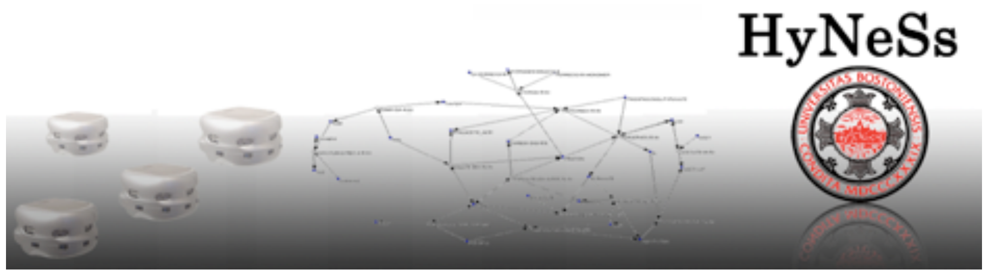
Hybrid and Networked Systems Lab



Formal Verification and Control for Discrete-Time Linear Systems

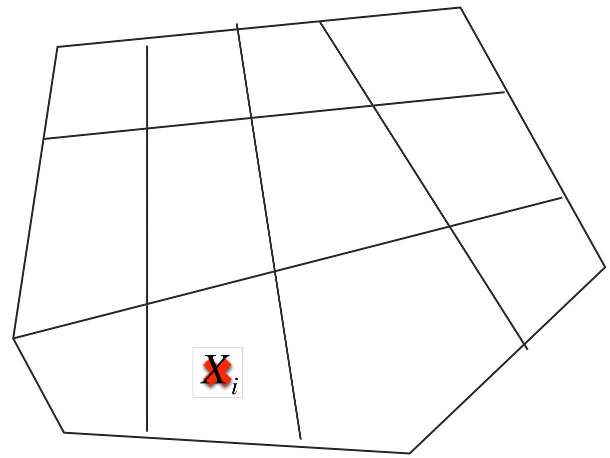
Calin Belta

Mechanical Engineering and Systems Engineering
Boston University

Discrete-time PWA systems

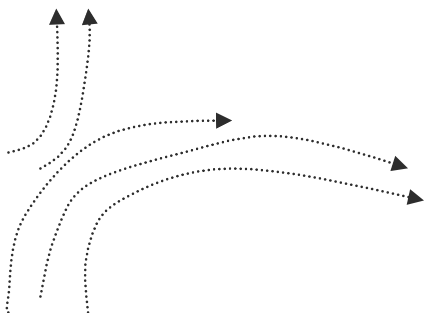
$$x_{k+1} = A_i x_k + B_i u_k,\ x_k \in X_i,\ i \in I,\ u_k \in U$$

$X_i, i \in I, U$ polytopes



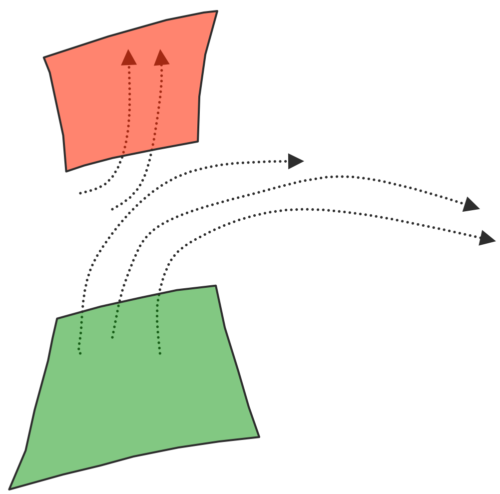
- Can approximate nonlinear systems with arbitrary accuracy [Lin and Unbehauen, 1992].
- Under mild assumptions, PWA systems are equivalent with several other classes of hybrid systems, including mixed logical dynamical (MLD), linear complementarity (LC), extended linear complementarity (ELC), and maxmin-plus-scaling (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]

Finite quotients of continuous-space systems



$\dot{x} = f(x)$ (or $x(k+1) = f(x(k))$)

