of hybrid systems, including mixed logical dynamical (MLD), linear complementarity (LC), extended linear complementarity (ELC), and maxmin-plus-scaling (MMPS) systems [Heemels et al., 2001, Geyer et al., 2003].
- There exist tools for the identification of PWA systems from experimental data [Paoletti, Juloski, Ferrari-Trecate, Vidal, 2007]

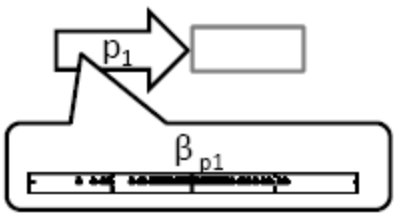


# Piecewise Affine (PWA) Systems

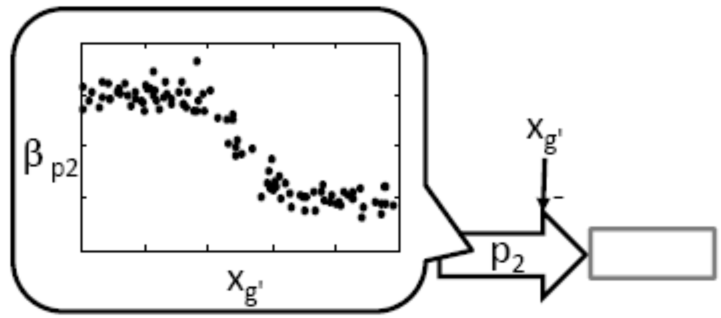
## Why PWA systems?

- Specific classes of PWA models can be directly derived from first principles




(a) Constitutive promoter




(b) Regulated promoter

$$\beta_p \in B_p^c$$

$$x_g(k+1) = \alpha_g x_g(k) + \beta_p.$$

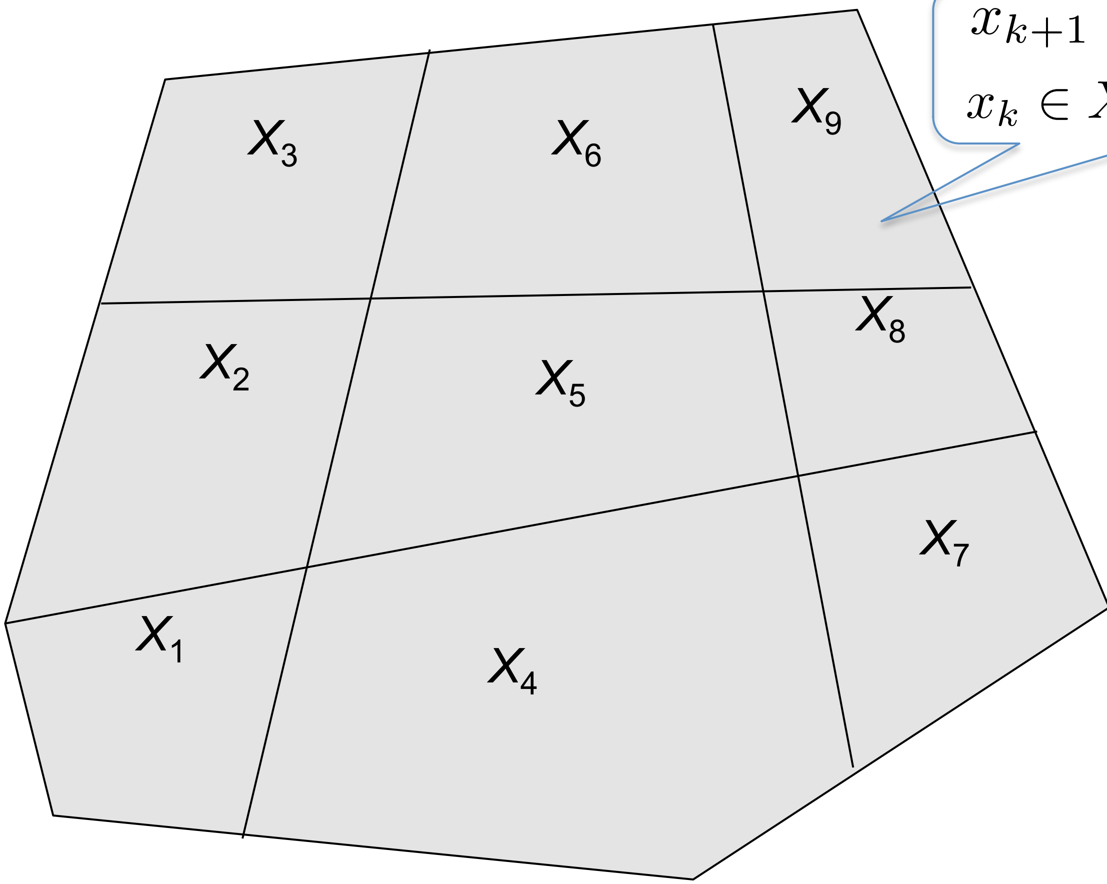
$$\beta_p \in B_p(x_{g'})$$

- PWA systems admit finite quotients and can be formally analyzed / controlled

# Outline

1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

# Verification of PWA Systems with Fixed Parameters



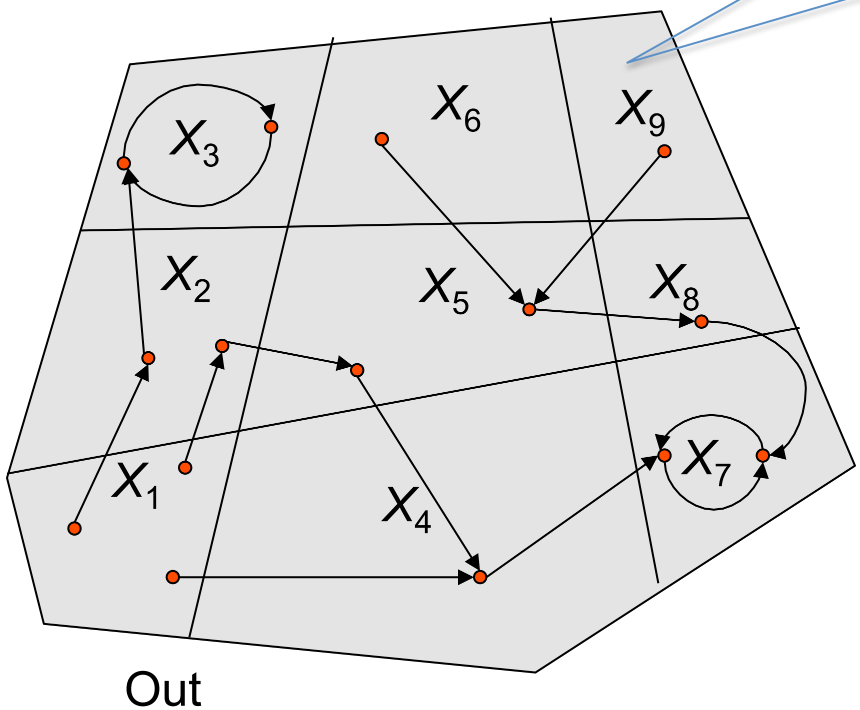

$$x_{k+1} = A_l x_k + b_l$$
$$x_k \in X_l \qquad l \in L$$

$$\mathcal{X} = \cup_{l \in L} X_l$$

**Problem formulation: Find the largest subset of $\mathcal{X}$ such that all trajectories originating there satisfy an LTL formula $\phi$ over $L$ while always staying inside $\mathcal{X}$**

# Verification of PWA Systems with Fixed Parameters

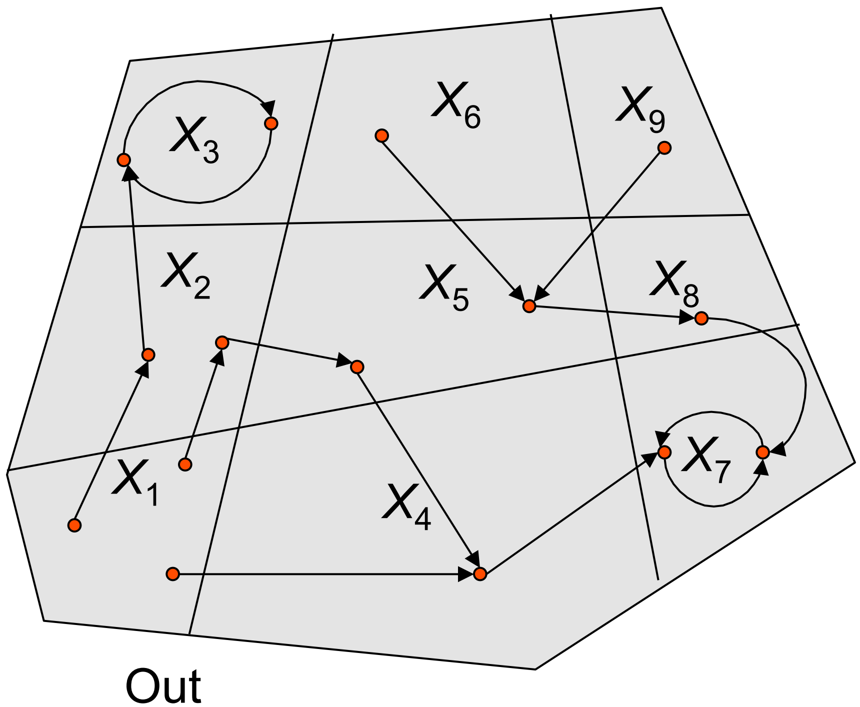
$$x_{k+1} = A_l x_k + b_l$$
$$x_k \in X_l \qquad l \in L$$

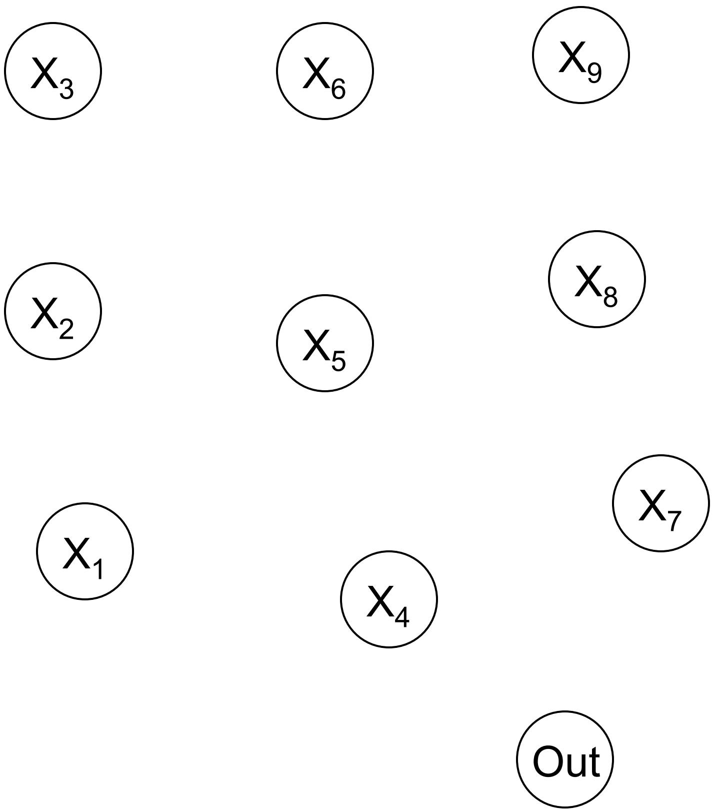



**Embed the PWA system into an infinite deterministic transition system $T_e$ with set of observations $L \cup \{Out\}$**

# Verification of PWA Systems with Fixed Parameters

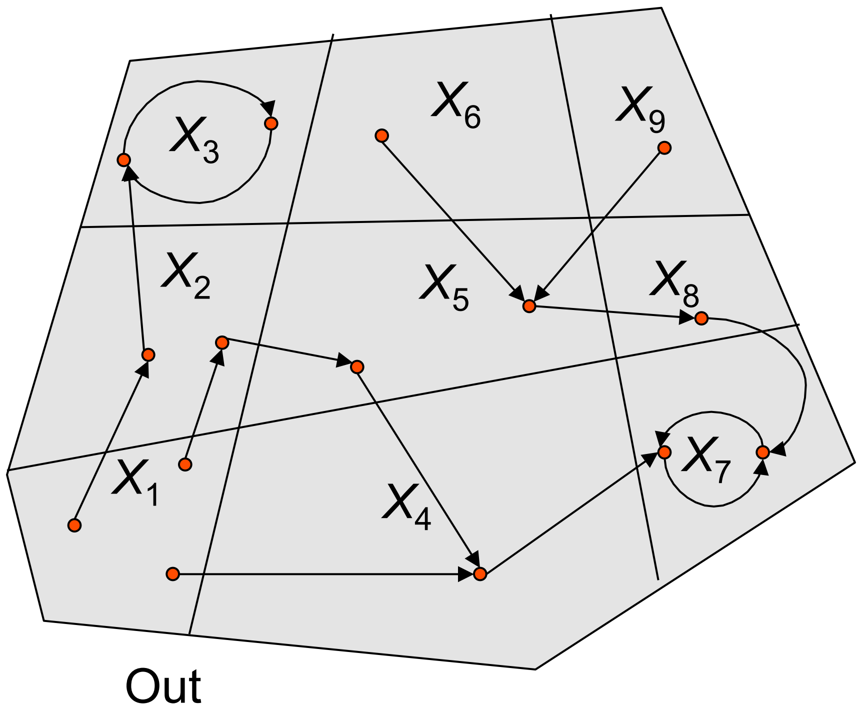
**Construct the observational equivalence quotient** $T_e/_{\sim}$

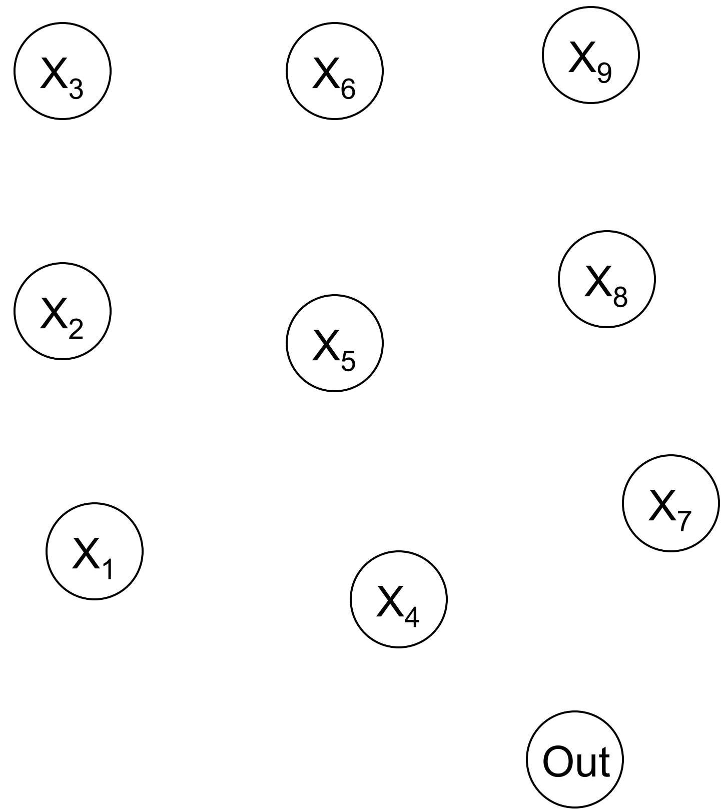


# Verification of PWA Systems with Fixed Parameters

**Construct the observational equivalence quotient** $T_e/_\sim$



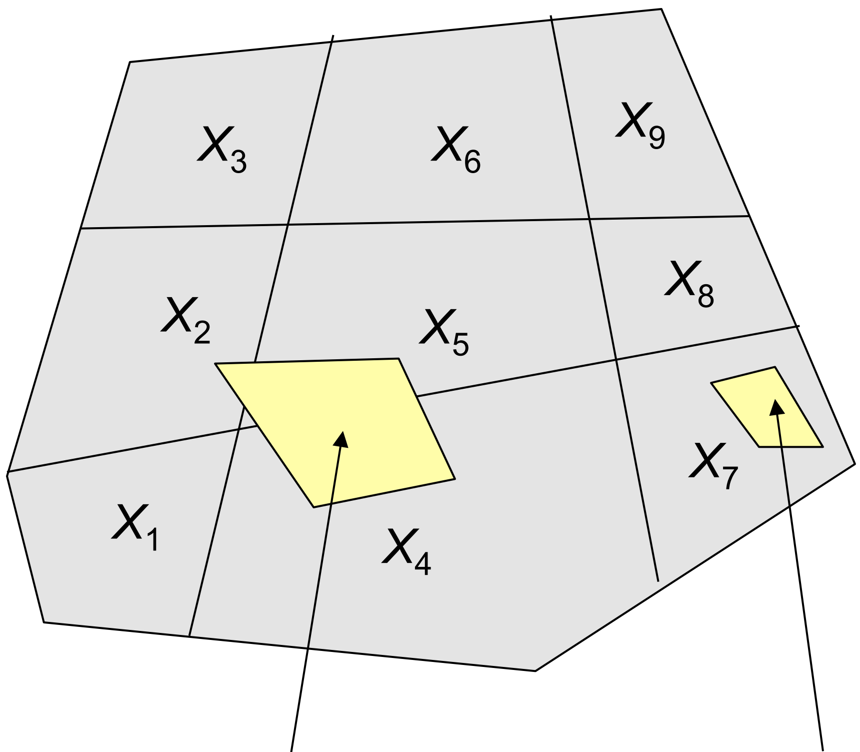

$$(X, X') \in \rightarrow_{e,\sim} \text{ if and only if } \mathrm{Post}_{T_e}(con(X)) \cap con(X') \neq \emptyset$$

$\mathrm{Post}_{T_e}$ is computable

$$\mathrm{Post}_{T_e}(con(X_l)) = A_l X_l + b_l$$

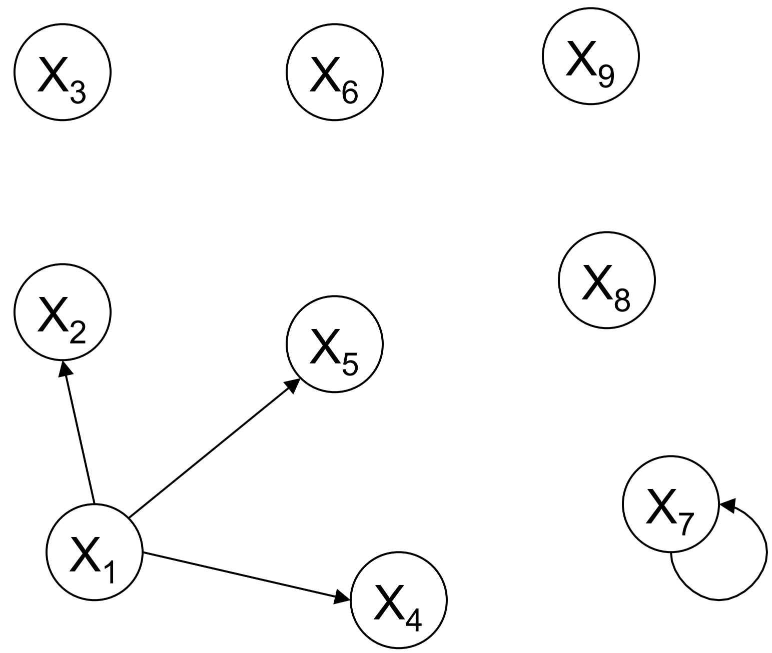
# Verification of PWA Systems with Fixed Parameters

## Construct the observational equivalence quotient $T_e/_\sim$




$\text{Post}_{T_e}(con(X_7)) = A_7 X_7 + b_7$

$\text{Post}_{T_e}(con(X_1)) = A_1 X_1 + b_1$

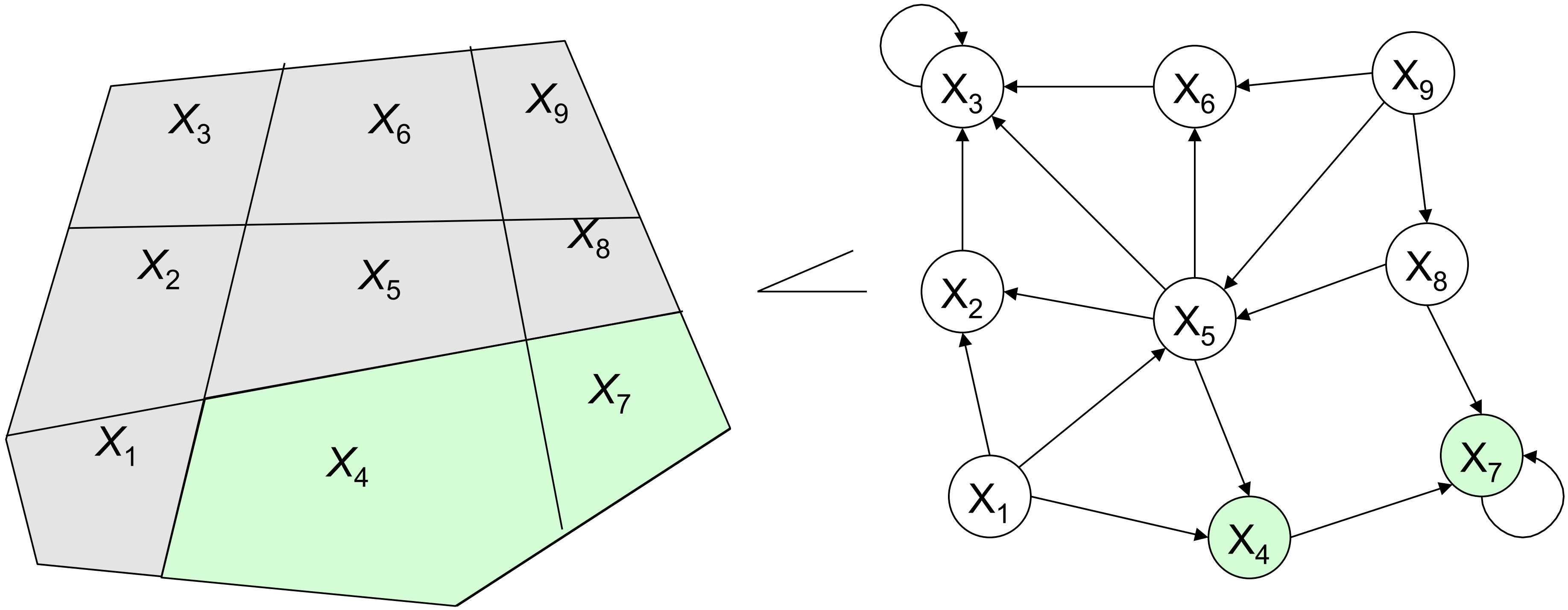



$T_e/_\sim$ is nondeterministic

$(X, X') \in \rightarrow_{e,\sim}$ if and only if $\text{Post}_{T_e}(con(X)) \cap con(X') \neq \emptyset$

# Verification of PWA Systems with Fixed Parameters

**Solve the problem on $T_e/_\sim$**




$T_e/_\sim$ simulates $T_e$

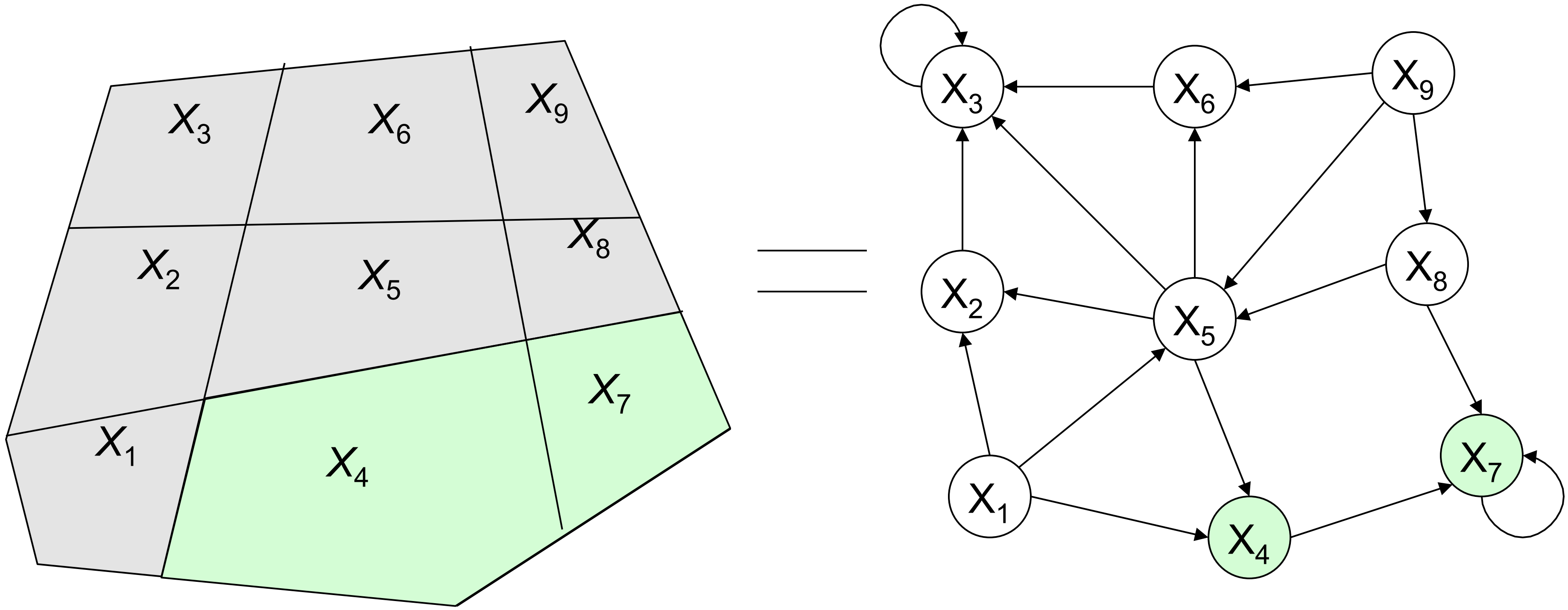
LTL formula "◇□7"

1) solve the problem on $T_e/_\sim$

2) map the solution to $T_e$ -> satisfying region for $T_e$ but not the largest

# Verification of PWA Systems with Fixed Parameters

**Solve the problem on $T_e/_\sim$**



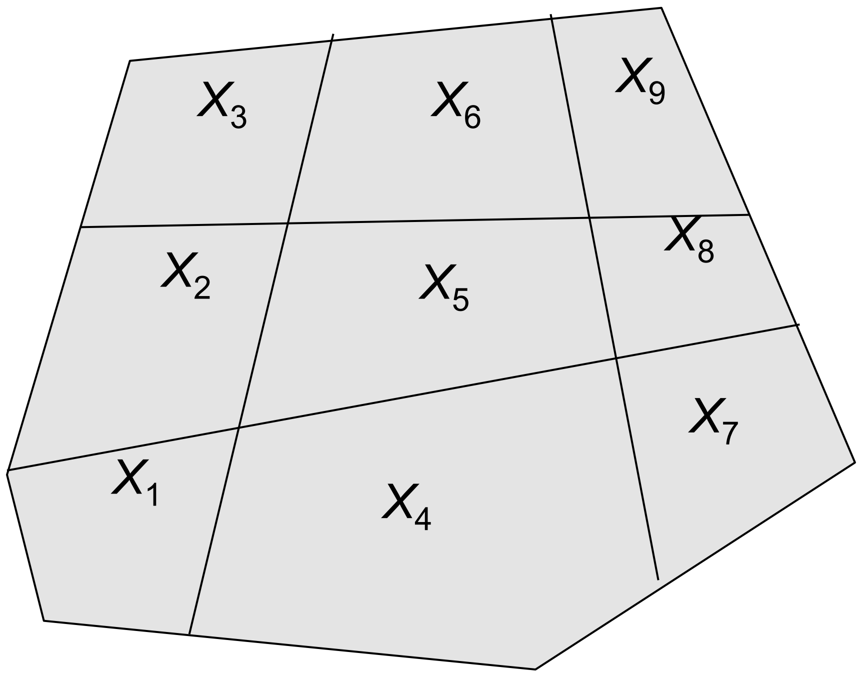

if $T_e/_\sim$ was a bisimulation quotient

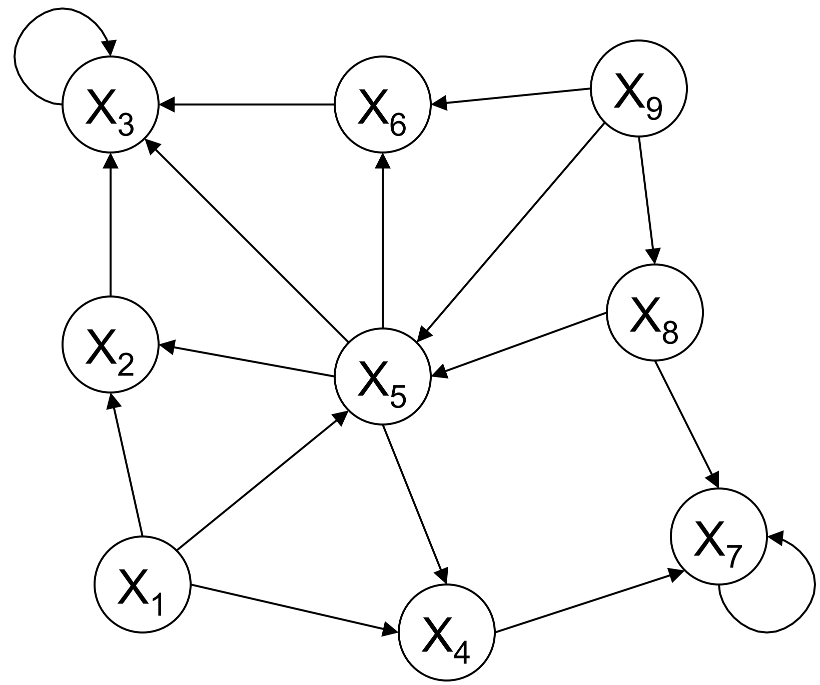
LTL formula “◇□7”

solving the problem on $T_e/_\sim$ is equivalent to solving it on $T_e$

# Verification of PWA Systems with Fixed Parameters

## Bisimulation algorithm



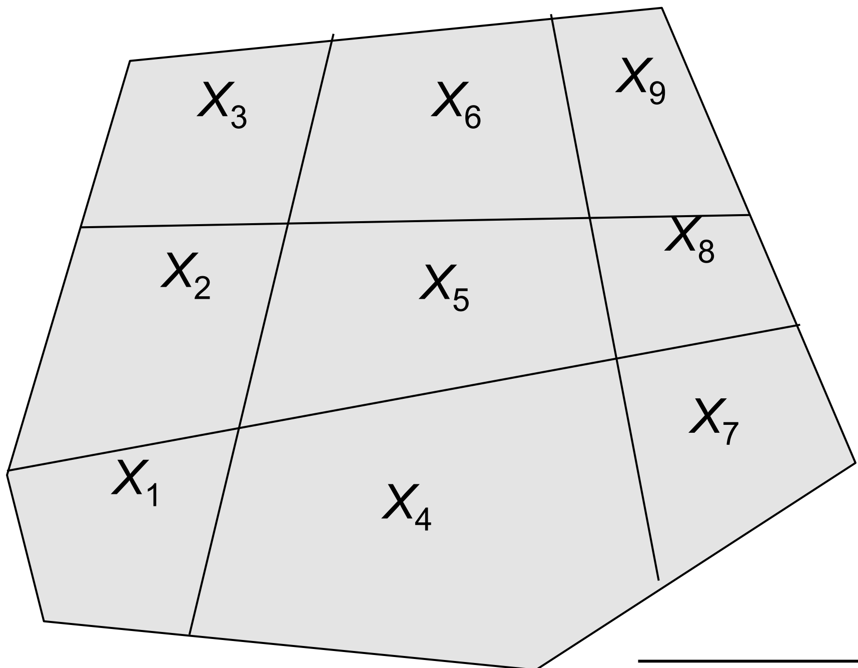



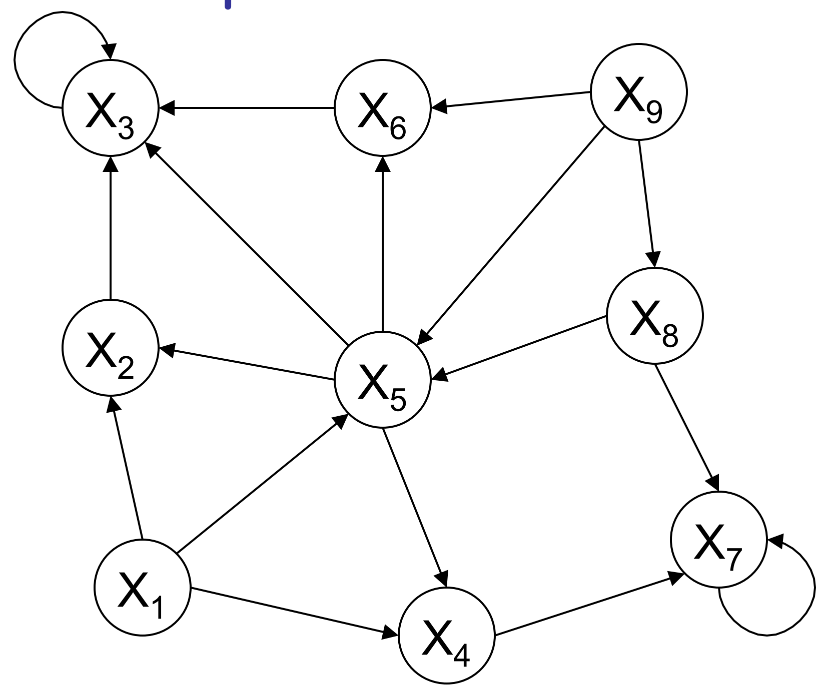

**Algorithm 1** $\sim$=BISIMULATION($\mathcal{T}$): Coarsest bisimulation $\sim$ of $\mathcal{T}$

Initialize $\sim$ with observational equivalence
**while** there exist $X, X' \in Q/_\sim$ such that $\emptyset \subset con(X) \cap Pre_{\mathcal{T}}(con(X')) \subset con(X)$ **do**
    Construct state $X_1$ such that $con(X_1) := con(X) \bigcap Pre_{\mathcal{T}}(con(X'))$;
    Construct state $X_1$ such that $con(X_2) := con(X) \setminus Pre_{\mathcal{T}}(con(X'))$;
    $Q/_\sim := Q/_\sim \setminus \{X\} \bigcup \{X_1, X_2\}$;
**end while**
return $\sim$;

# Verification of PWA Systems with Fixed Parameters

## Can the bisimulation algorithm be used to solve the problem?



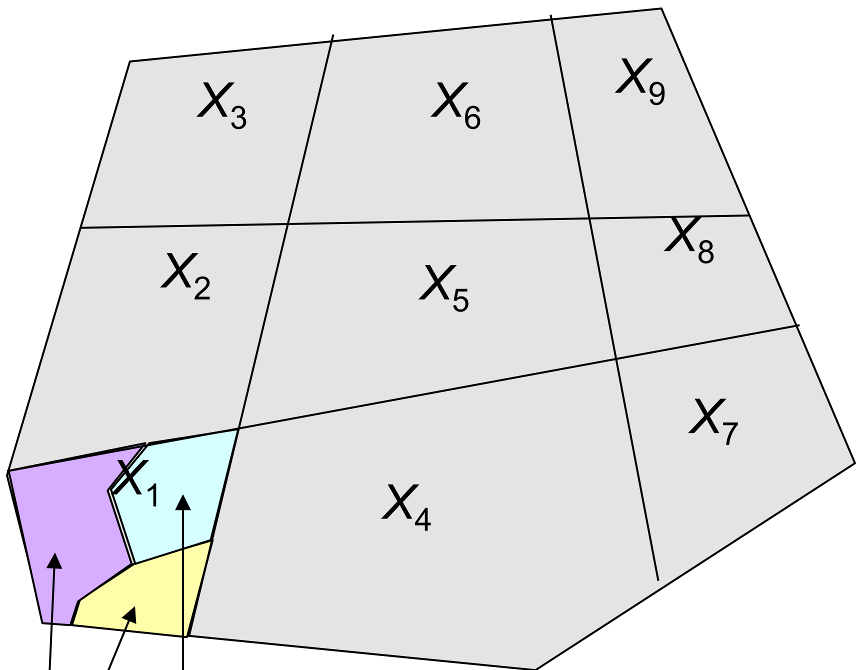

**Algorithm 1** $\sim$=BISIMULATION($\mathcal{T}$): Coarsest bisimulation $\sim$ of $\mathcal{T}$

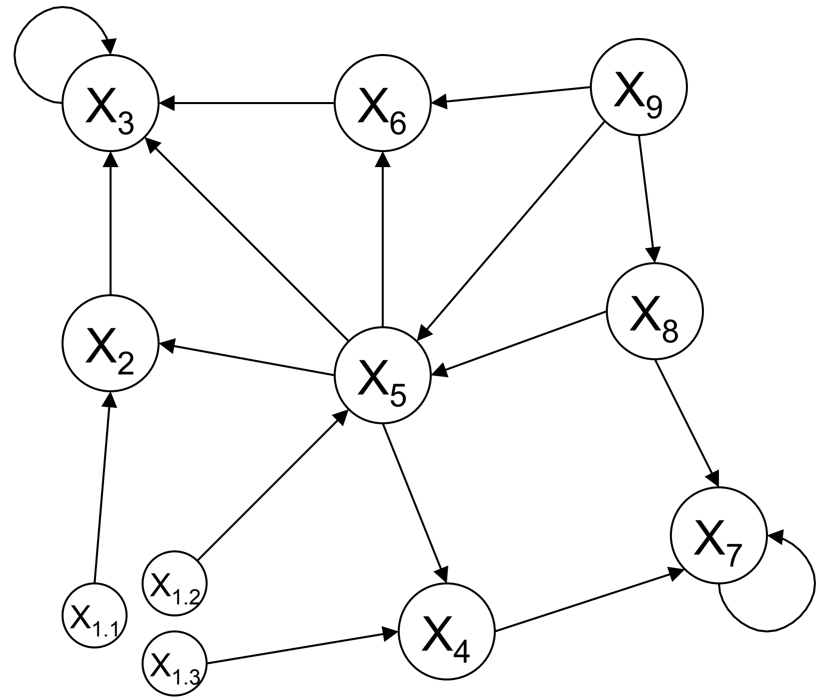
Initialize $\sim$ with observational equivalence
**while** there exist $X, X' \in Q/_{\sim}$ such that $\emptyset \subset con(X) \cap Pre_{\mathcal{T}}(con(X')) \subset con(X)$ **do**
    Construct state $X_1$ such that $con(X_1) := con(X) \bigcap Pre_{\mathcal{T}}(con(X'))$;
    Construct state $X_1$ such that $con(X_2) := con(X) \setminus Pre_{\mathcal{T}}(con(X'))$;
    $Q/_{\sim} := Q/_{\sim} \setminus \{X\} \bigcup \{X_1, X_2\}$;
**end while** ← Construct and model check the quotient
return $\sim$;


A. Chutinan and B. H. Krogh, "Verification of infinite-state dynamic systems using approximate quotient transition systems," IEEE Transactions on automatic control, vol. 46, no. 9, pp. 1401–1410, 2001.

# Verification of PWA Systems with Fixed Parameters

**In principle, yes.**




**Algorithm 1** $\sim$=BISIMULATION($\mathcal{T}$): Coarsest bisimulation $\sim$ of $\mathcal{T}$

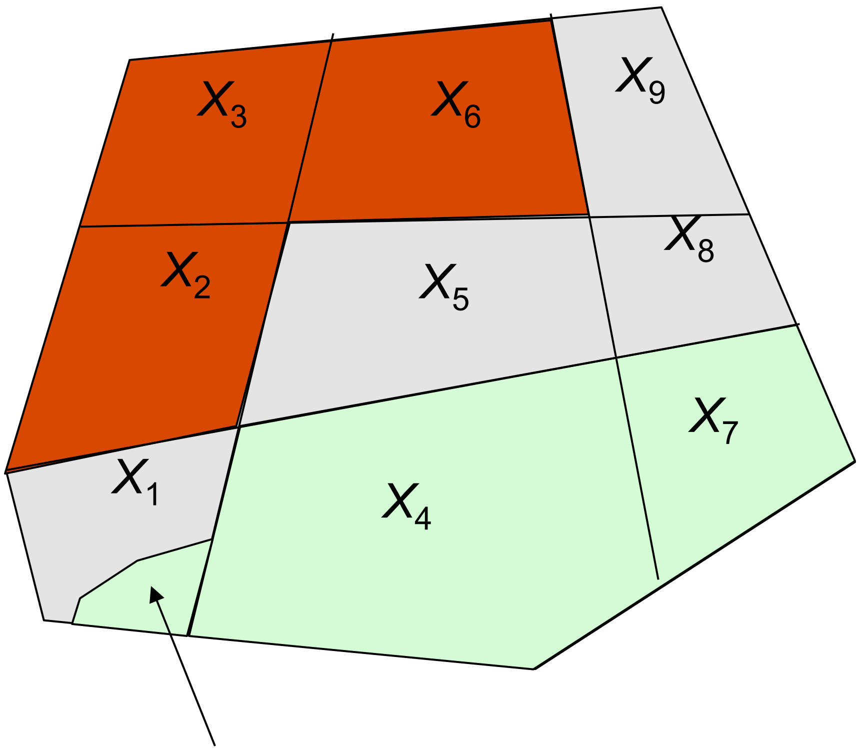
Initialize $\sim$ with observational equivalence
**while** there exist $X, X' \in Q/_\sim$ such that $\emptyset \subset con(X) \cap Pre_{\mathcal{T}}(con(X')) \subset con(X)$ **do**
  Construct state $X_1$ such that $con(X_1) := con(X) \bigcap Pre_{\mathcal{T}}(con(X'))$;
  Construct state $X_1$ such that $con(X_2) := con(X) \setminus Pre_{\mathcal{T}}(con(X'))$;
  $Q/_\sim := Q/_\sim \setminus \{X\} \bigcup \{X_1, X_2\}$;
**end while**
return $\sim$;

Construct and model check the quotient

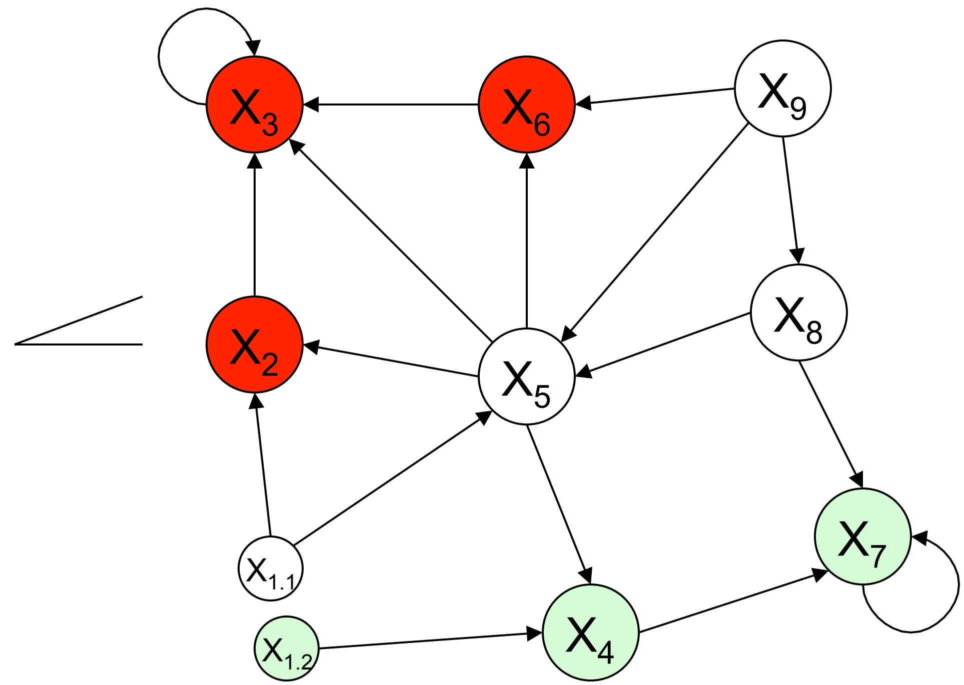
$$con(X_{l_1}) \cap Pre(con(X_{l_2})) = X_{l_1} \cap A_{l_1}^{-1}(con(X_{l_2}) - b_{l_1})$$

# Verification of PWA Systems with Fixed Parameters

## A better approach




$con(X_1) \cap Pre_{T_e}(con(X_4))$




LTL formula "◇□7"

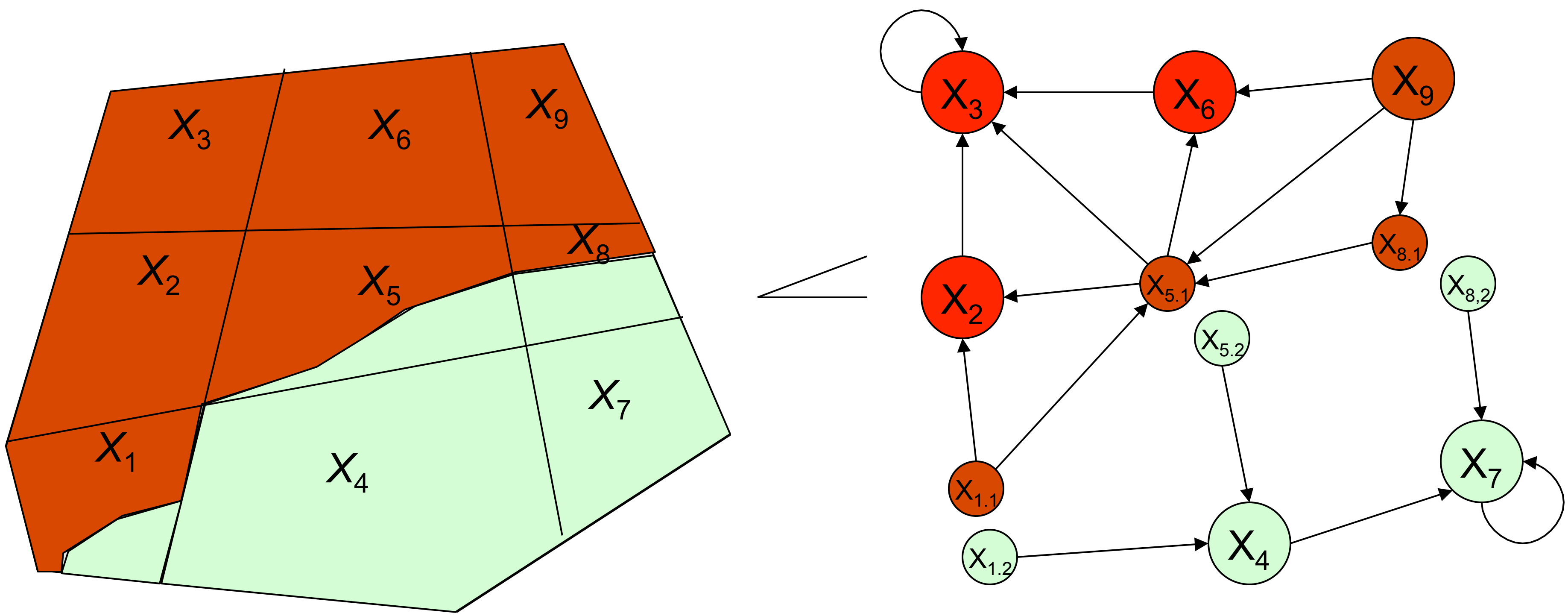
1) Expand the satisfying region
2) Do not refine satisfying regions
3) Construct satisfying sets for both the LTL formula and its negation simultaneously

Yordanov, B., Batt, G., and Belta, C., ECC '07

Yordanov, B. and Belta, C., IEEE Trans. Autom. Control, 2010

# Verification of PWA Systems with Fixed Parameters

## A better approach



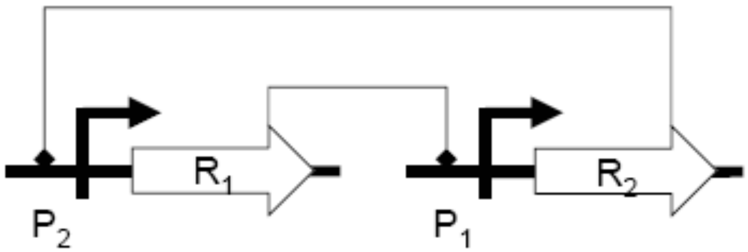

LTL formula “◇□7”

This procedure might terminate when the bisimulation algorithm does not - the idea of formula guided refinement (formula equivalent quotients).