Finite quotients of continuous-space systems

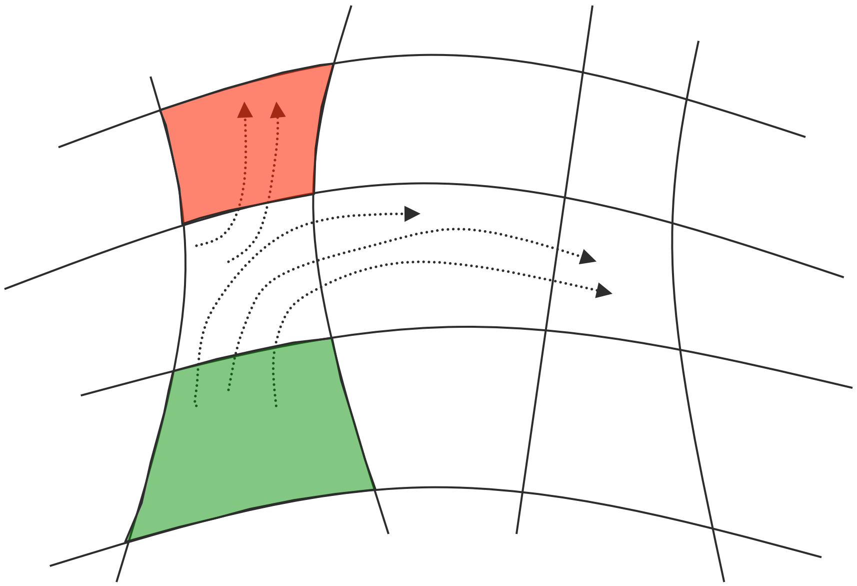


$$\dot{x} = f(x)$$

"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems

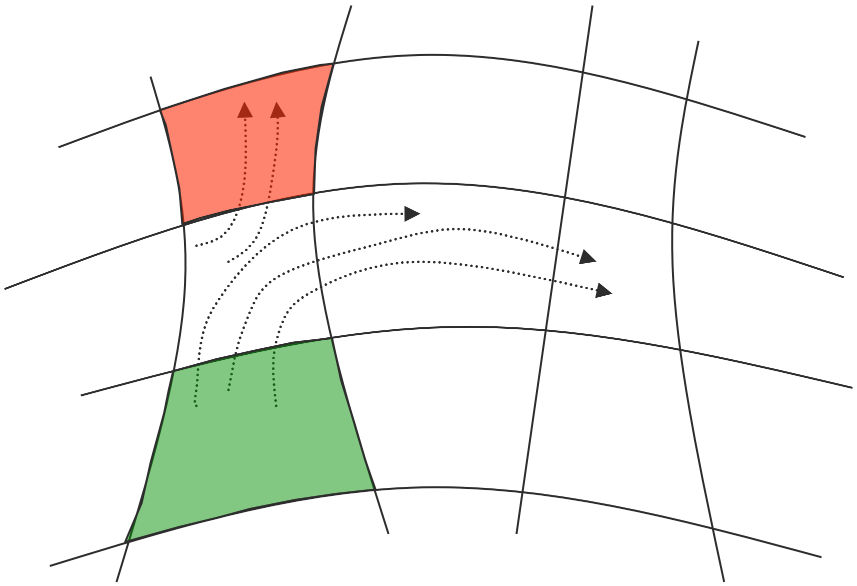


$$\dot{x} = f(x)$$

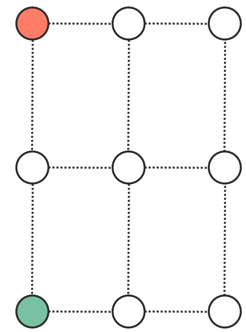
"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