Yordanov, B., Tumova, J., Belta, C., Cerna, I., and Barnat, J., CDC ‘10

# Verification of PWA Systems with Fixed Parameters

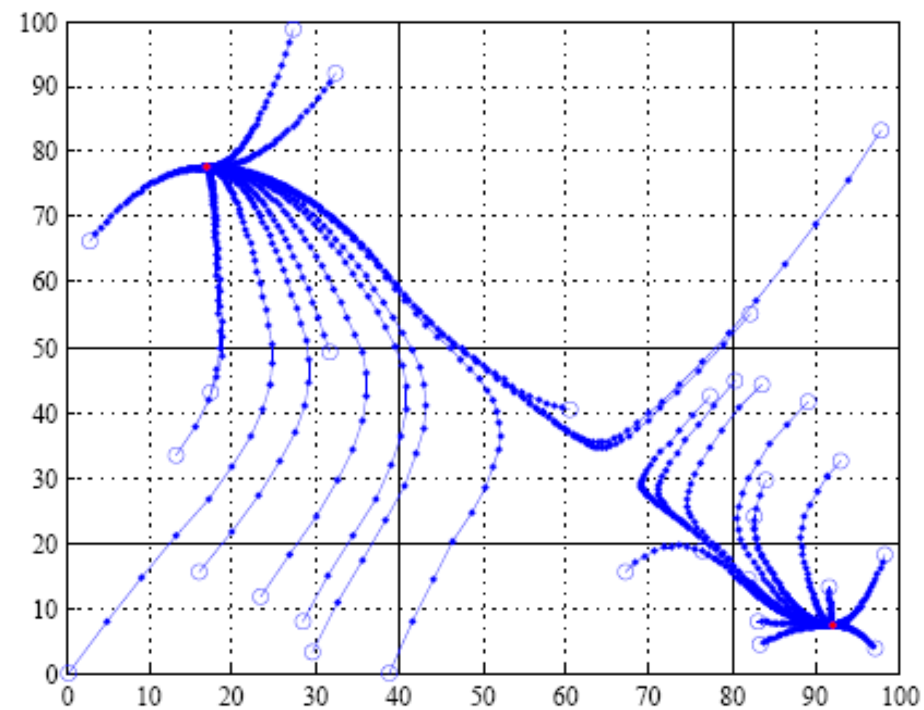
## Example: toggle switch - model with fixed parameters




Gardner et al., 2000

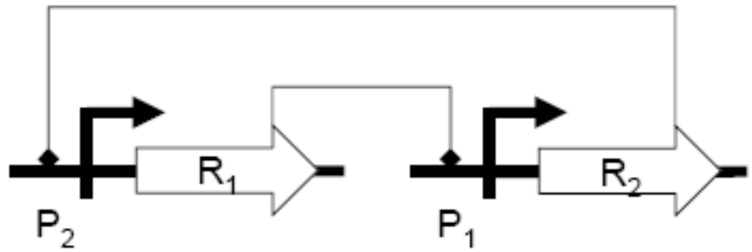
$\Diamond\Box(R_1>80 \wedge R_2<20)$

$\Diamond\Box(R_1<40 \wedge R_2>50)$

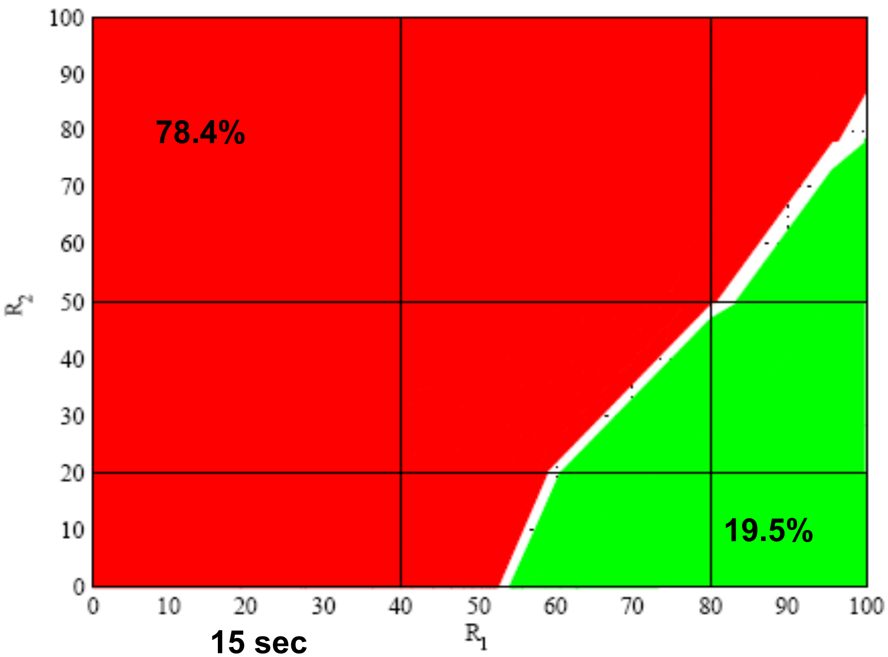


# Verification of PWA Systems with Fixed Parameters

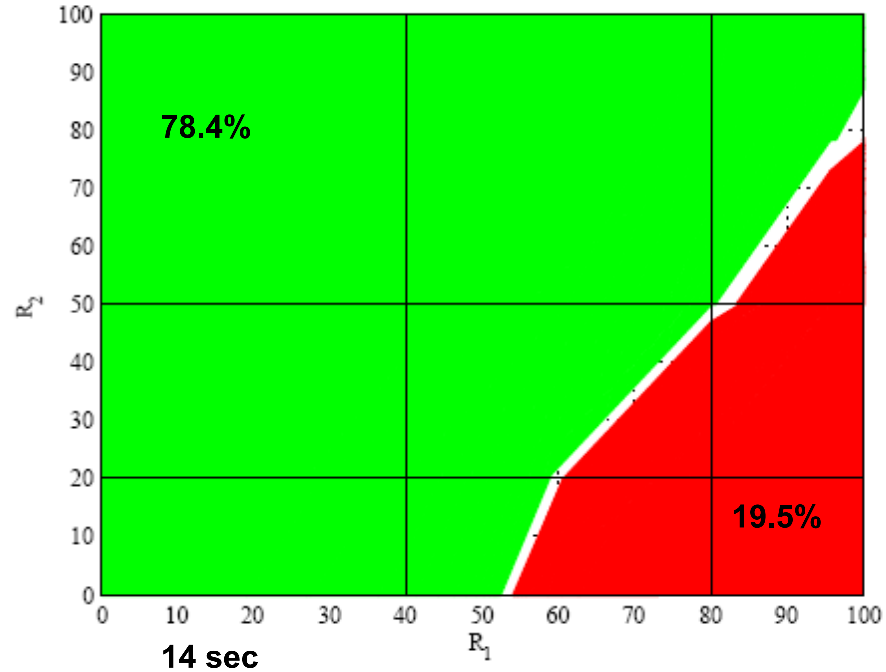
## Example: toggle switch - model with fixed parameters




$\Diamond\Box(R_1>80 \wedge R_2<20)$

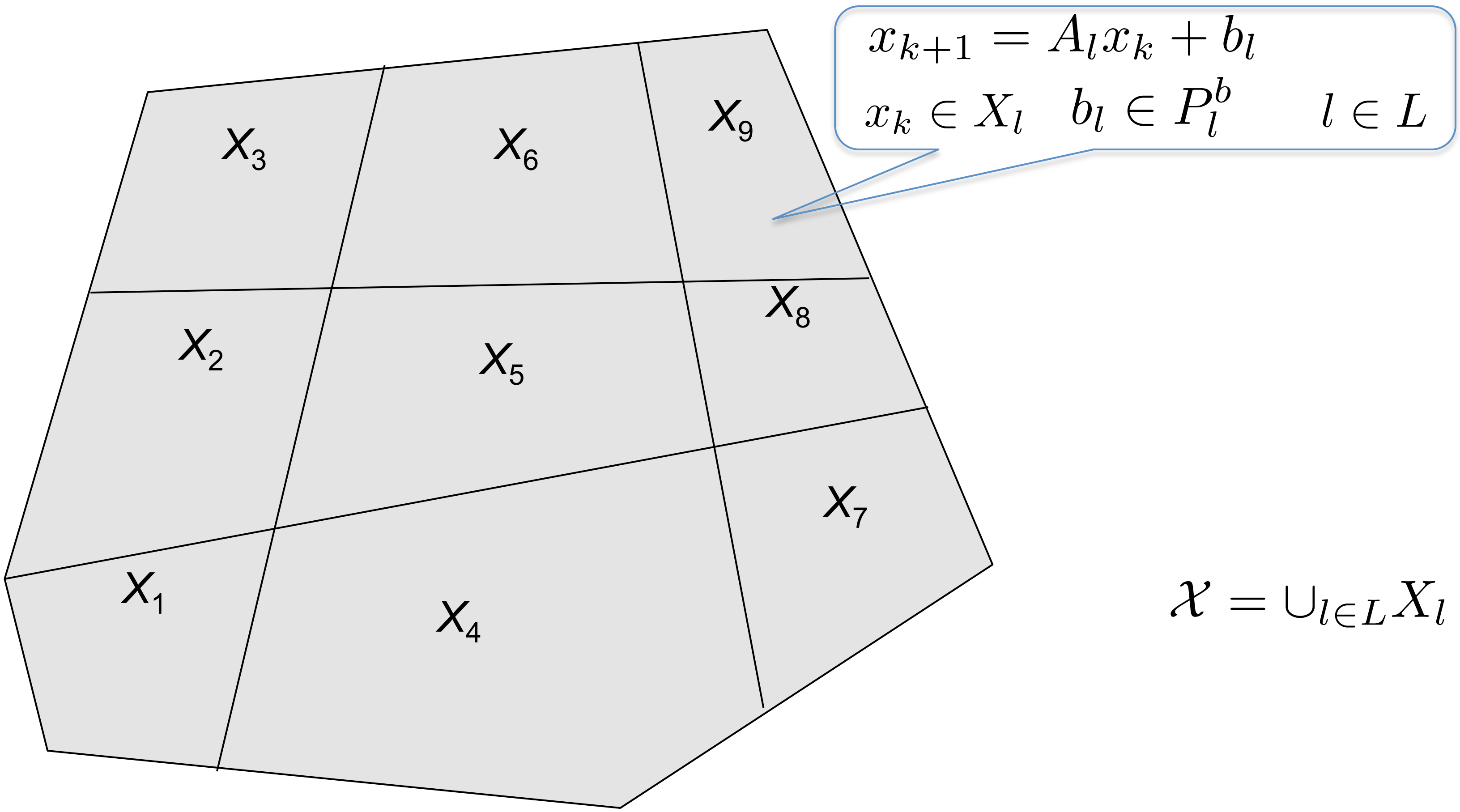



$\Diamond\Box(R_1<40 \wedge R_2>50)$




Matlab tool: "FaPAS"
(hyness.bu.edu/software)

# Verification of PWA Systems with Additive Uncertainty

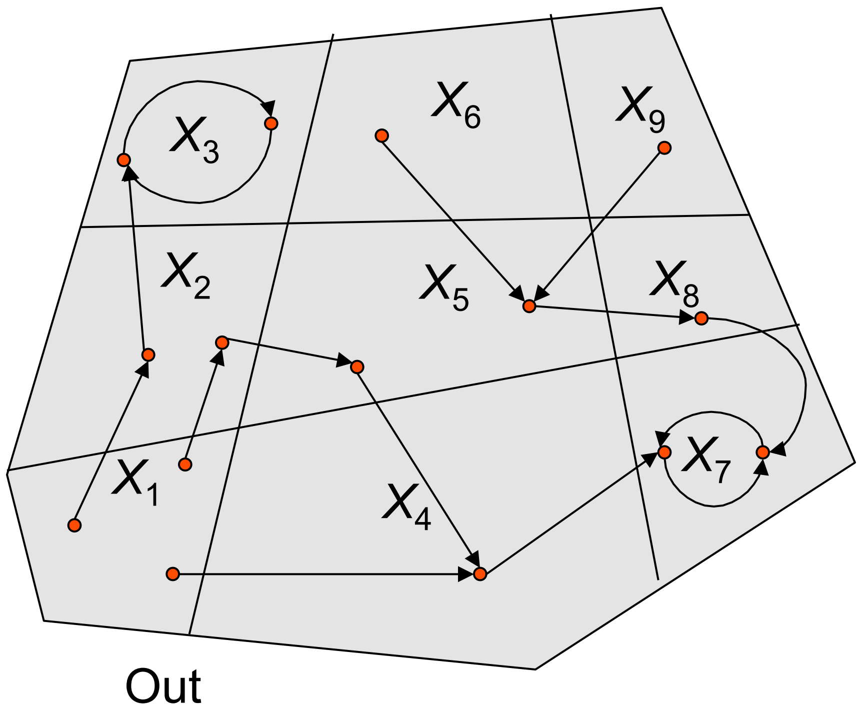


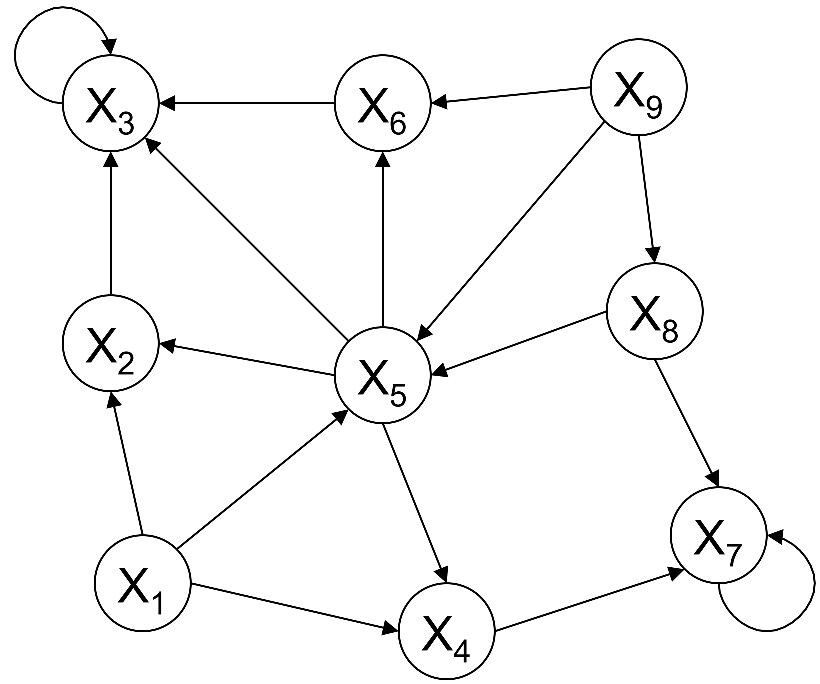

$$\mathcal{X} = \cup_{l \in L} X_l$$

**Problem formulation: Find the largest subset of $\mathcal{X}$ such that all trajectories originating there satisfy an LTL formula $\phi$ over $L$ while always staying inside $\mathcal{X}$**

# Verification of PWA Systems with Additive Uncertainty

**Construct the observational equivalence quotient** $T_e/_\sim$



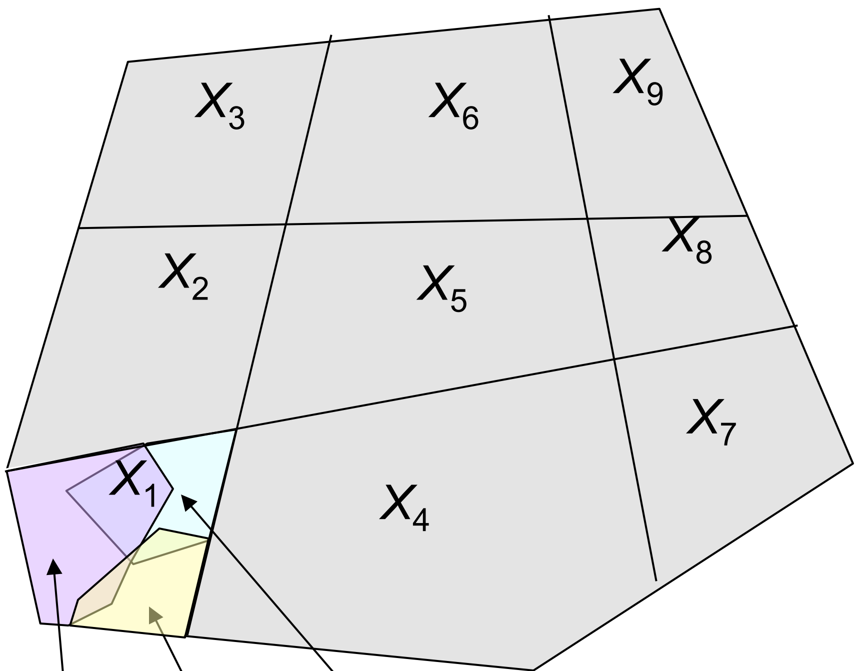

$$(X, X') \in \rightarrow_{e,\sim} \text{ if and only if } \mathrm{Post}_{T_e}(con(X)) \cap con(X') \neq \emptyset$$

$\mathrm{Post}_{T_e}$ is still computable and therefore $T_e/_\sim$ is computable

$$\mathrm{Post}_{T_e}(con(X_l)) = A_l X_l + P_l^b$$

# Verification of PWA Systems with Additive Uncertainty

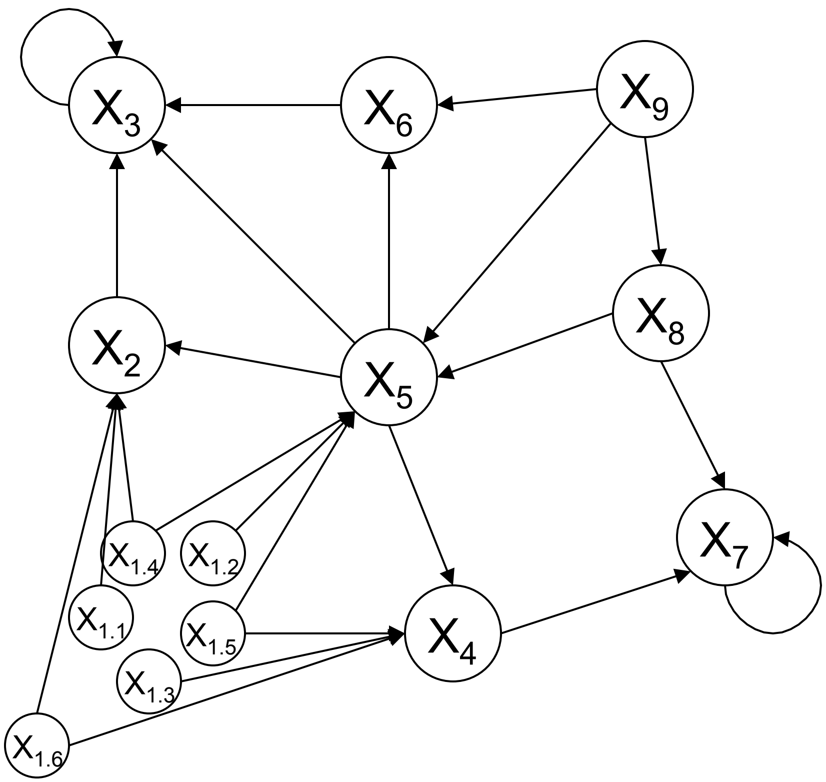
## Refinement




$con(X_1) \cap Pre_{T_e}(con(X_5))$

$con(X_1) \cap Pre_{T_e}(con(X_4))$

$con(X_1) \cap Pre_{T_e}(con(X_2))$

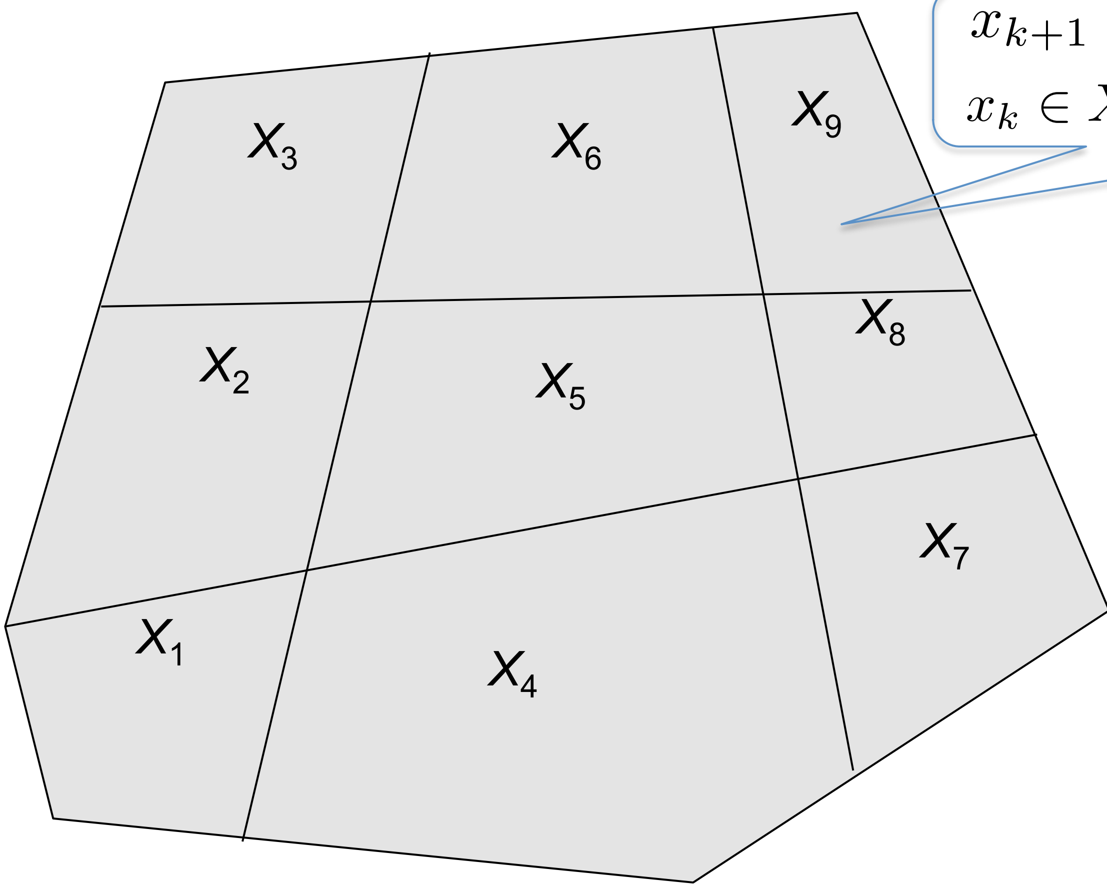



Pre is still computable

$$con(X_{l_1}) \cap Pre_{T_e}(con(X_{l_2}) = A_{l_1}^{-1}(con(X_{l_2}) - P^b_{l_1})$$

The only difference from the fixed parameter case is that there will be more states and more transitions (nondeterminism) in the quotient at each step of the refinement

# Verification of PWA Systems with Uncertain Parameters




$$x_{k+1} = A_l x_k + b_l$$

$$x_k \in X_l \quad A_l \in P_l^A \quad b_l \in P_l^b \quad l \in L$$

$$\mathcal{X} = \cup_{l \in L} X_l$$

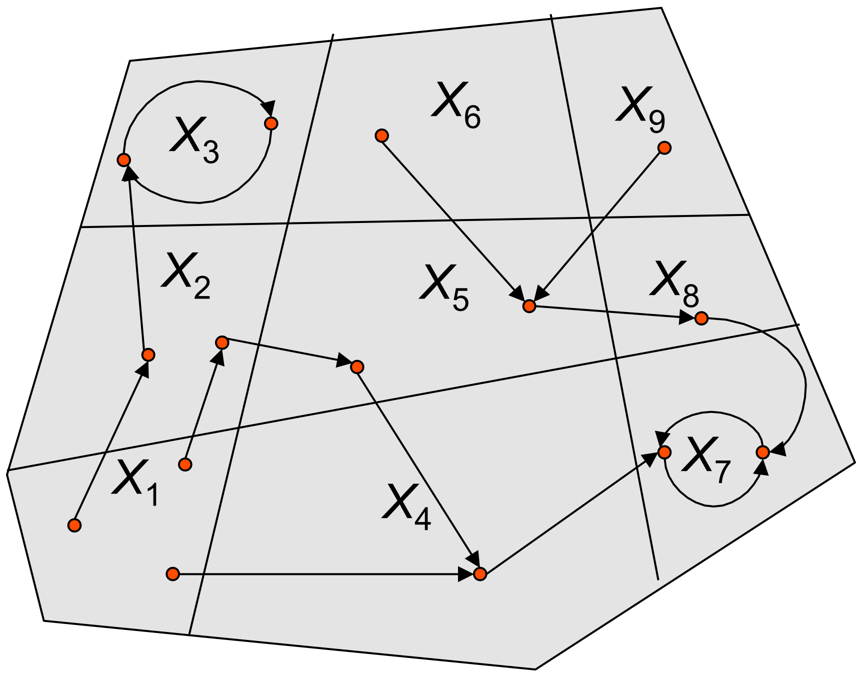
**Problem formulation: Find the largest subset of $\mathcal{X}$ such that all trajectories originating there satisfy an LTL formula $\phi$ over $L$ while always staying inside $\mathcal{X}$**

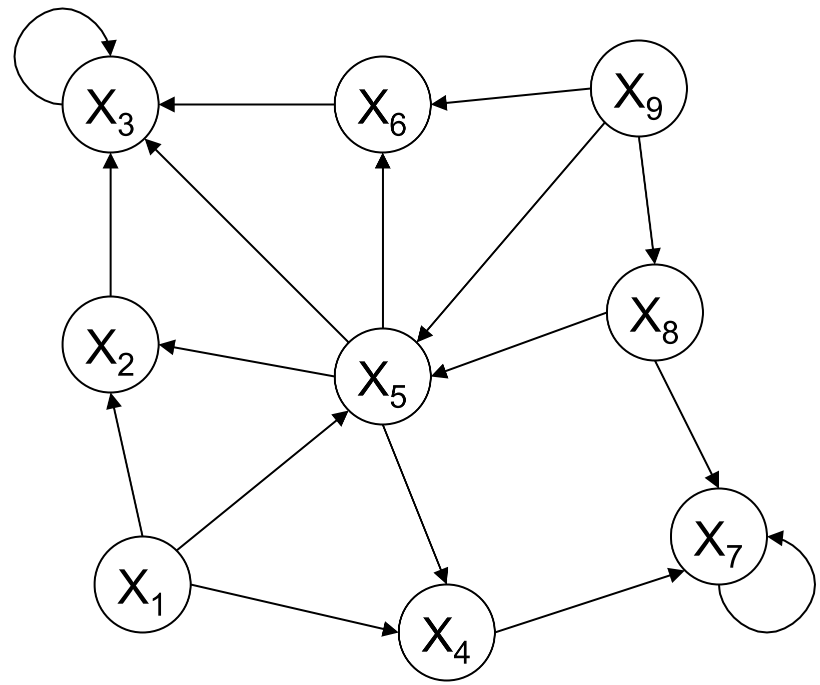
Yordanov, B. and Belta, C., ACC '08

Yordanov, B. and Belta, C., IEEE Trans. Autom. Control, 2010

# Verification of PWA Systems with Uncertain Parameters

**Construct an over-approximation of the observational equivalence quotient $T_e/_\sim$**




$$(X, X') \in \rightarrow_{e,\sim} \text{ if and only if } \mathrm{Post}_{T_e}(con(X)) \cap con(X') \neq \emptyset$$

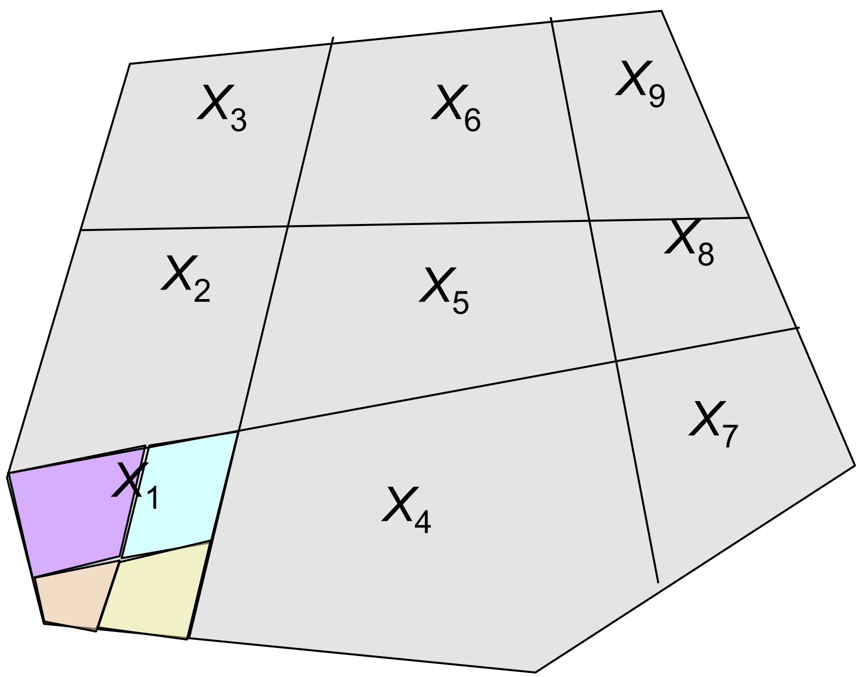
$\mathrm{Post}_{T_e}$ is not computable and therefore $T_e/_\sim$ is not computable

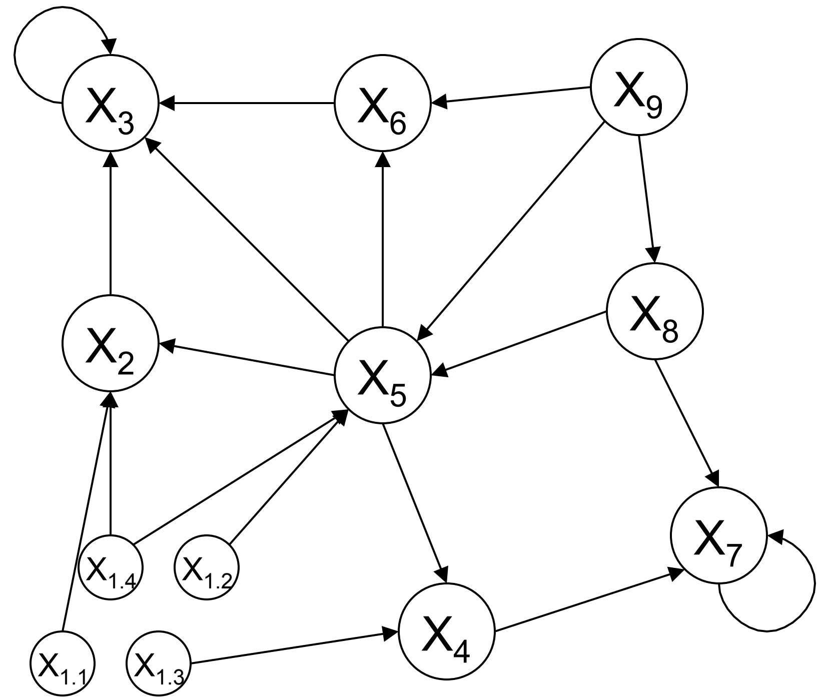
An over-aproximation $\overline{\mathrm{Post}}_{T_e}(con(X_l)) = hull(\{A_l X_l \mid A \in V(P_l^A)\}) + P_l^b$ is computable

An over-aproximation $\overline{T}_e/_\sim$ of $T_e/_\sim$ is computable

# Verification of PWA Systems with Uncertain Parameters

## Refinement




Pre is not computable and any partition scheme that does not capture the dynamics can be used, e.g., quad-tree partition.

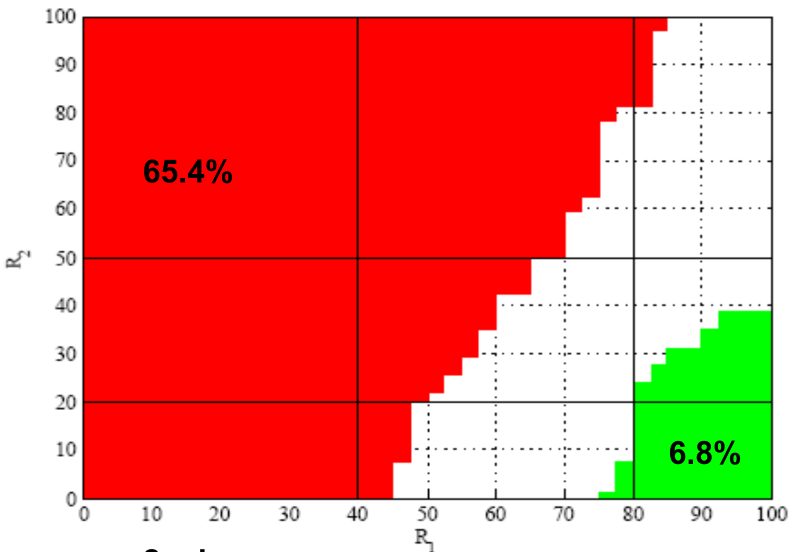
# Verification of PWA Systems

## Example: toggle switch - model with uncertain parameters

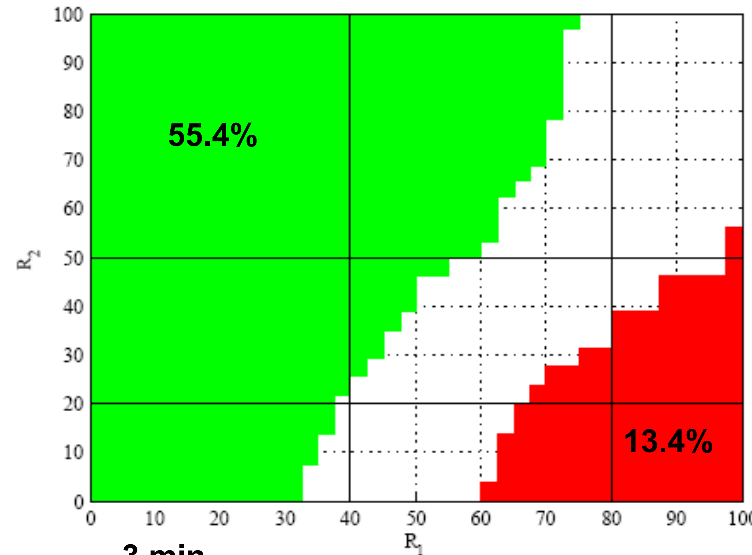
$\Diamond\Box(R_1>80 \wedge R_2<20)$

$\Diamond\Box(R_1<40 \wedge R_2>50)$

1% parameter noise

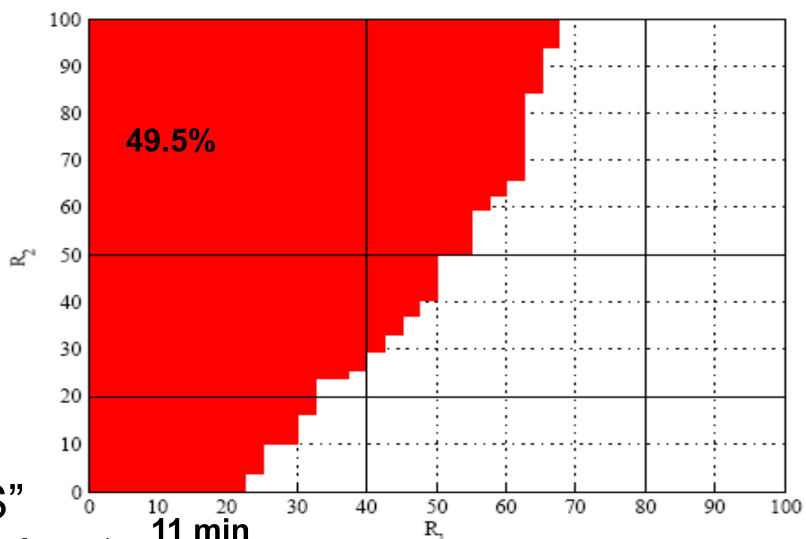



2 min

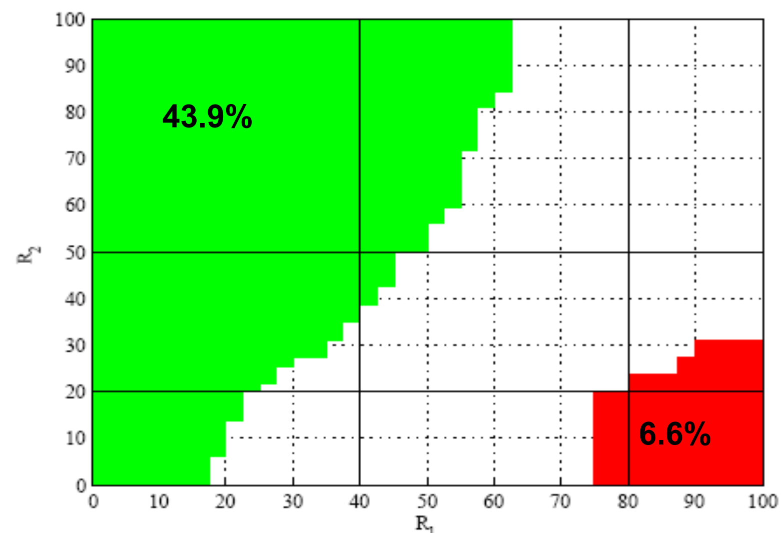



3 min

10% parameter noise




11 min



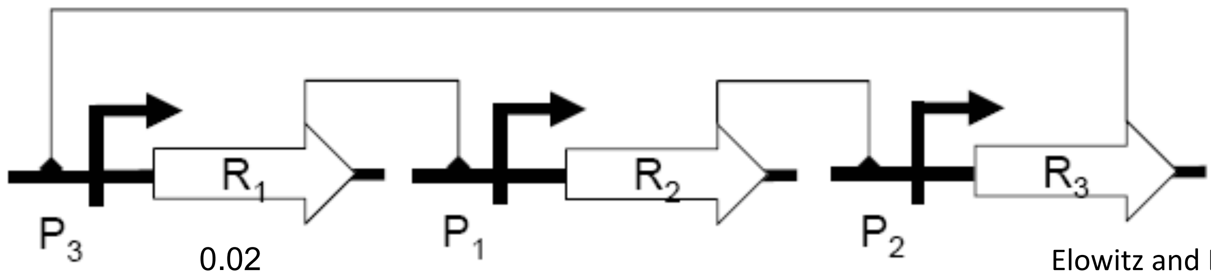

18 min

"FaPAS"

(hyness.bu.edu/software)

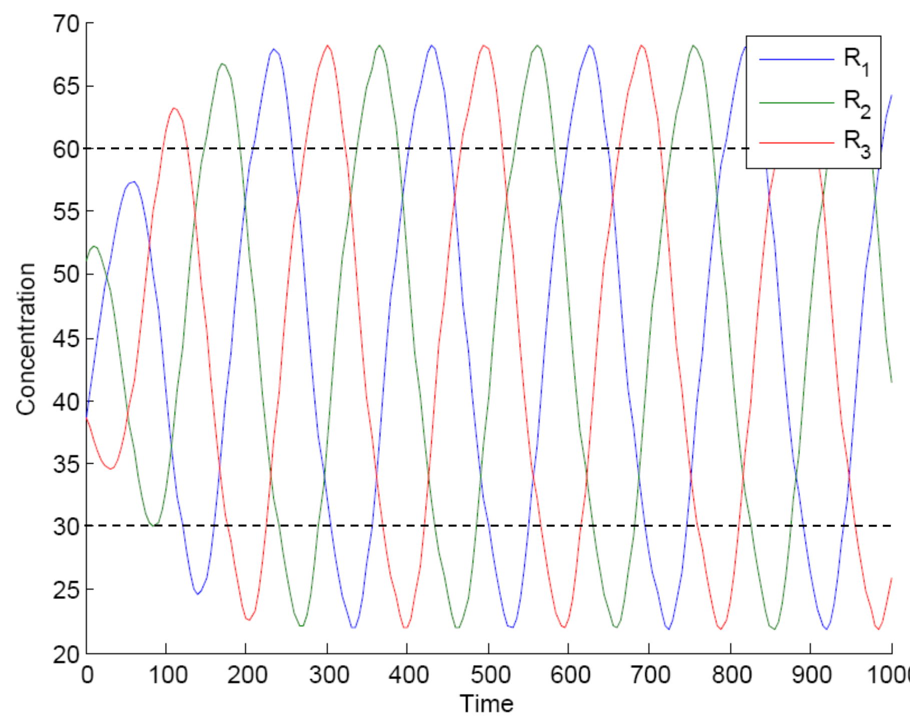
# Verification of PWA Systems

## Example: repressilator



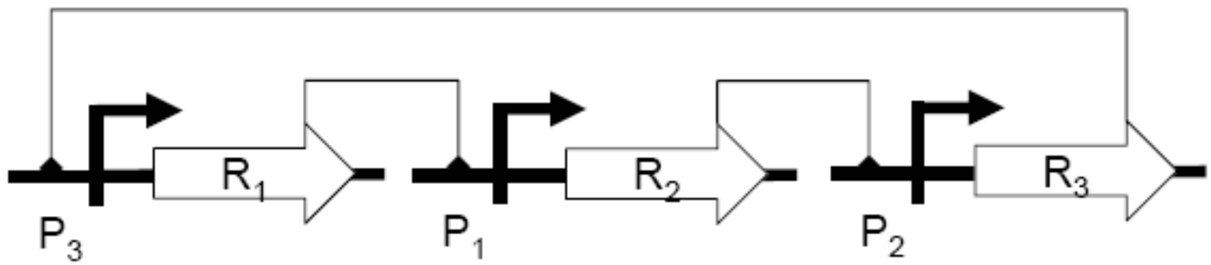

Elowitz and Leibler, 2000

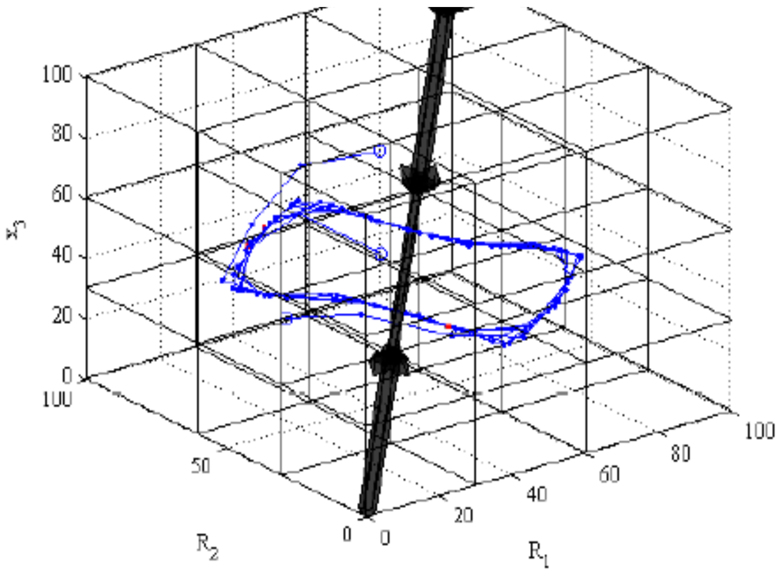
$\Box(\Diamond (R_3>60) \wedge \Diamond (R_3<30))$

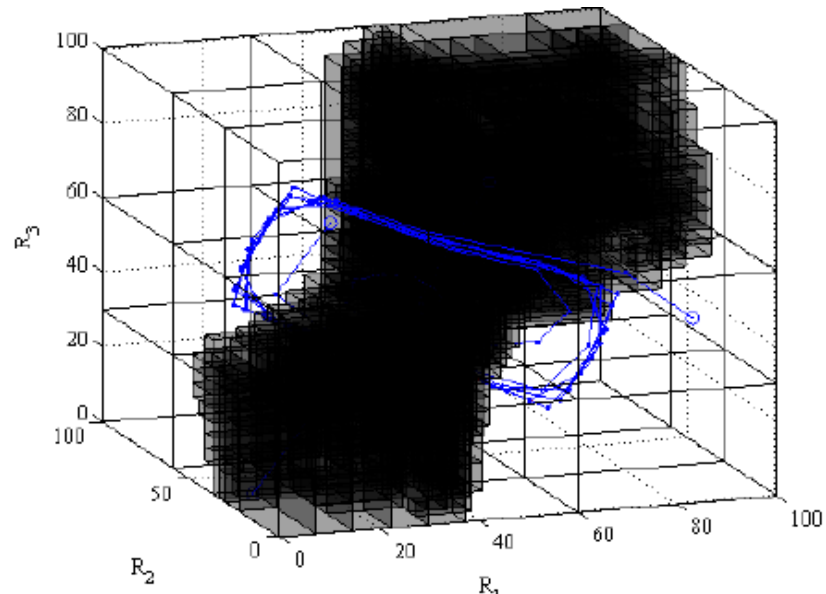


# Verification of PWA Systems

**Example: repressilator**



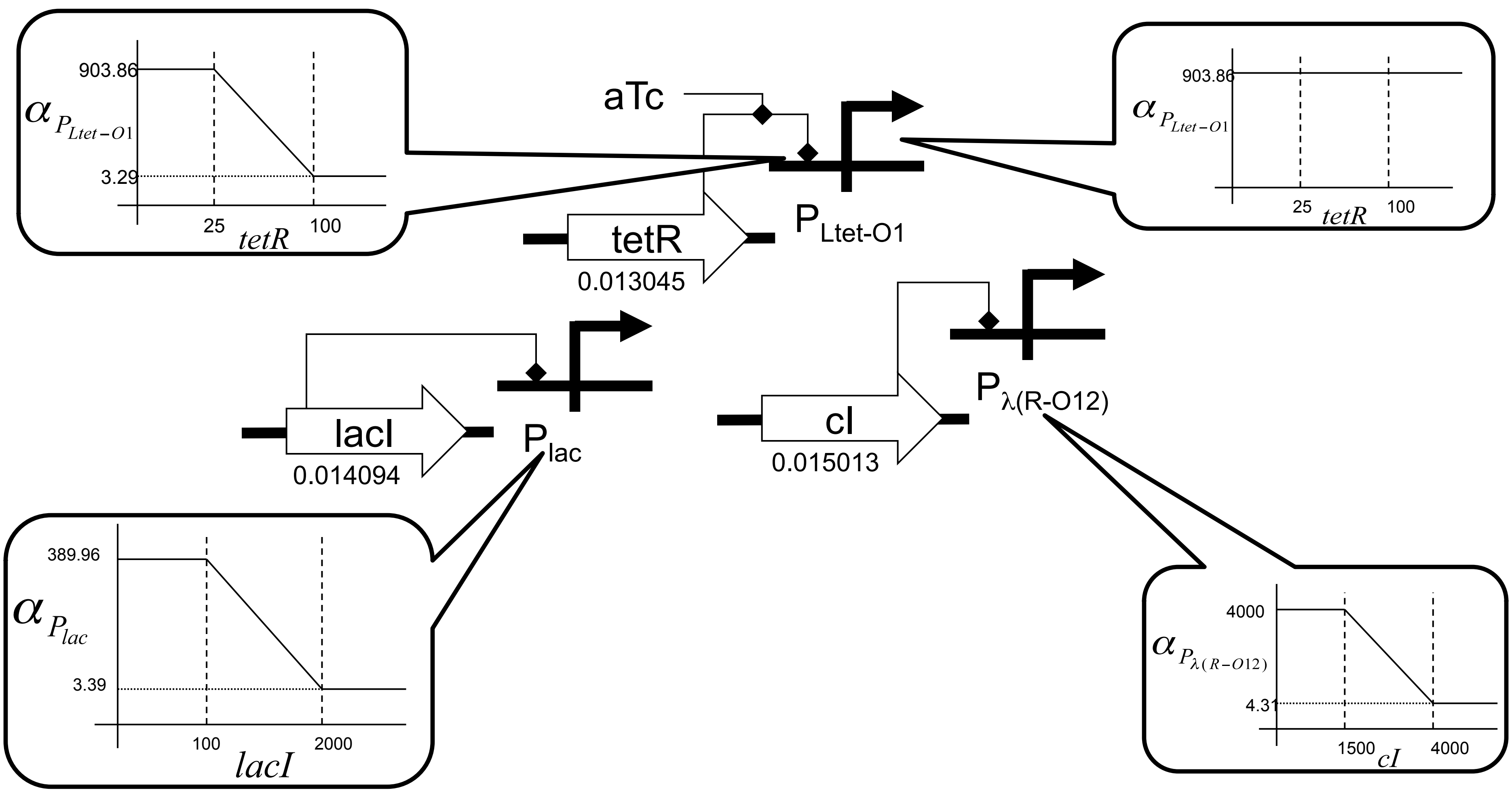

Fixed parameters:
99.8% of state space was satisfying
Computation time was 11 min

1% parameter noise:
69% of state space was satisfying
Computation time was 3 h

Matlab tool: "FaPAS"
(hyness.bu.edu/software)