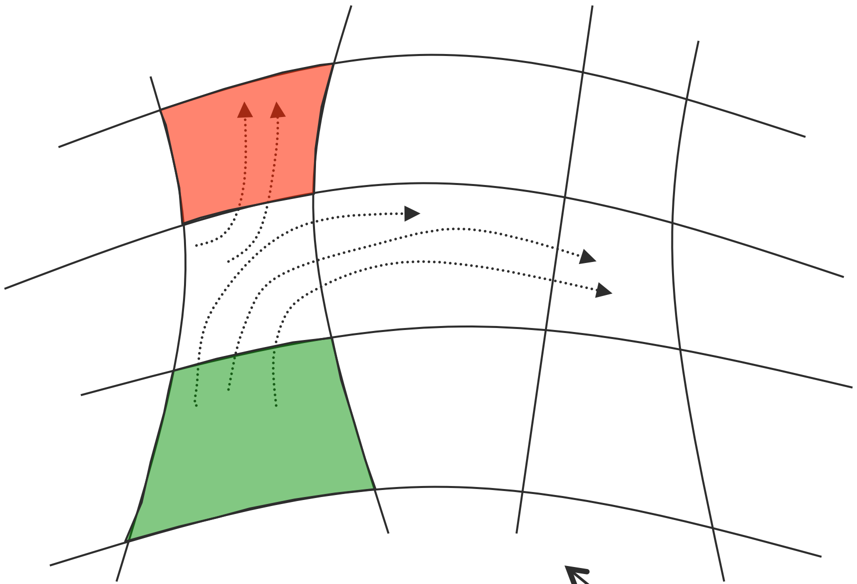
$$\dot{x} = f(x)$$



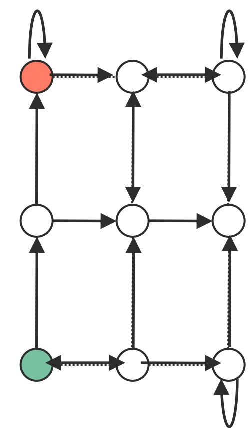
"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



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



ideally

$\Leftrightarrow$

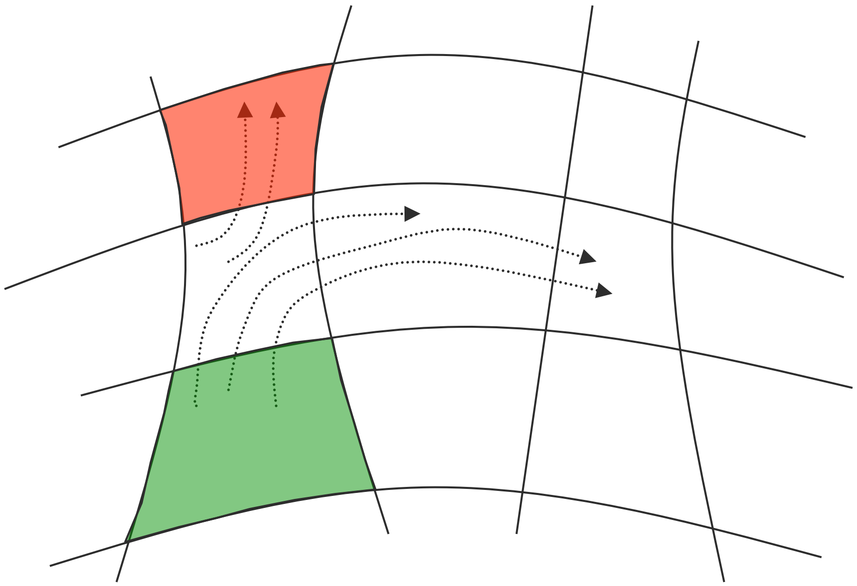
or, at least

$\Leftarrow$

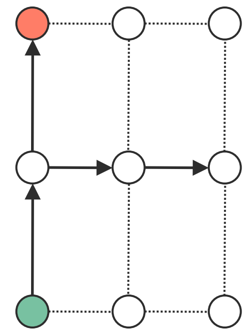
"There is no trajectory reaching from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



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

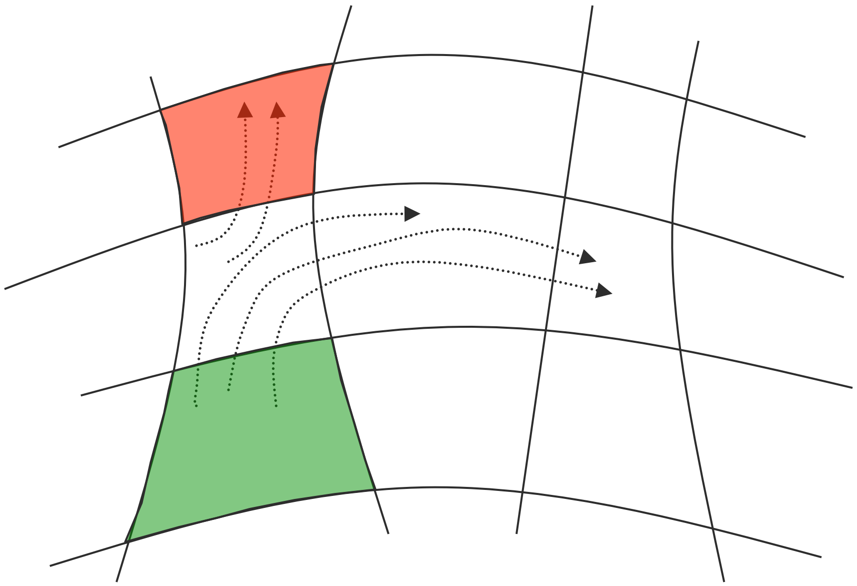


Assume we can decide whether there is a trajectory going from one region to an adjacent another