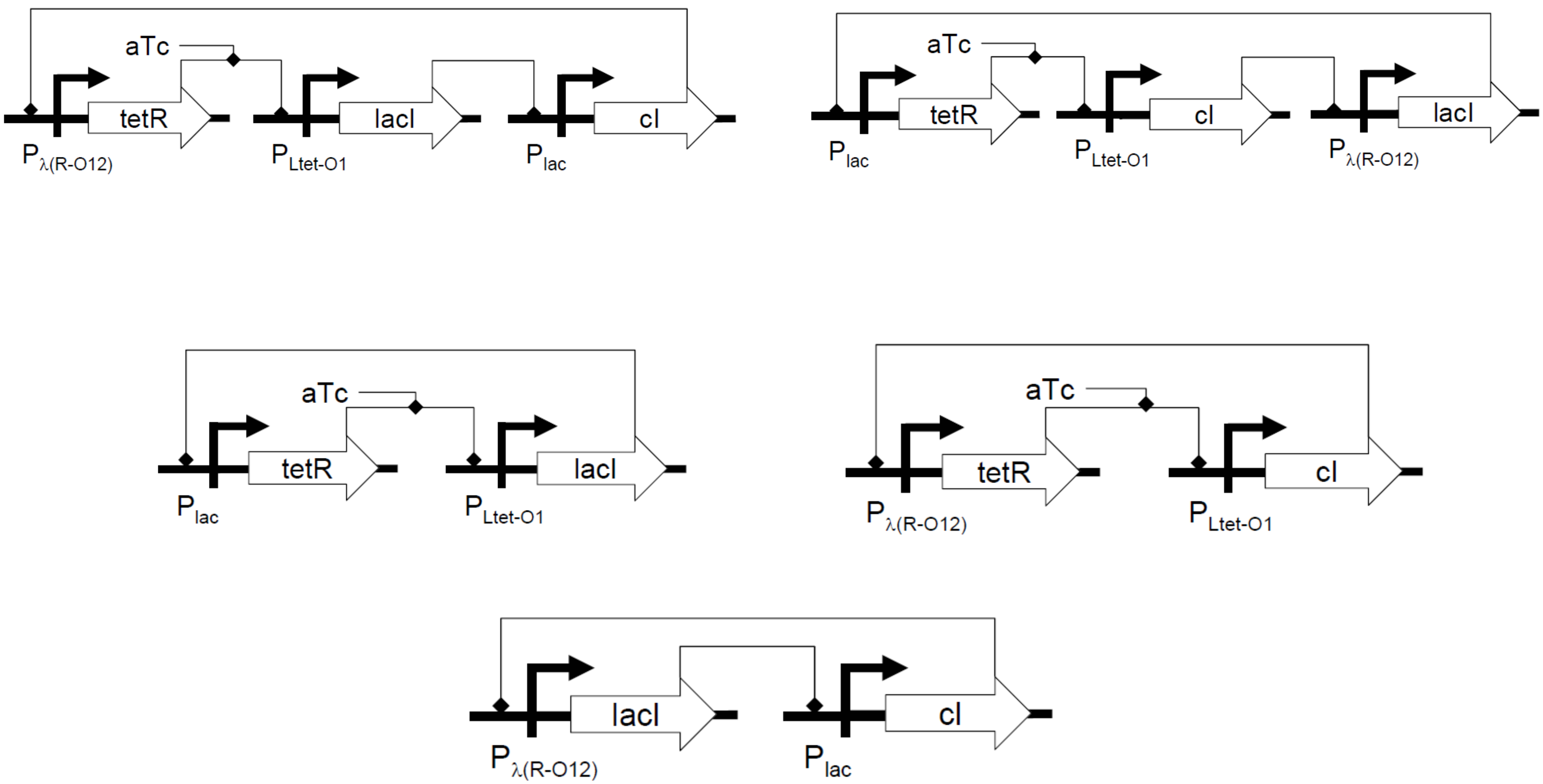
# Verification of PWA Systems

## Example: selection of devices built from parts
## Parts list



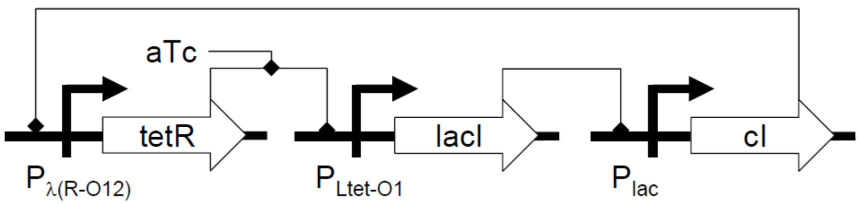
# Verification of PWA Systems

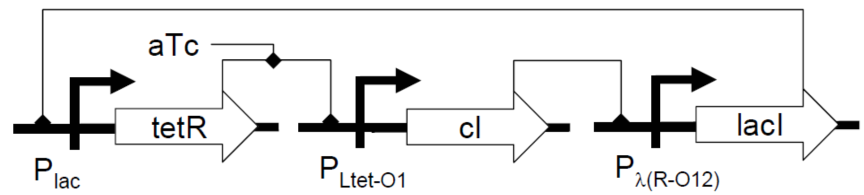
**Example: selection of devices built from parts**
**Biologically feasible devices**



# Verification of PWA Systems

**Example: selection of devices built from parts**
**Selection of possible repressilators**

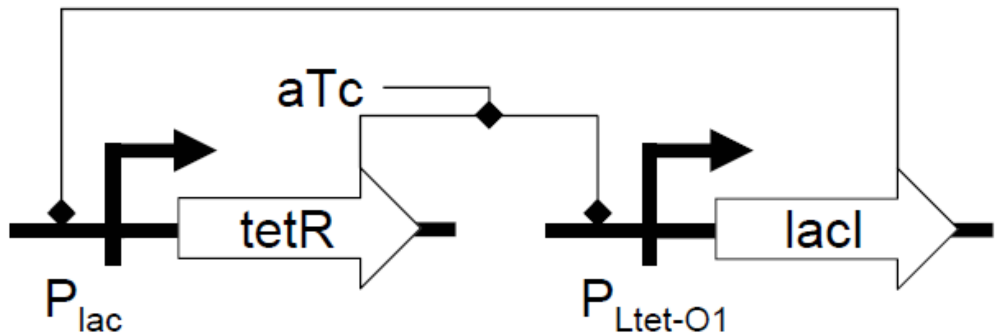



| | $\Box(\Diamond(cI<1000) \wedge \Diamond(cI>20000))$ | | | $\Box(\Diamond(lacI<1000) \wedge \Diamond(lacI>250000))$ | | |
|---|---|---|---|---|---|---|
| | Satisfying | Violating | Time | Satisfying | Violating | Time |
| Without aTc | 0% | 100% | 2.5 sec | 0% | 99.96% | 1.5 sec |
| With aTc | 0% | 99.8% | 1.5 sec | 0% | 99.96% | 1.5 sec |

# Verification of PWA Systems

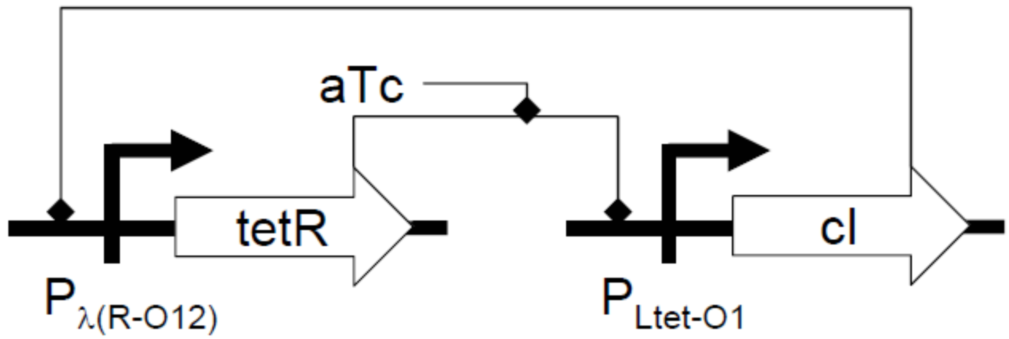
**Example: selection of devices built from parts**
**Selection of possible toggle switches**




| | $\Diamond\Box((lacI>60000) \wedge (tetR<250))$ | | |
|---|---|---|---|
| | Satisfying | Violating | Time |
| Without aTc | 0% | 100% | 1.5 sec |
| With aTc | 0% | 100% | 1.0 sec |

# Verification of PWA Systems

**Example: selection of devices built from parts**
**Selection of possible toggle switches**




| | $\Diamond\Box$((cI>60000) $\wedge$ (tetR<500)) | | | $\Diamond\Box$((cI<250) $\wedge$ (tetR>300000)) | | |
|---|---|---|---|---|---|---|
| | Satisfying | Violating | Time | Satisfying | Violating | Time |
| Without aTc | 0% | 100% | 1.0 sec | 100% | 0% | 4 sec |
| With aTc | 99.9% | 0% | 1.0 sec | 0% | 99.9% | 1.0 sec |

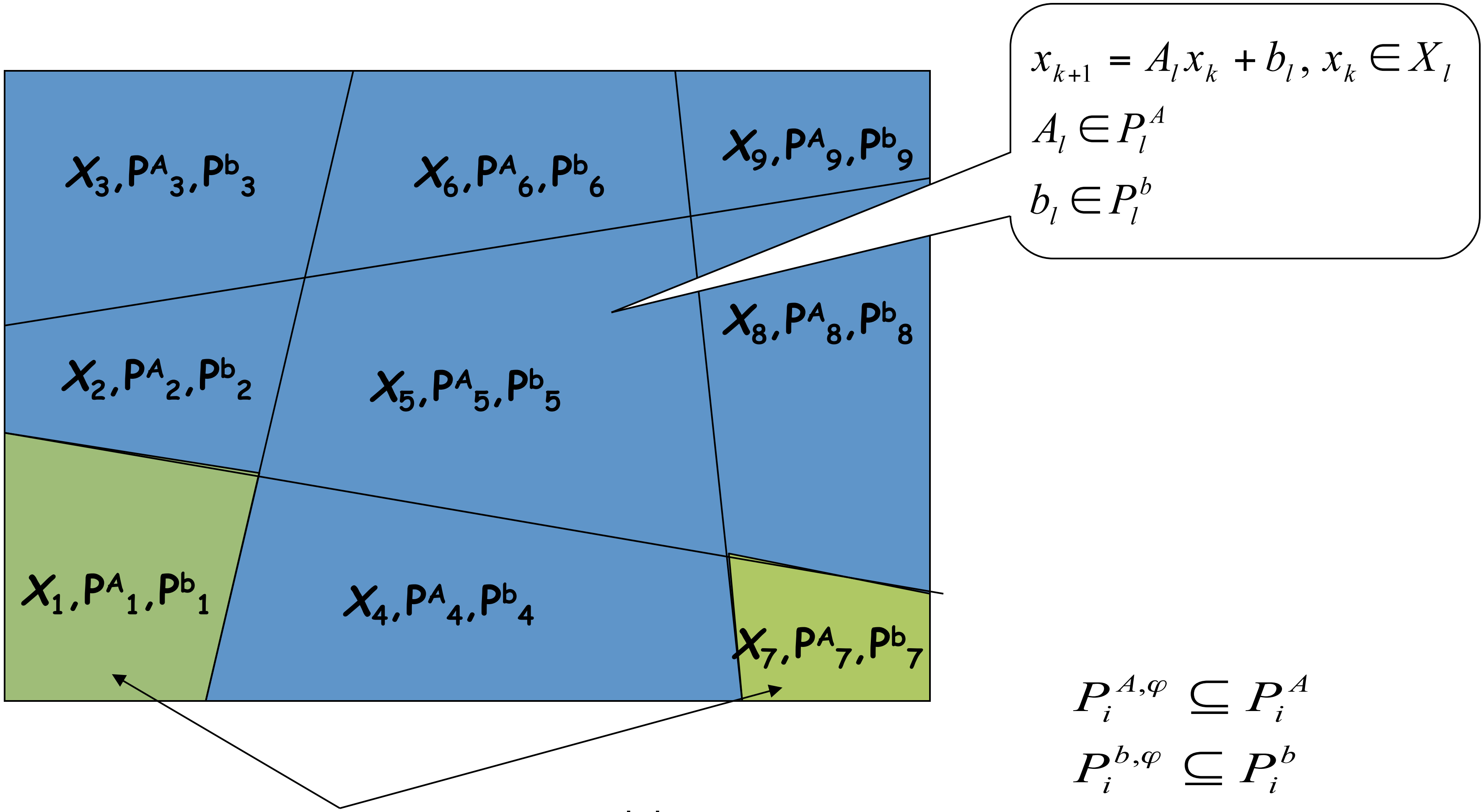
# Outline

1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

# Parameter Synthesis for PWA Systems

## Problem formulation

Given a LTL formula φ over linear predicates in the state, find a subset of the parameter sets, such that all trajectories of the system satisfy the formula.




**Initial Set** $X_0 = X_1 \cup X_7$

$$P_i^{A,\varphi} \subseteq P_i^A$$

$$P_i^{b,\varphi} \subseteq P_i^b$$

# Parameter Synthesis for PWA Systems

**Approach**

- Embed PWA system into $T_e$
- Construct an over-approximation $\overline{T}_e/_\sim$ of $T_e/_\sim$
- While there exist violating runs in $\overline{T}_e/_\sim$
    - Trim $\overline{T}_e/_\sim$ to remove a transition of a violating run
    - Limit the parameter values in the PWA to ensure the removal of the transition
- End While
- Result: $\overline{T}_e^\phi/_\sim$

The language of the obtained PWA is included in the language of $\overline{T}_e^\phi/_\sim$

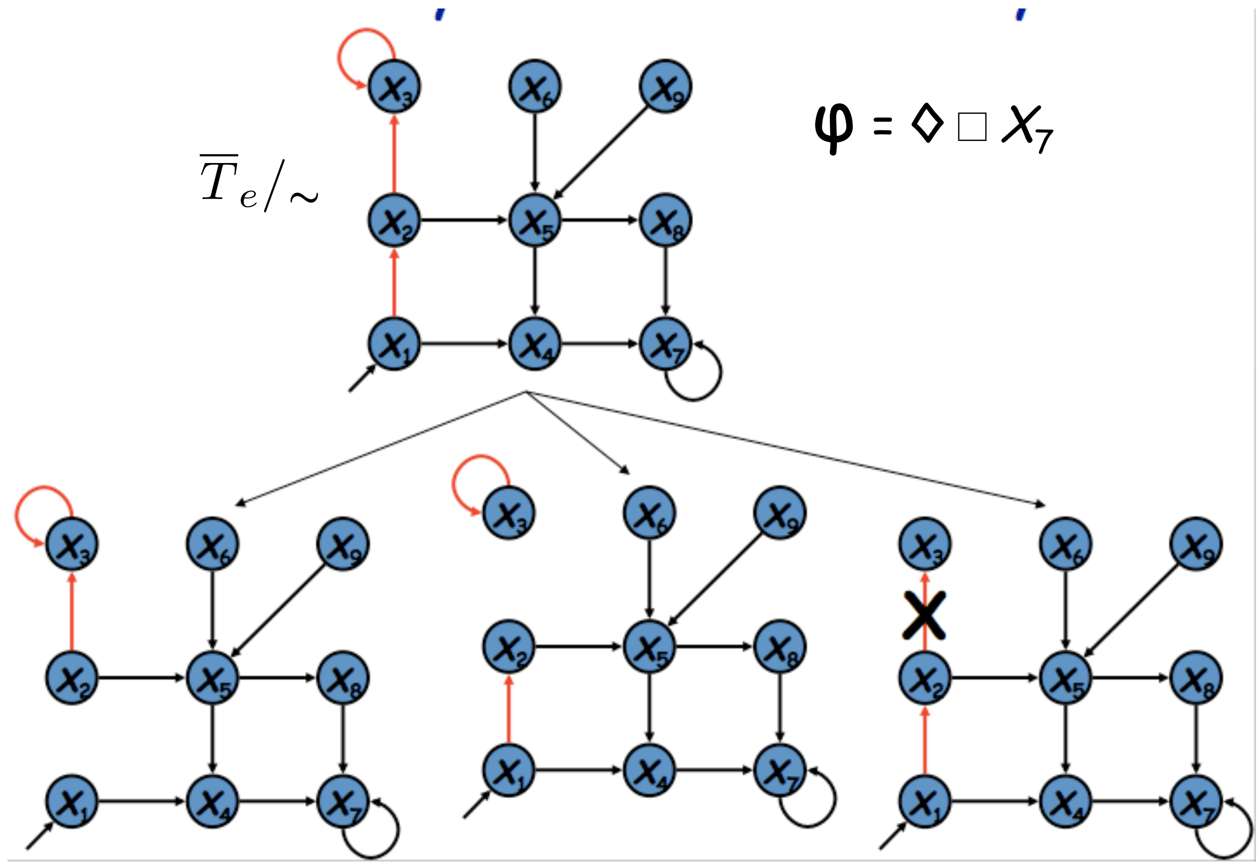

E. Clarke, A. Fehnker, Z. Han, B. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald, "Abstraction and counterexample-guided refinement in model checking of hybrid systems," International Journal of Foundations of Computer Science, vol. 14, no. 4, pp. 583–604, 2003.

*Frehse, Jha, Krogh. A Counterexample-Guided Approach to Parameter Synthesis for Linear Hybrid Automata. In HSCC 2008*

Yordanov, B. and Belta, C., HSCC '08

# Parameter Synthesis for PWA Systems

## Counterexample - guided transition elimination



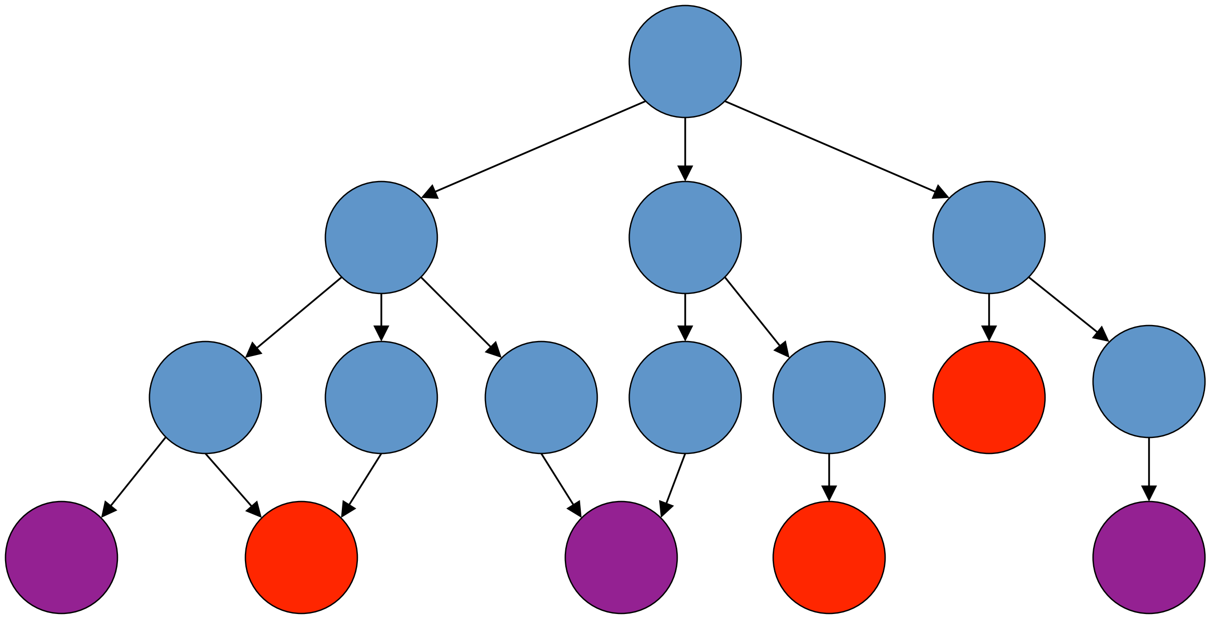

When a transition is removed, the set of parameters of the PWA system is restricted

1) Other transitions might be disabled as a side effect

2) Some states might become blocking - the transitions to these states need to removed as well by further restricting the parameters of the PWA system

# Parameter Synthesis for PWA Systems

## Satisfying quotients tree



Non-satisfying finite quotients that generate further counterexamples

Finite quotients with blocking initial states
(no more counterexamples can be generated but the formula is not satisfied)

Satisfying finite quotients without any reachable blocking states

# Parameter Synthesis for PWA Systems

**Parameter sets disabling transitions in $\overline{T}_e/_\sim$**

Let $P^{X_i \not\to X_j}$ denote the set of all parameters for which $Post(X_i) \cap X_j = \emptyset$

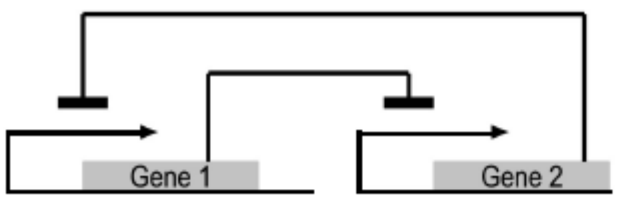
Removing a transition means restricting the parameters to $P^{X_i \not\to X_j}$

$P^{X_i \not\to X_j}$ is not computable

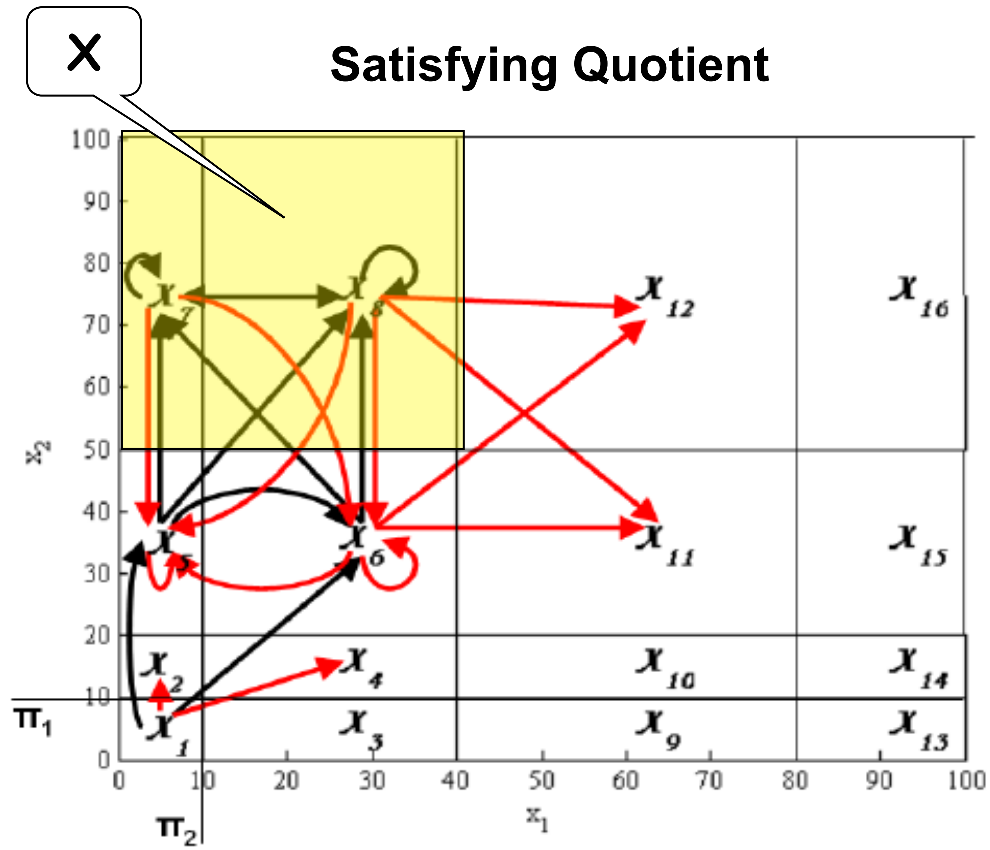
An under-approximation $\underline{P^{X_i \not\to X_j}}$ can be computed

# Parameter Synthesis for PWA Systems

**Example**

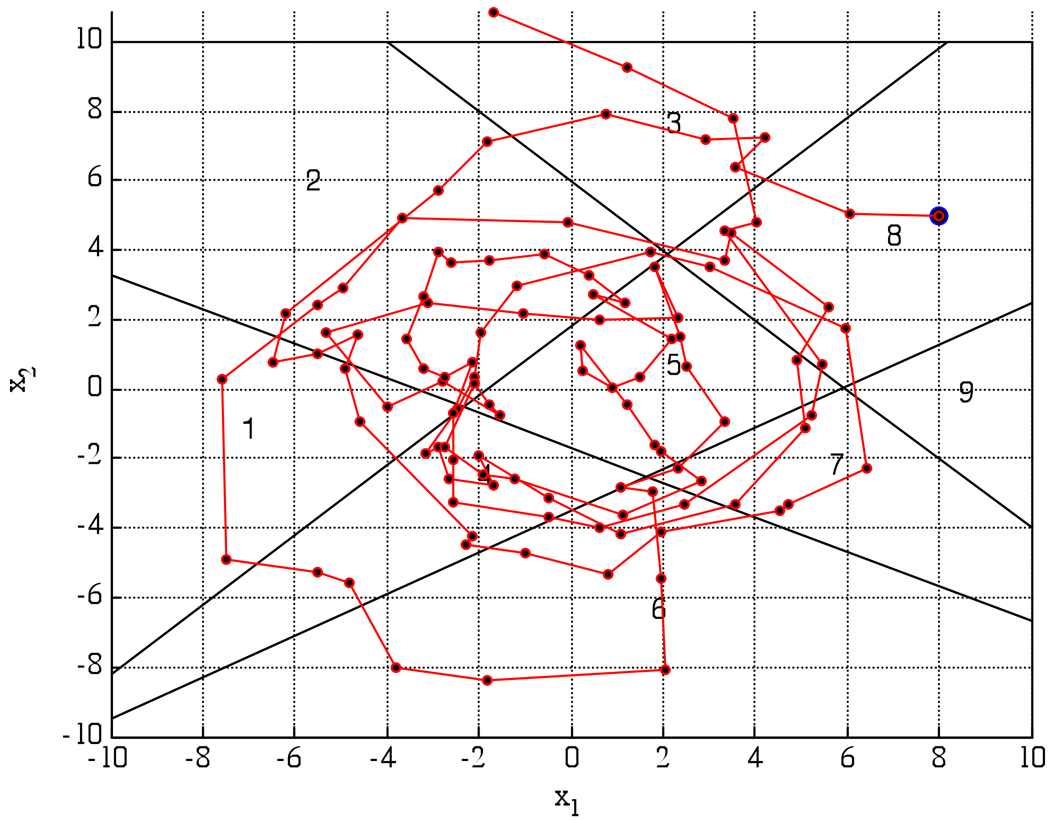



$\varphi = \Diamond \Box X$



# Parameter Synthesis for PWA Systems

**Example**




$$x_{k+1} = Ax_k + c, A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}, c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix},$$