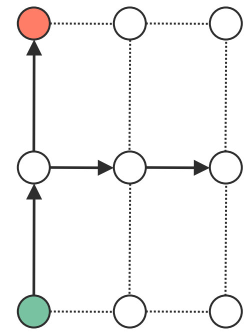
"There is no trajectory reaching from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



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

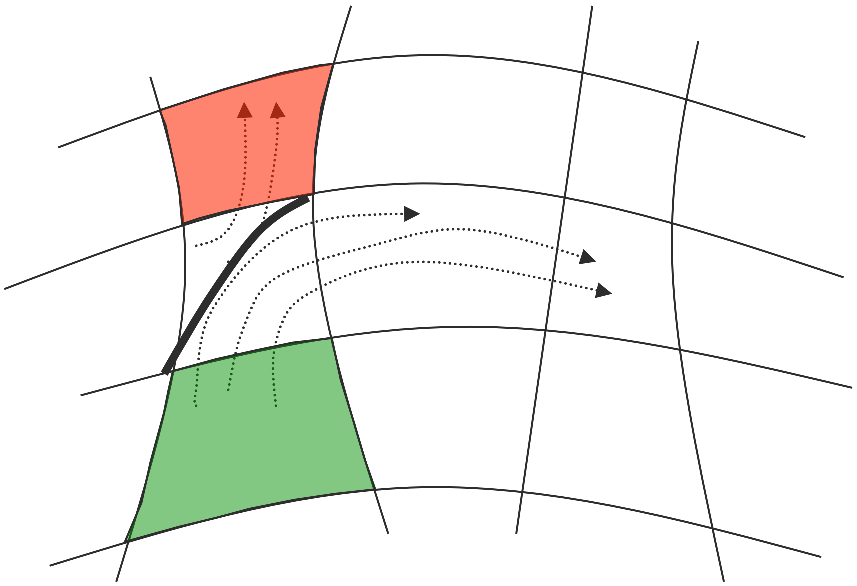


FALSE

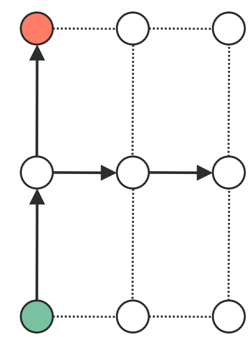
"There is no trajectory reaching from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



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

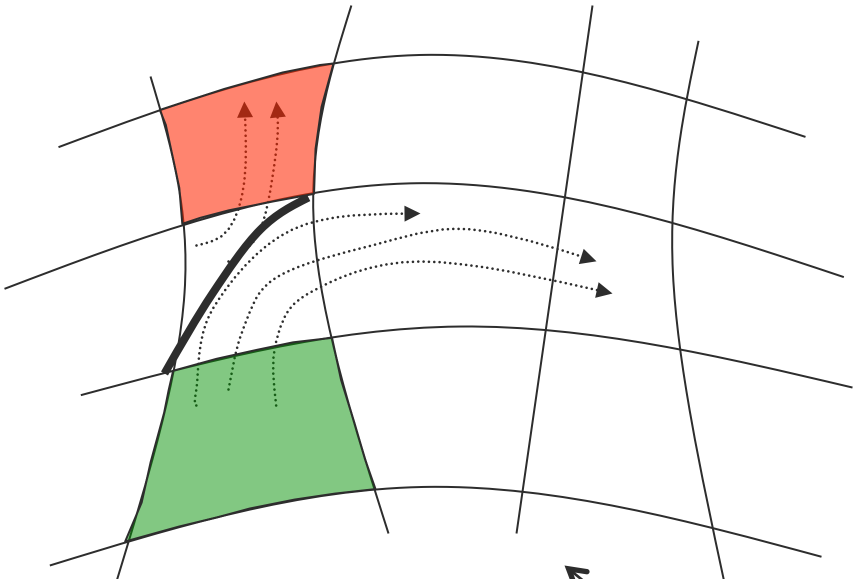


FALSE

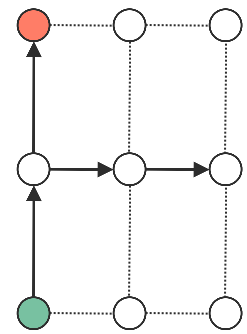
"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems



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



TRUE

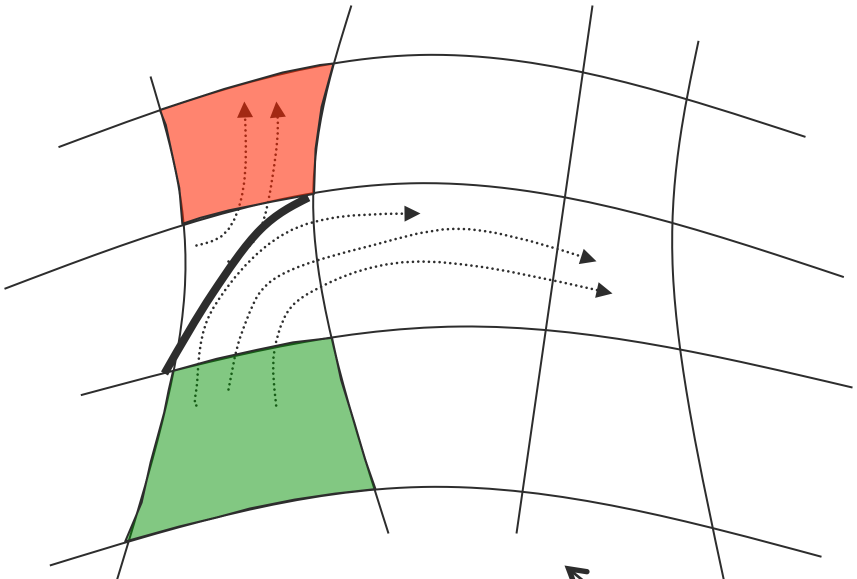
FALSE

"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems

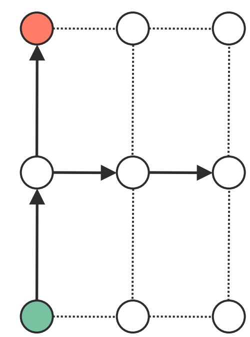
Is there something wrong with the quotient?



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

simulation

<



TRUE

FALSE

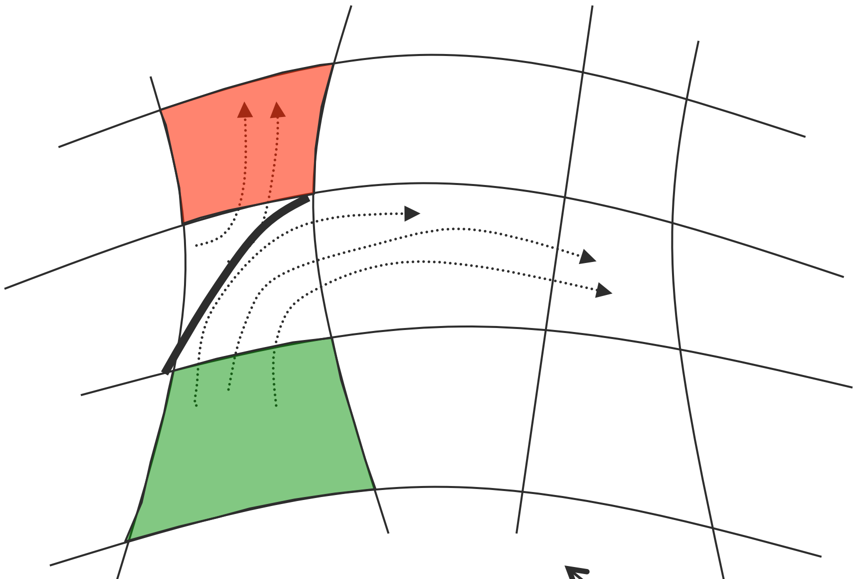
"There is no trajectory reaching from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems

Is there something wrong with the quotient?