$a_1 \in [0.8,1], a_2 \in [-0.55,-0.05],$

$a_3 \in [0.05,0.55], a_4 \in [0.8,1],$

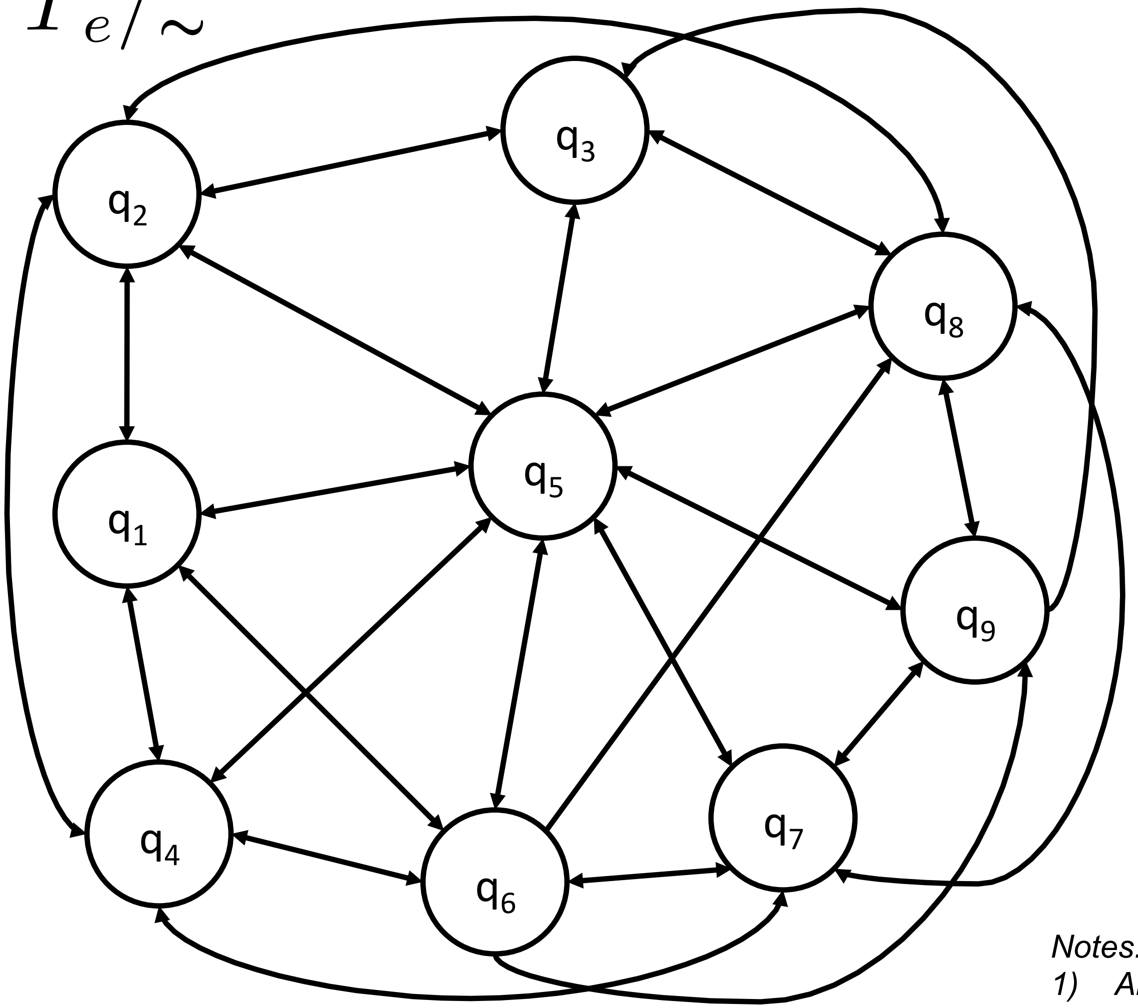
$c_1 \in [-1,1], c_2 \in [-1,1],$

**Specification**: "Keep surveying all regions except 5, which should never be visited", i.e., "always (eventually 1 and eventually 2 ... and eventually 4 and eventually 6 ... and eventually 9) and always not 5. Do not go out of the [-10,10] x [-10 10] rectangle."

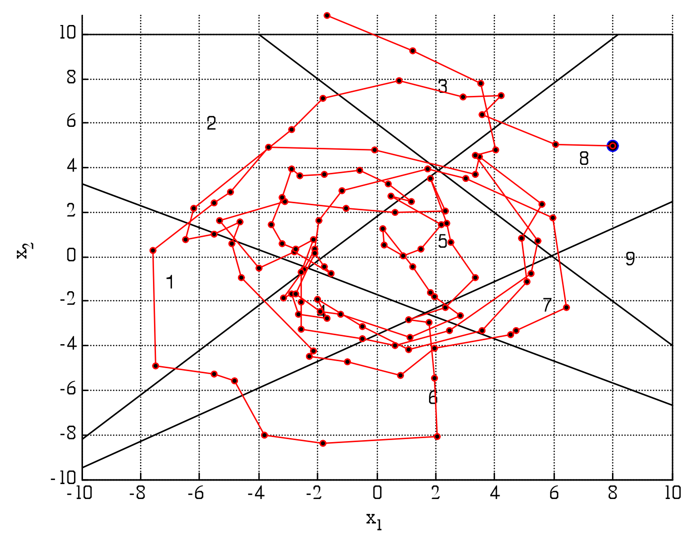
# Parameter Synthesis for PWA Systems

## Example

$\overline{T}_e/\sim$




*63 Transitions total*



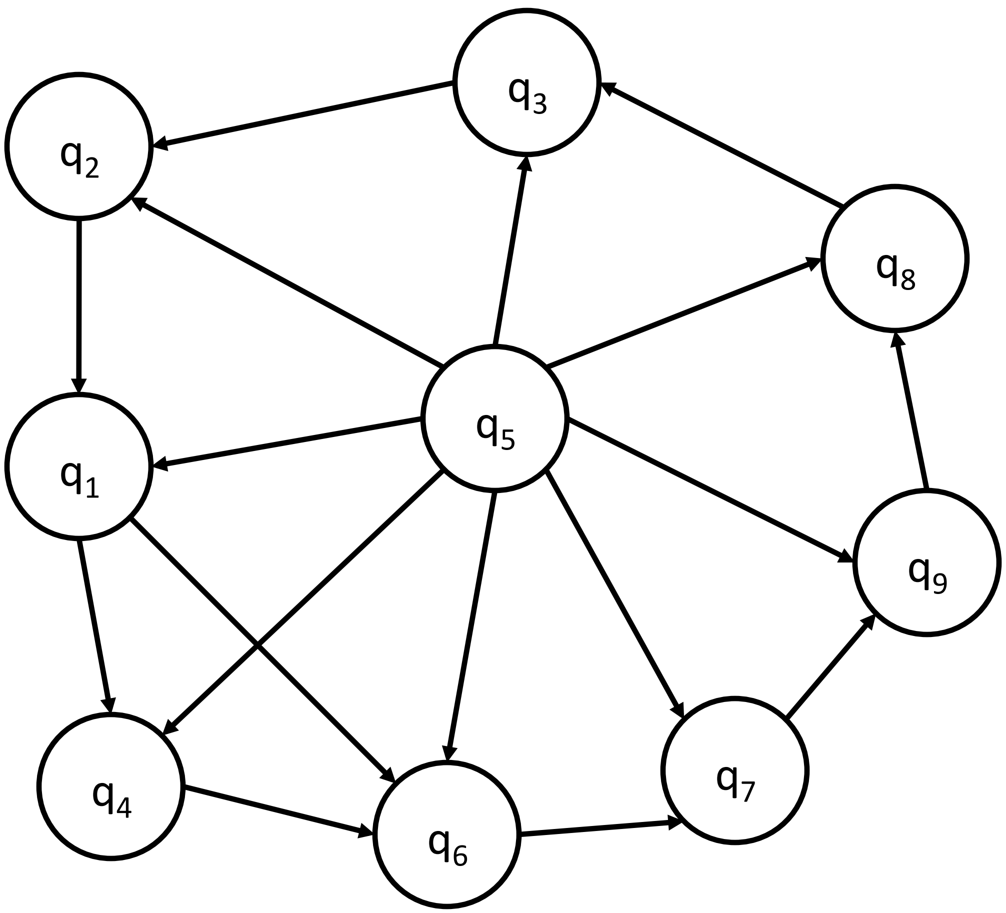

*Notes:*

1) *All states have self loops (omitted)*
2) *State 1,2,3,4,6,7,8,9 have transitions to Out (omitted)*

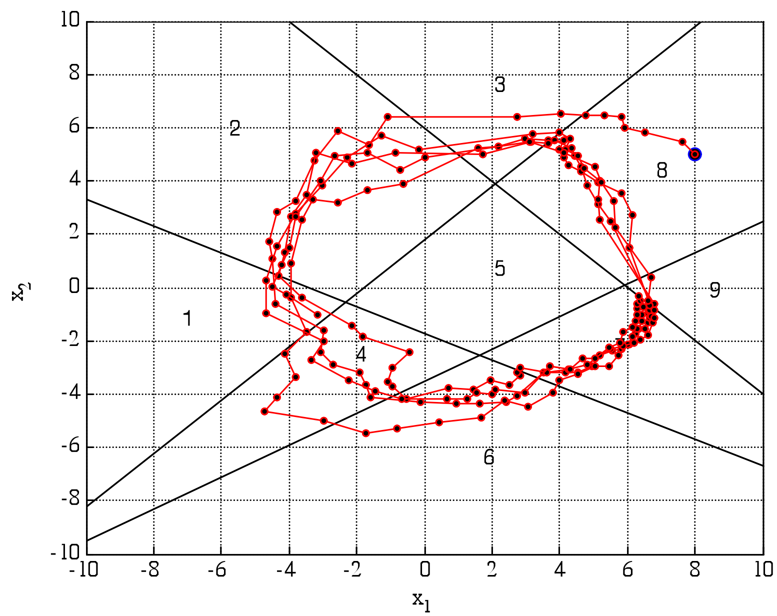
# Parameter Synthesis for PWA Systems

**Example**

**Trimmed** $\overline{T}_e/_\sim$




*28 Transitions total*

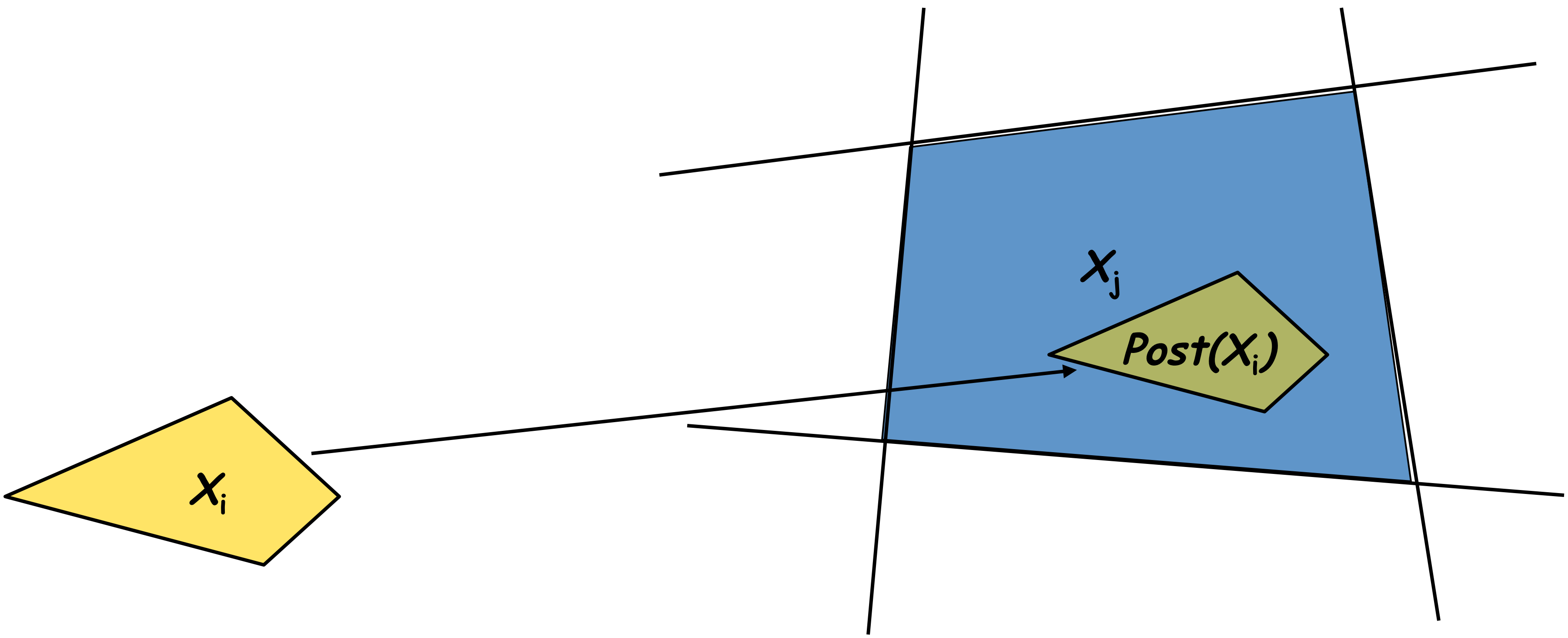


# Parameter Synthesis for PWA Systems

**Sets of parameters producing a bisimulation quotient**

Let $P^{X_i \to X_j}$ denote the set of all parameters for which $Post(X_i) \subseteq X_j$

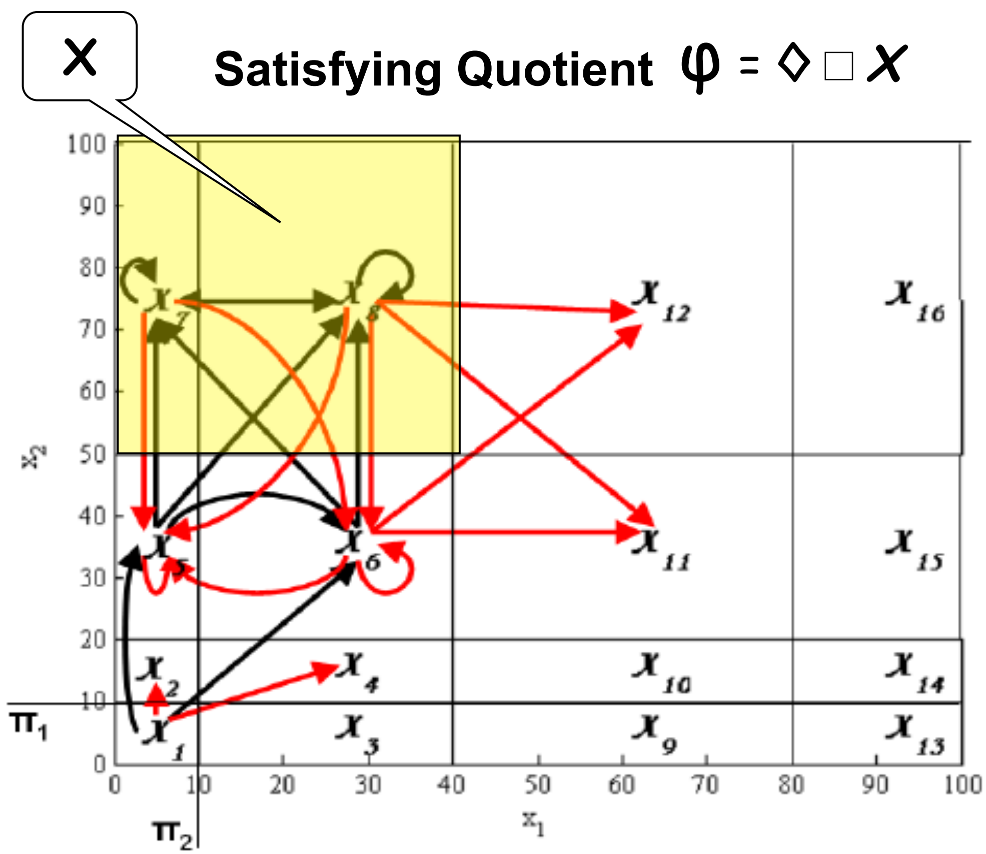
$P^{X_i \to X_j}$ is computable

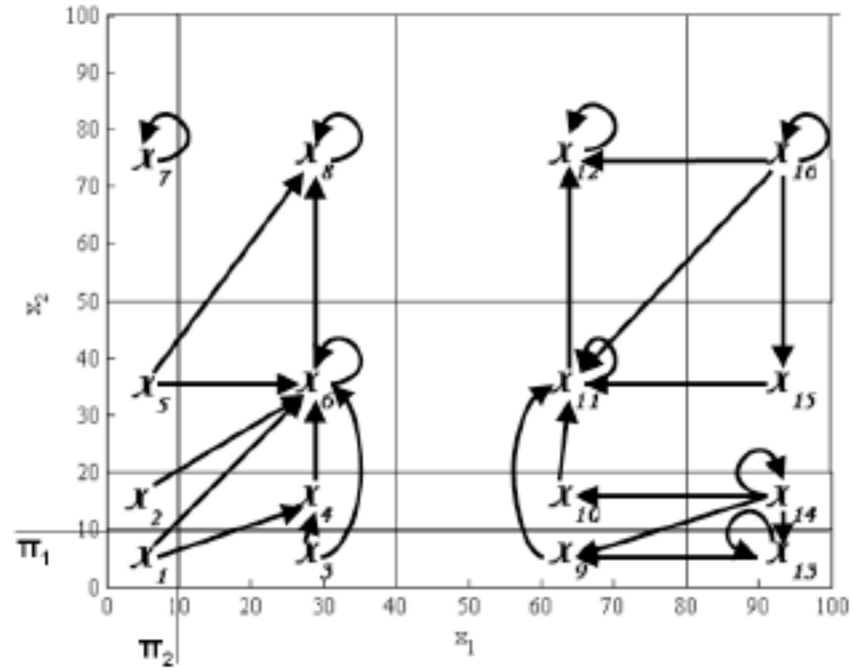


# Parameter Synthesis for PWA Systems

## Parameter synthesis


**Satisfying Quotient** $\varphi = \Diamond \Box X$




**Bisimulation Quotient**

# Outline

1) LTL verification and control for finite systems
2) PWA Systems
3) Verification of PWA Systems
4) Parameter Synthesis for PWA Systems
5) LTL Control of PWA Systems

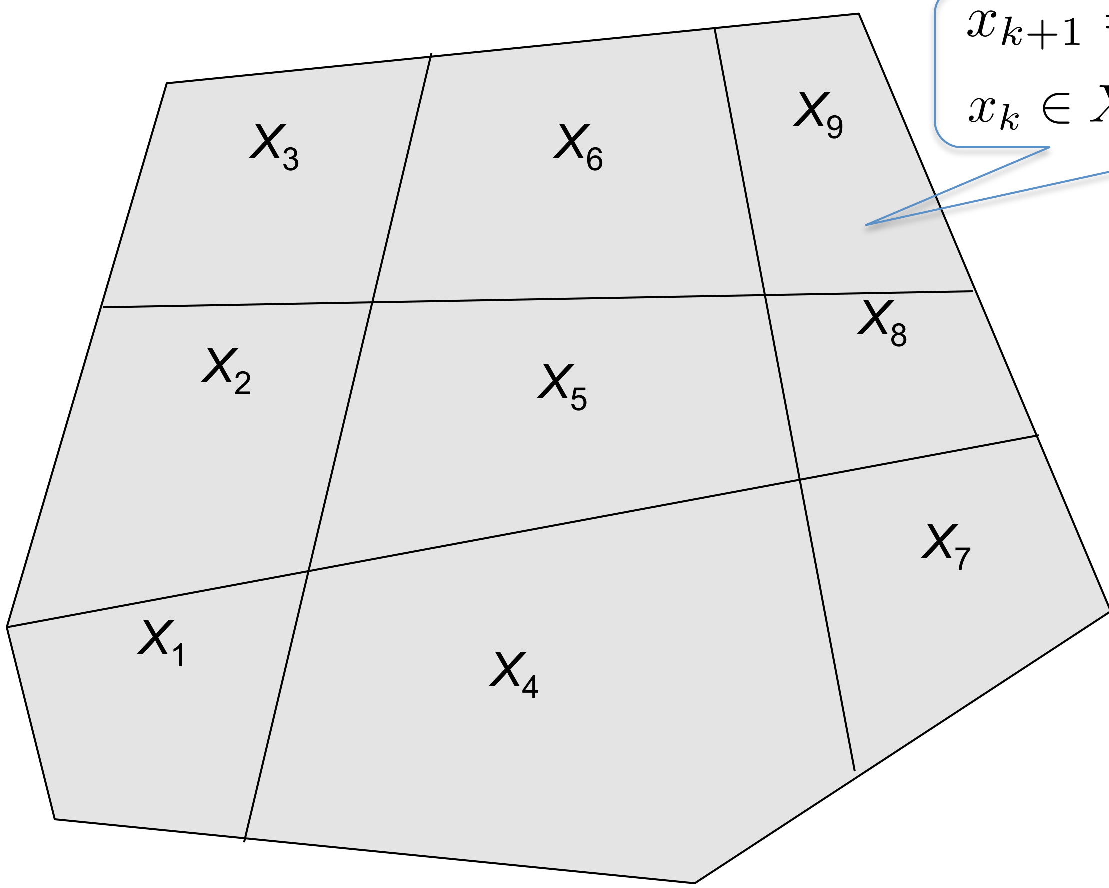
# LTL Control of PWA Systems

**Problem formulation**

Find a set of initial states and a state-feedback control strategy such that all the trajectories of the system satisfy an arbitrary LTL formula over linear predicates over the states.

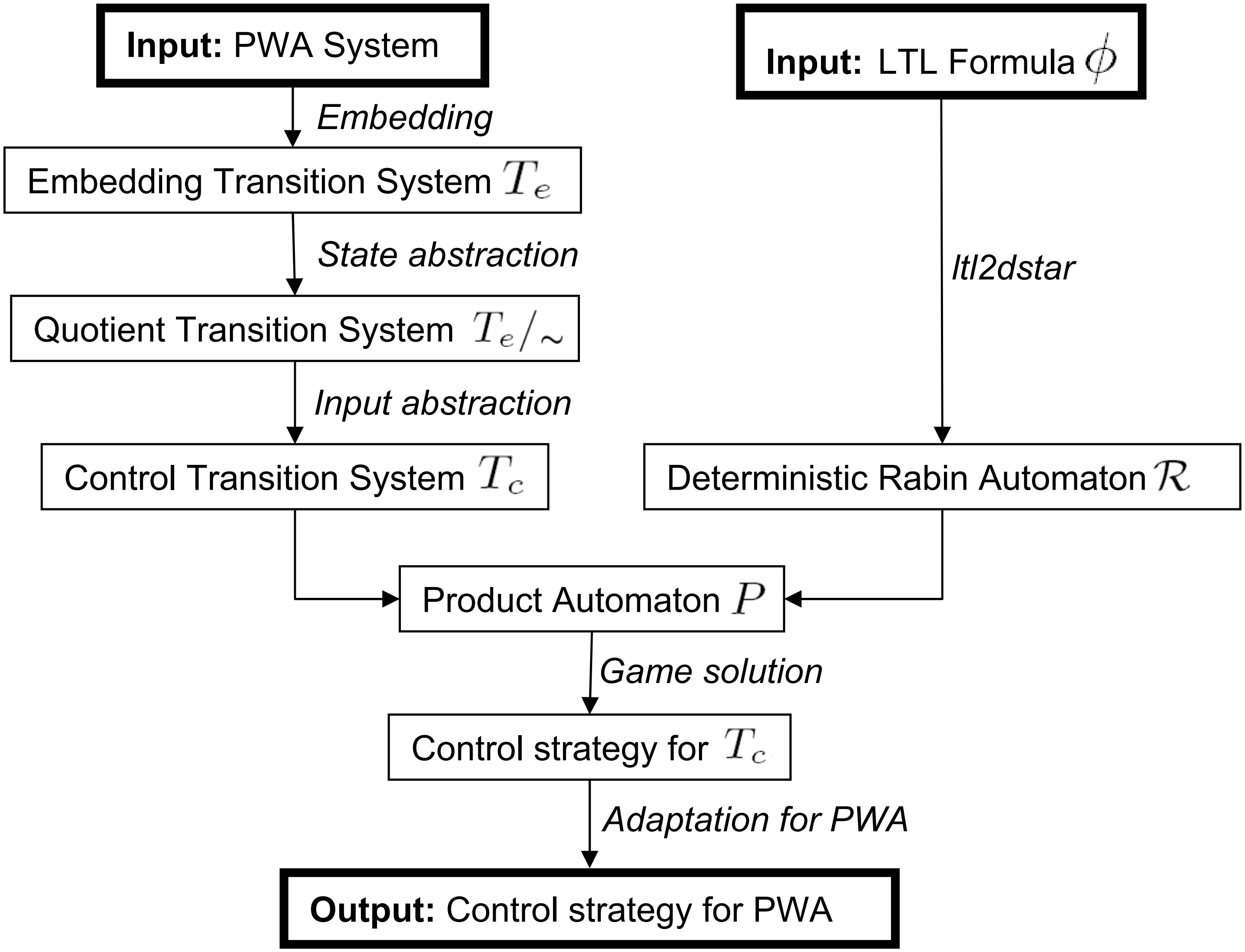
$$x_{k+1} = A_l x_k + B_l u_k + b_l$$

$$x_k \in X_l \qquad u_k \in U_l \qquad l \in L$$



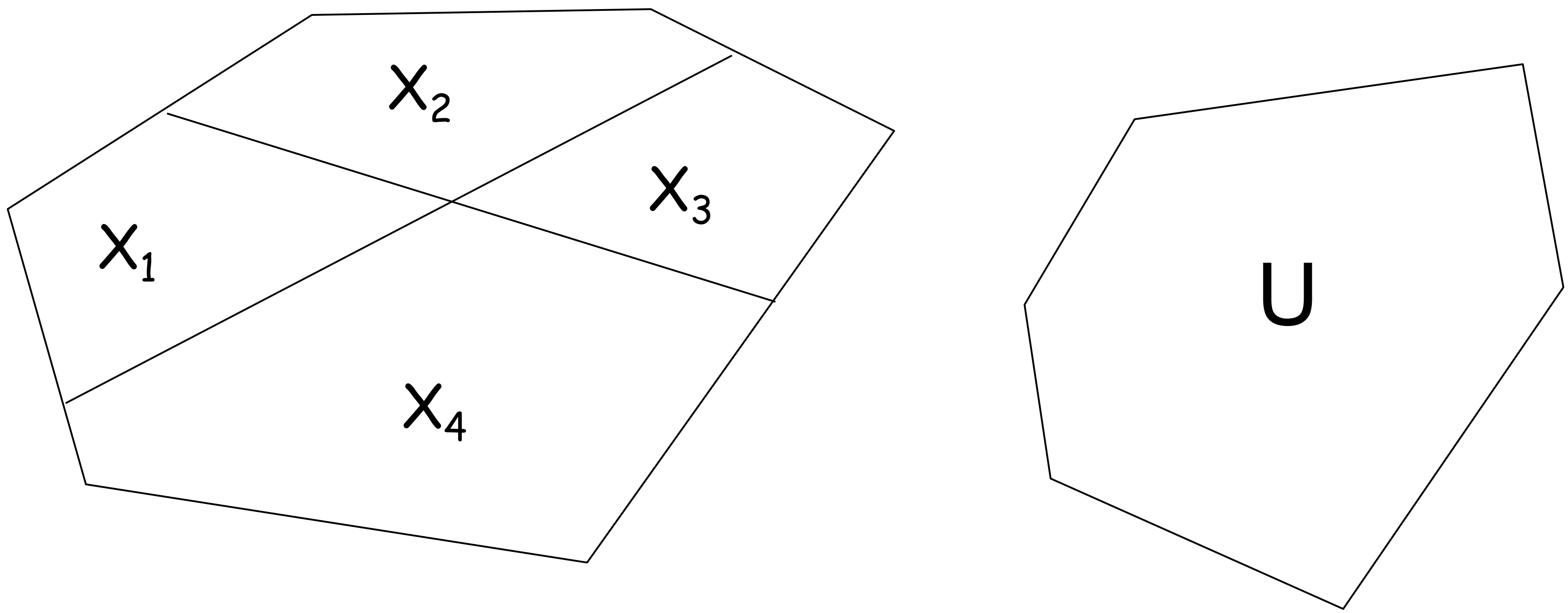
# LTL Control of PWA Systems

## Approach



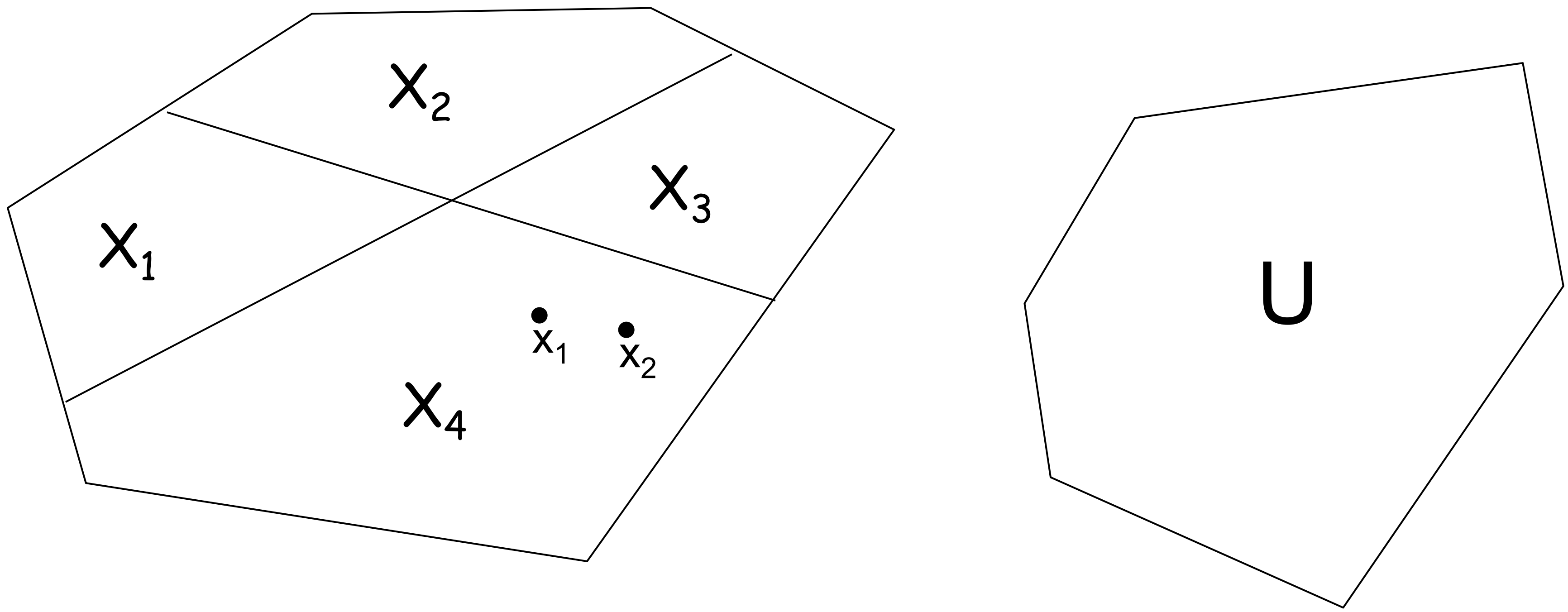
# LTL Control of PWA Systems

**State abstraction**



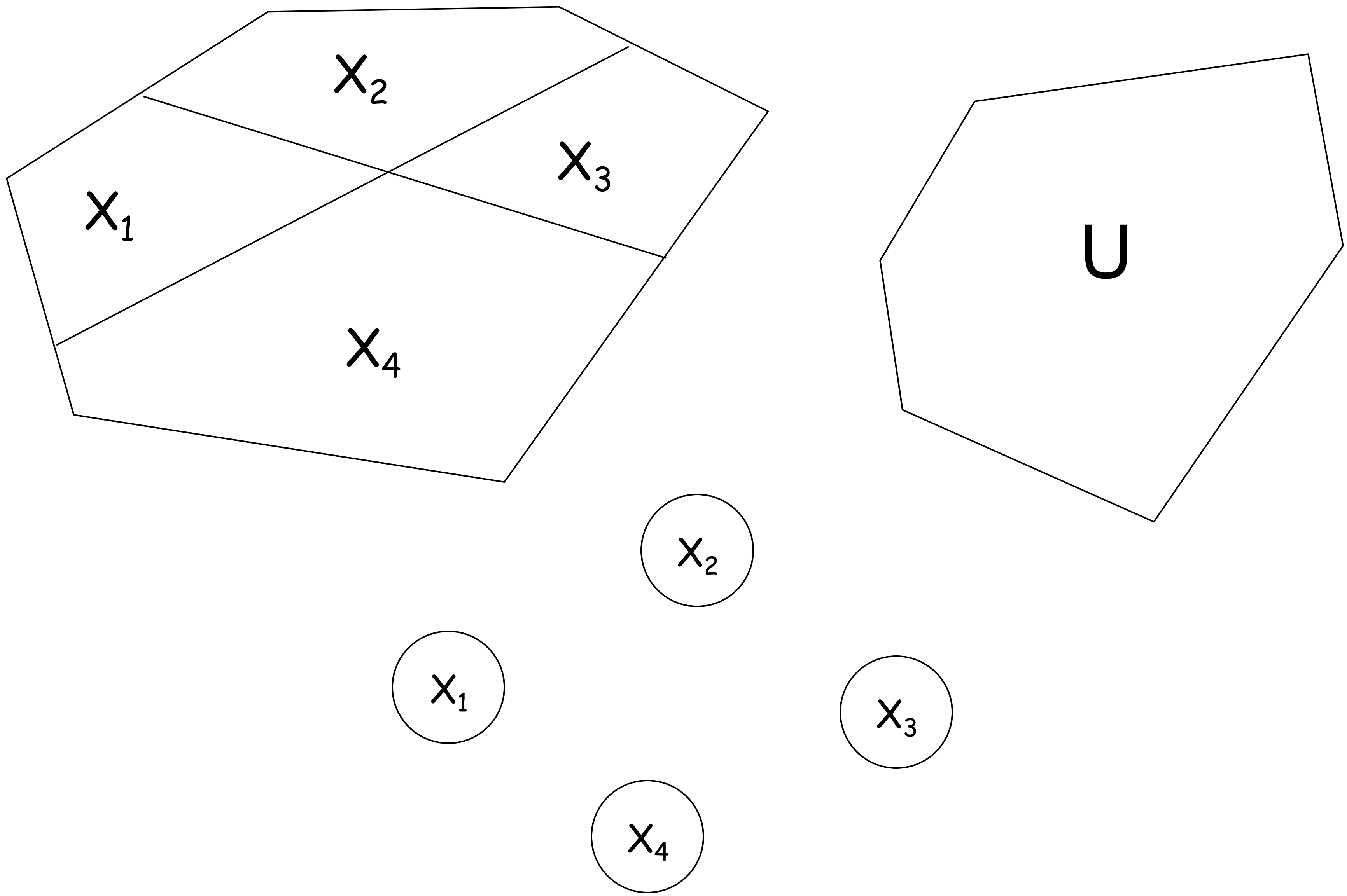
# LTL Control of PWA Systems

State abstraction



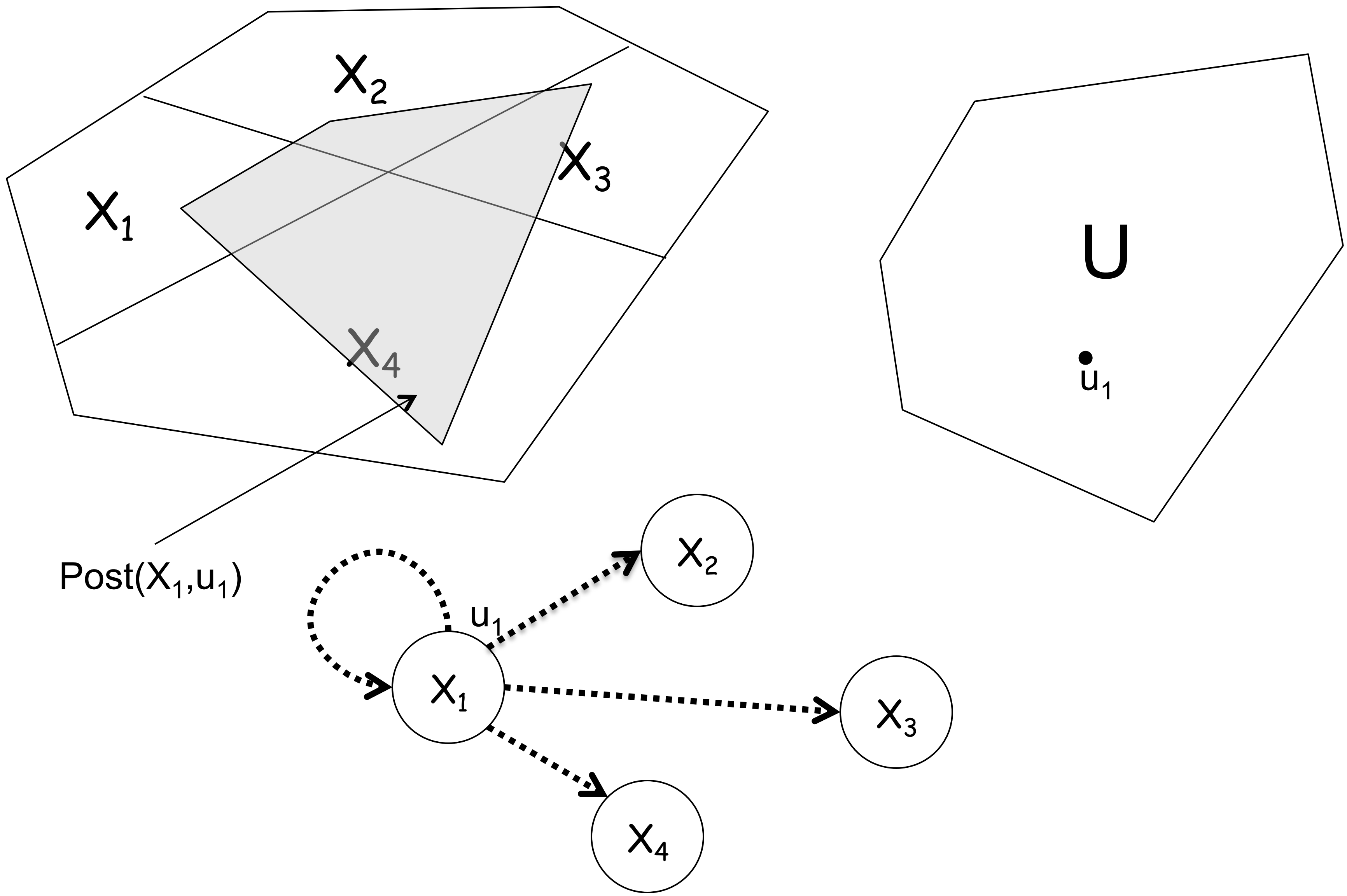
# LTL Control of PWA Systems

## State abstraction



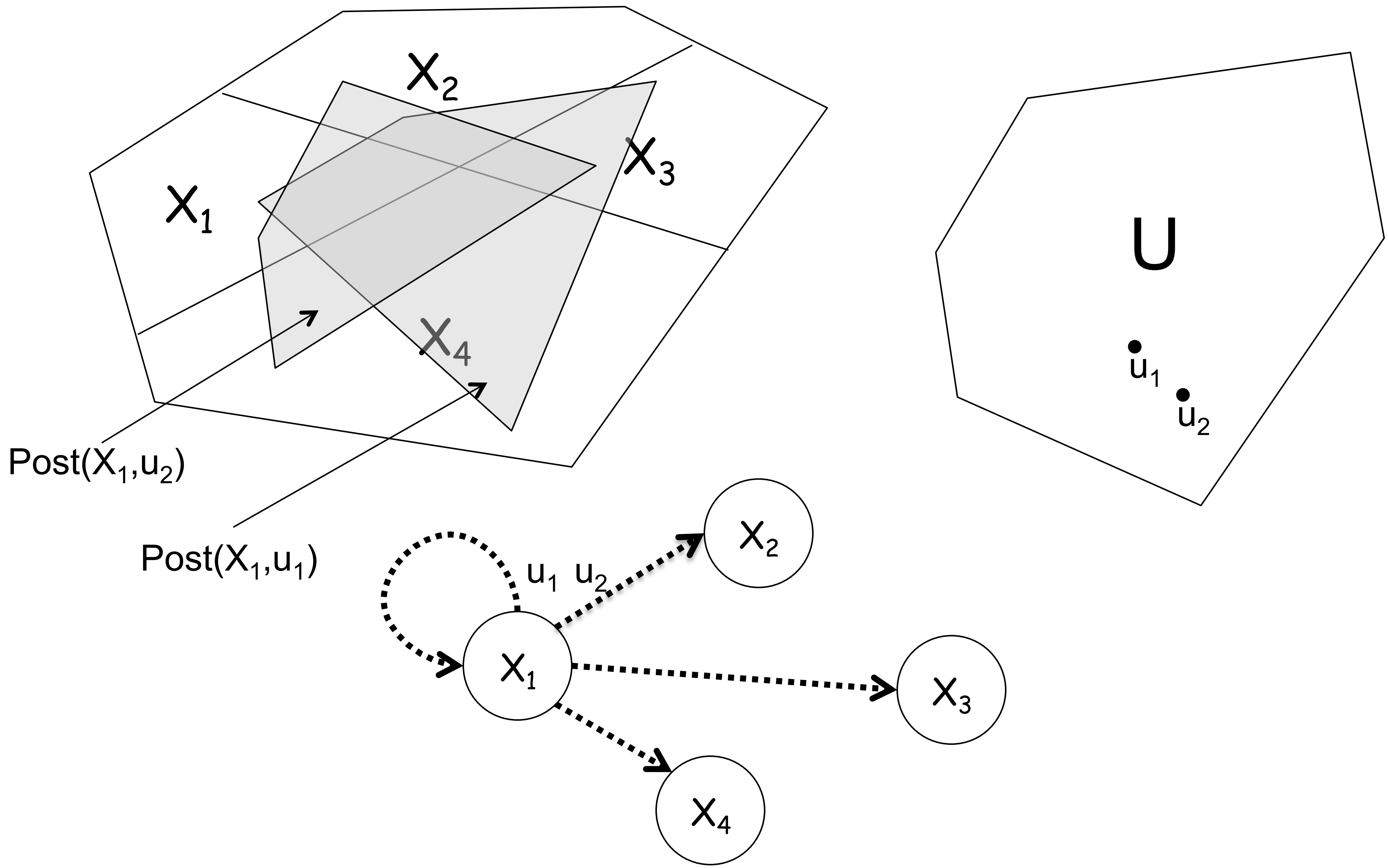
# LTL Control of PWA Systems

## Control abstraction



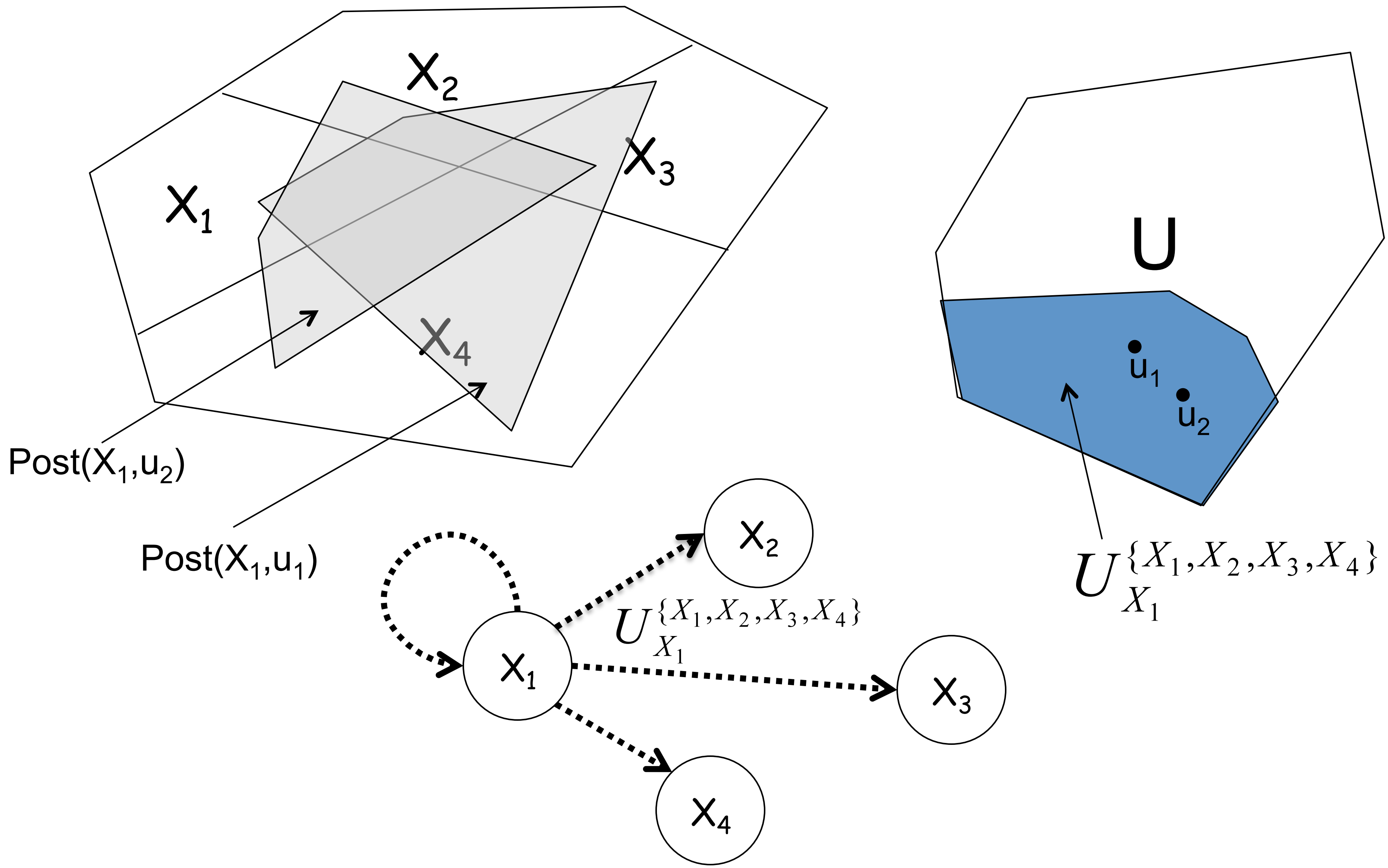
# LTL Control of PWA Systems

## Control abstraction



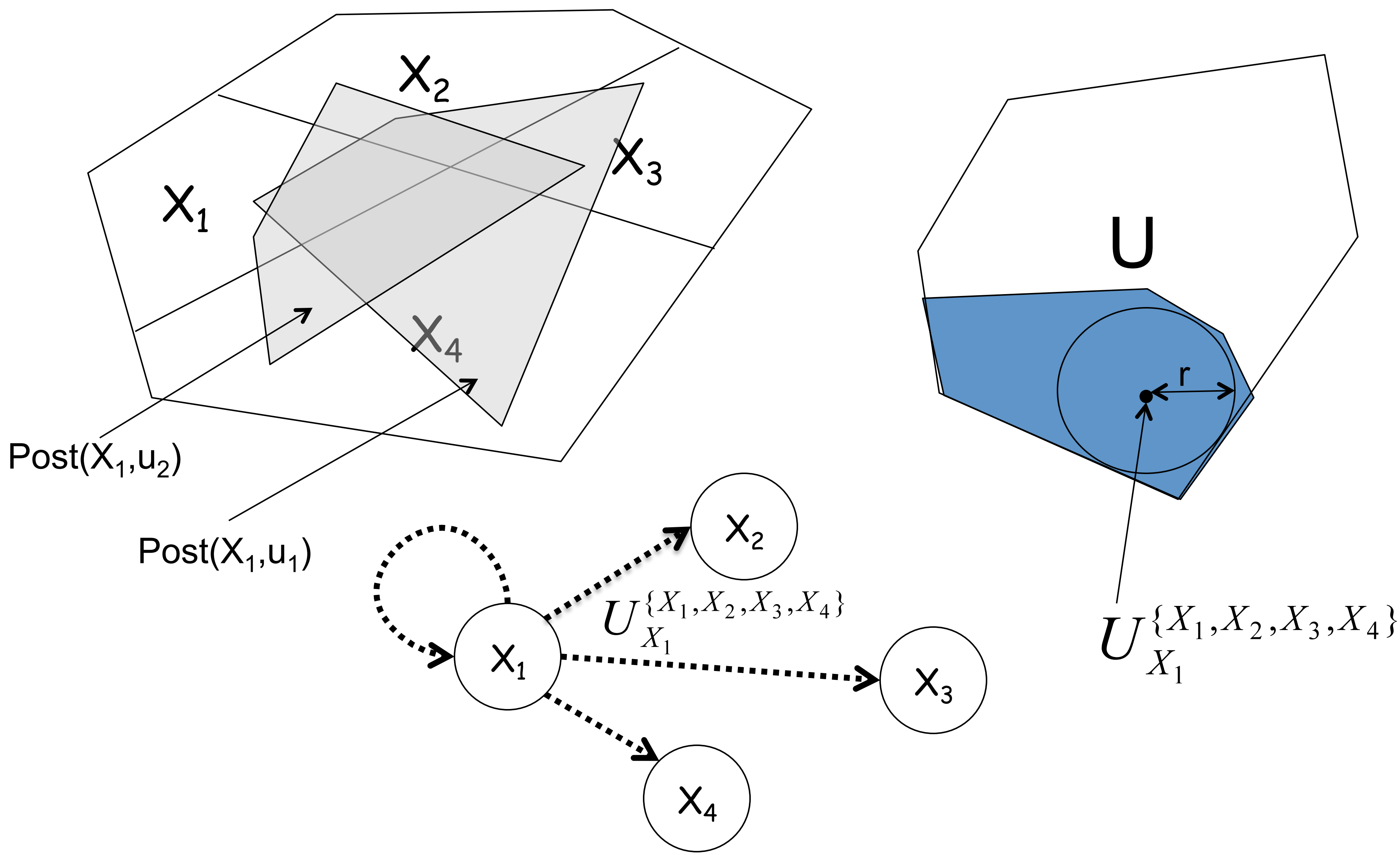
# LTL Control of PWA Systems

## Control abstraction



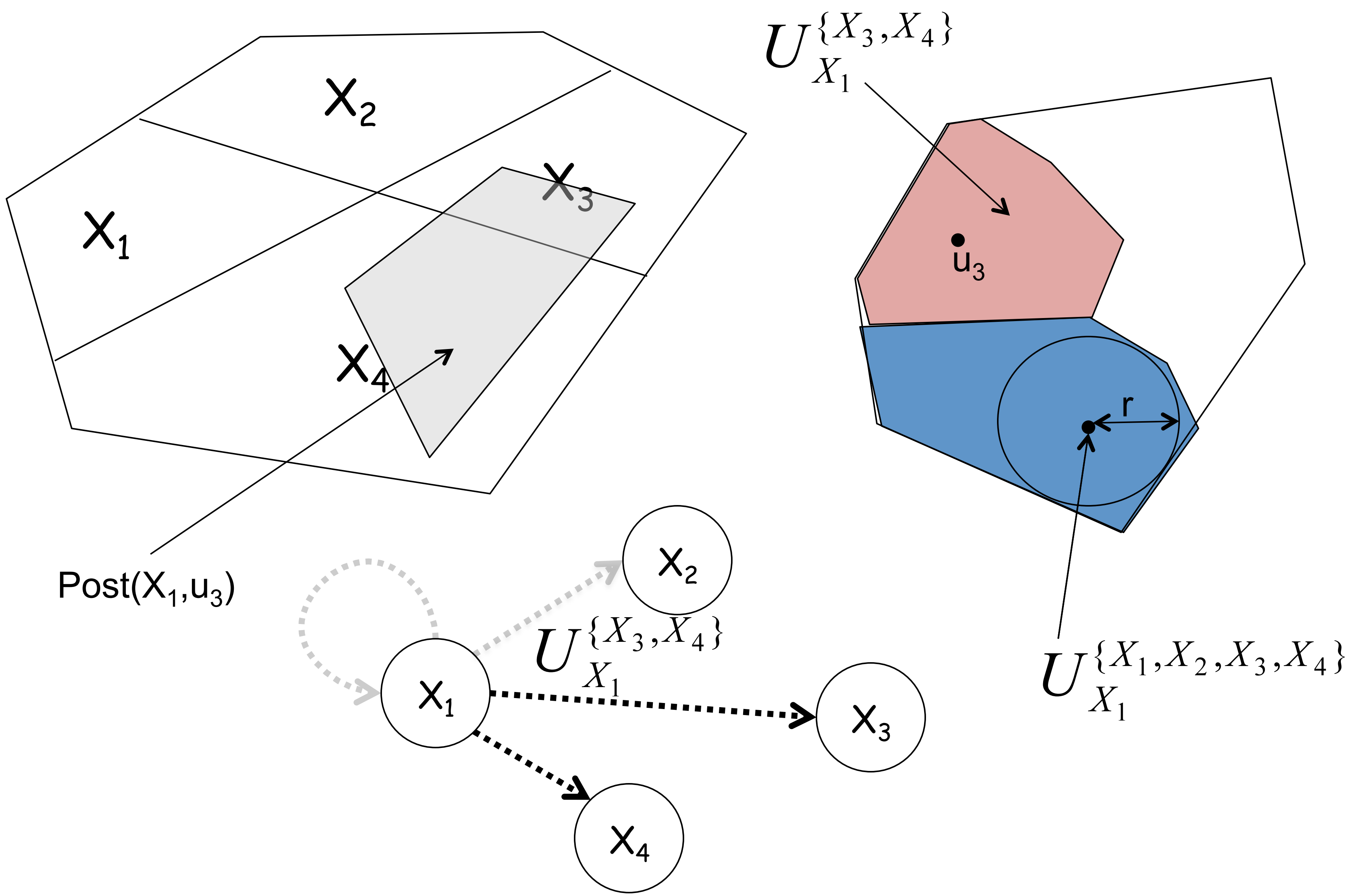
# LTL Control of PWA Systems

## Control abstraction



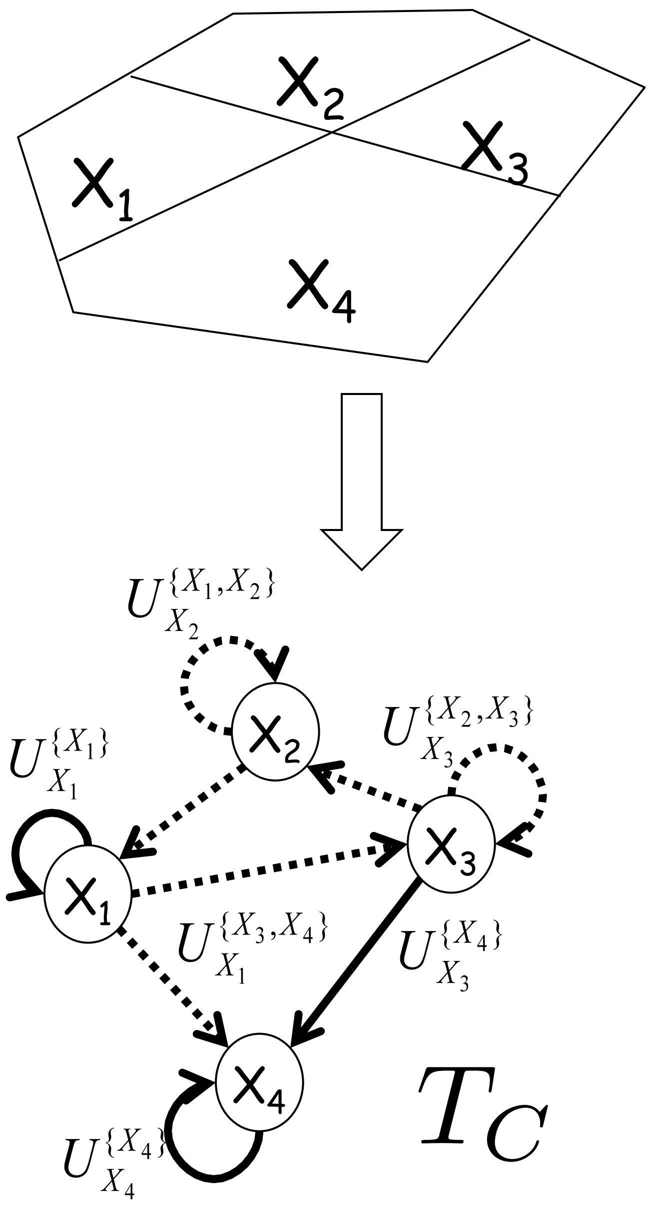
# LTL Control of PWA Systems

## Control abstraction



# LTL Control of PWA Systems

## Finite control transition system




1: Compute states (state equivalence classes)
2: For each state:
  2.1: Compute inputs (input equivalence classes)
  2.2: Remove inputs that are "too small"
  2.3: Keep only "most deterministic" inputs
3: Generate control strategy for control TS
4: Adapt the control strategy to the PWA system (language inclusion)

**The finite control transition system $T_C$ can be constructed using polyhedral operations only.**

Yordanov, B. and Belta, C., CDC '09

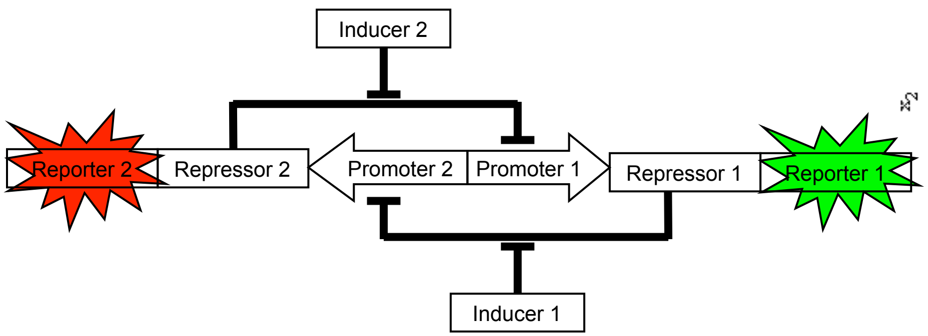
Tumova, J., Yordanov, B., Belta, C., Cerna, I., and Barnat, J., CDC '10

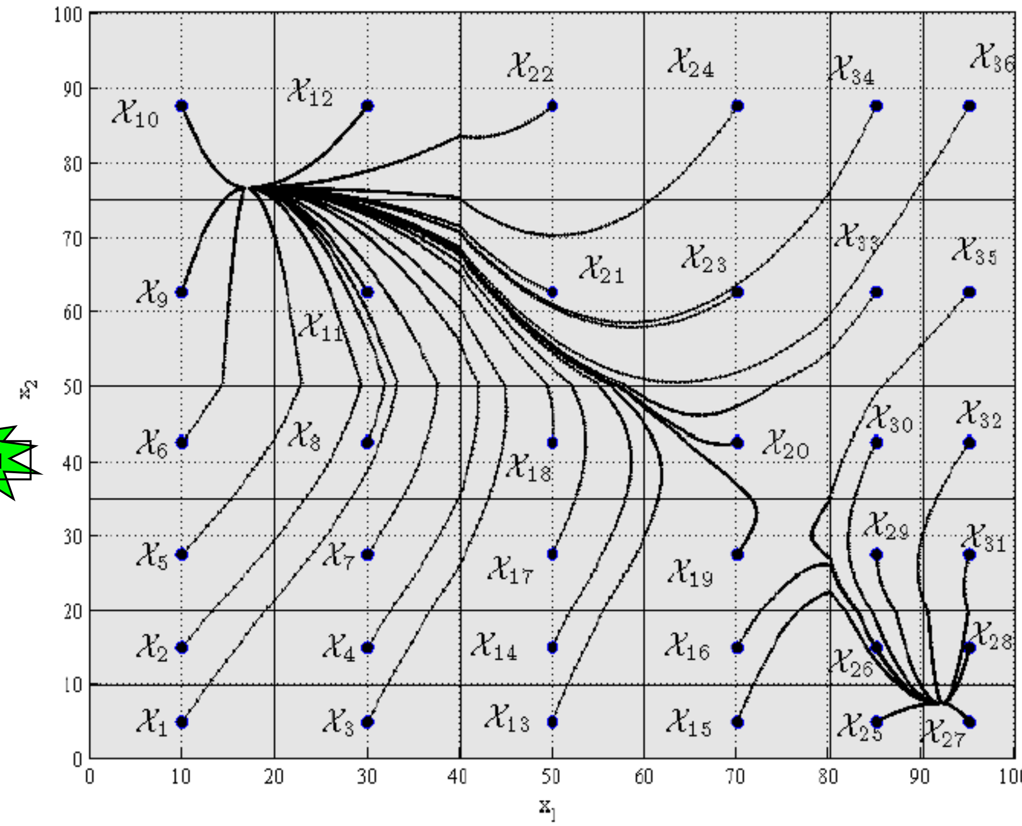
Yordanov, B. and Belta, C., Accepted in IEEE Trans. Autom. Control, 2011

# LTL Control of PWA Systems

## Example: Buchi game

$$\phi = \Diamond \mathcal{X}_1 \wedge \Diamond \mathcal{X}_{10} \wedge \Diamond \mathcal{X}_{27} \wedge \Diamond \mathcal{X}_{36}$$



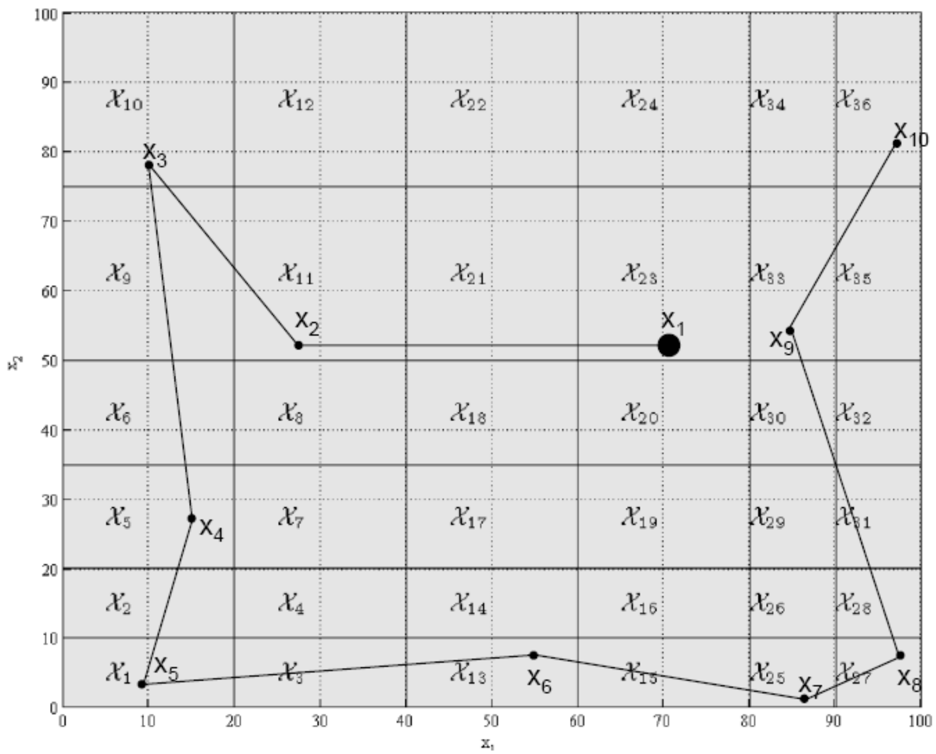



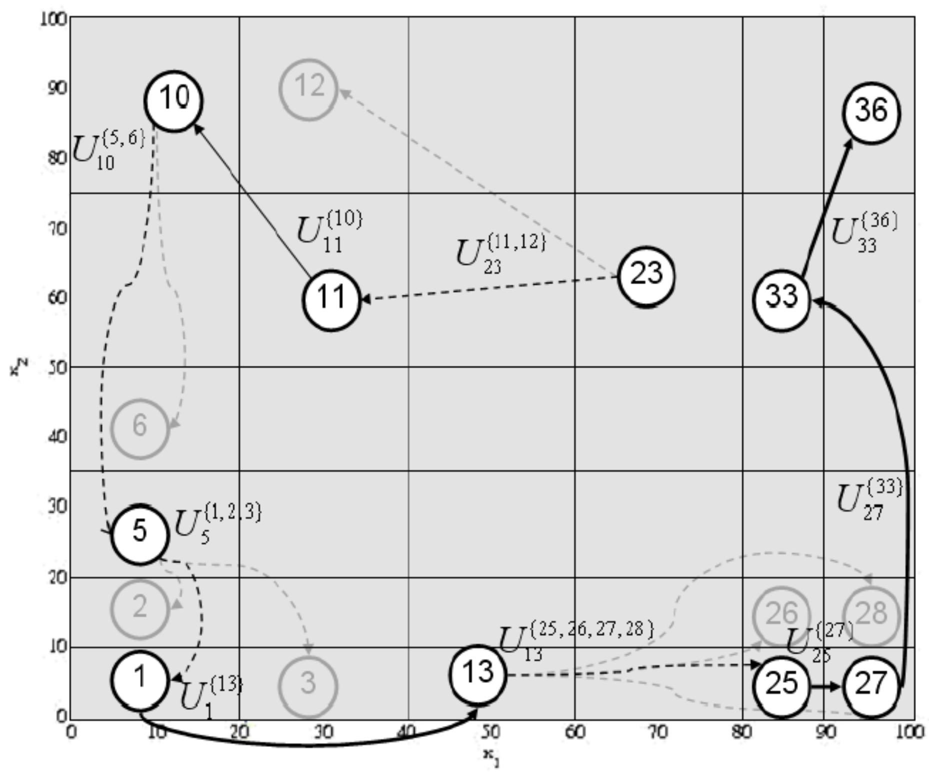

- 36 states
- 4115 nonempty input regions
- 3182 input regions were "large enough" (limit=0.05)
- 260 input regions induce deterministic transitions only
  (do not lead to a solution from any state - no solution can be found if the game is avoided!!)
- 691 "most deterministic" input regions were included
  (control strategies were found from all 36 states)

# LTL Control of PWA Systems

**Example: Buchi game**

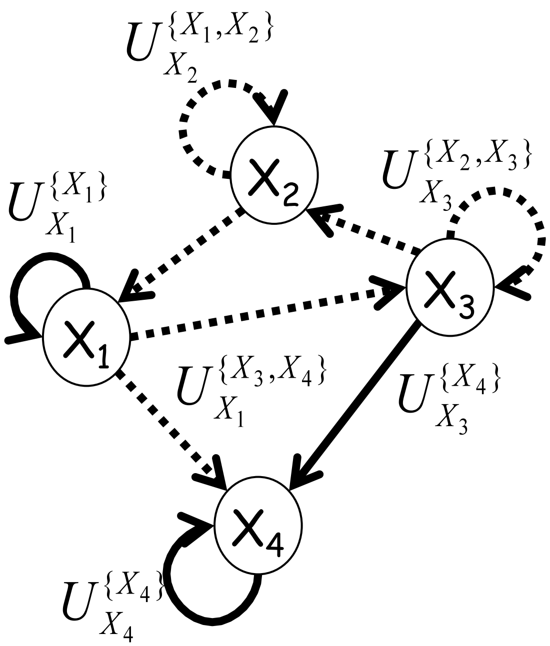
$$\phi = \Diamond \mathcal{X}_1 \wedge \Diamond \mathcal{X}_{10} \wedge \Diamond \mathcal{X}_{27} \wedge \Diamond \mathcal{X}_{36}$$





# LTL Control of PWA Systems

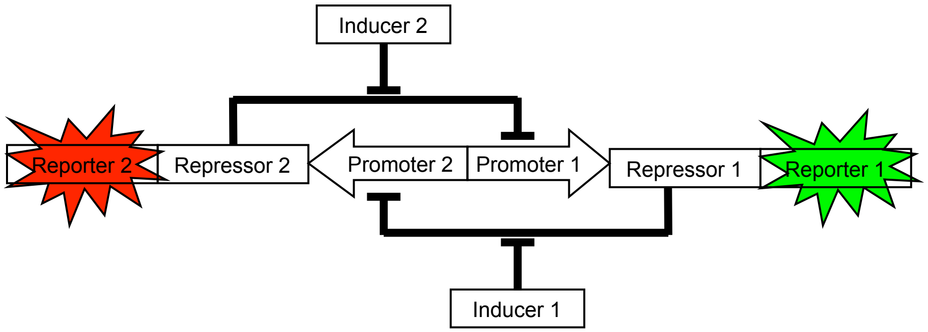
**Improving the solution: stuttering phenomena**



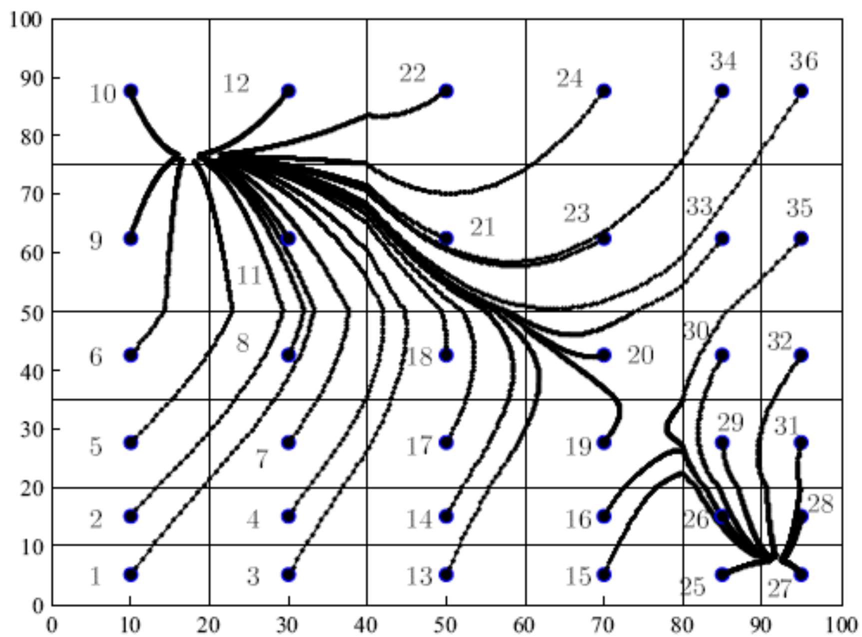

- For nondeterministic transitions in the control transition system that contain self-loops, the adversary can use the self-loop to win the game.
- We can characterize the input sets that are stuttering (guarantee to leave the region in finitely many steps)
- Stuttering inputs can be used in the game

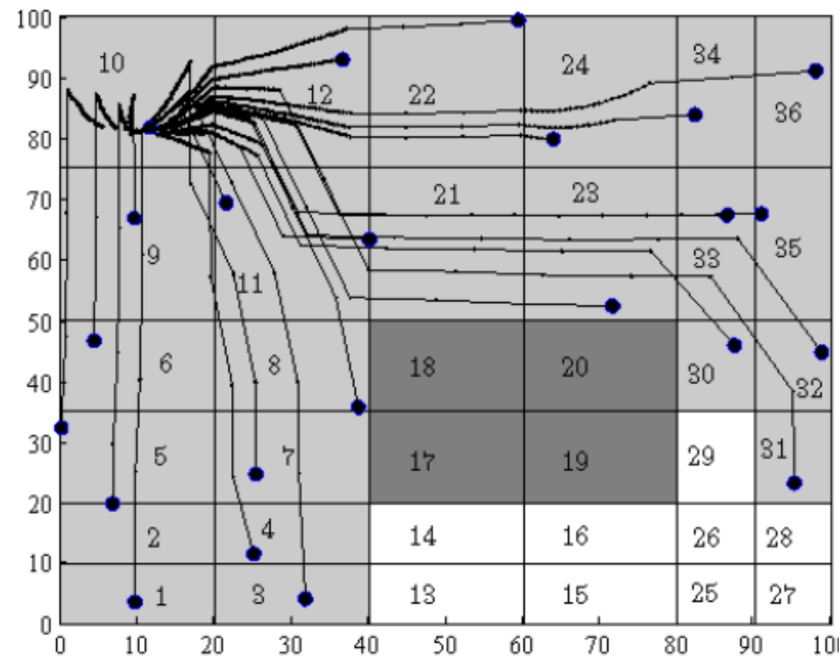
# LTL Control of PWA Systems

## Example: Rabin game




$$\Diamond\Box 10 \wedge \Box\neg(17 \vee 18 \vee 19 \vee 20)$$



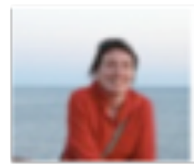




Matlab tool: "conPAS"
(hyness.bu.edu/software)

If stuttering is not accounted for, only 10 is a satisfying initial region.

# Acknowledgements

Boyan Yordanov



Jana Tumova