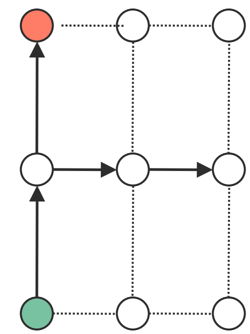
No, but it's too "rough" for proving this particular property.



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

simulation

$<$



TRUE

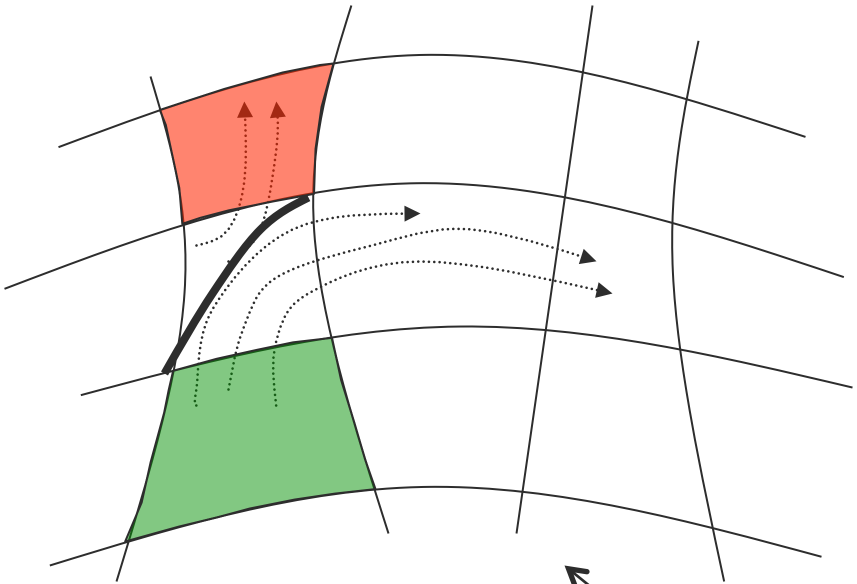
FALSE

"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

Finite quotients of continuous-space systems

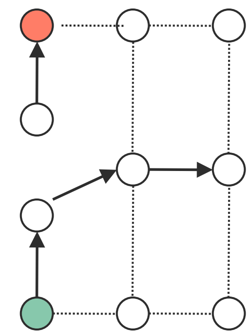
Refinement is necessary.



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

simulation

$<$



TRUE

$\Leftarrow$

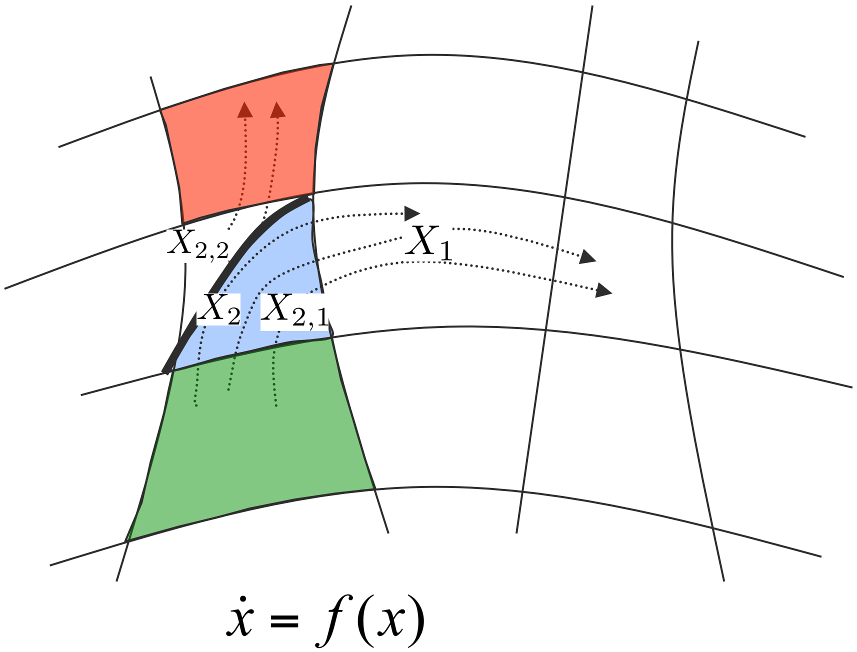
TRUE

"There is no trajectory reaching
from green to red" – True or False?

$\neg(green \wedge \Diamond red)$ for all trajectories

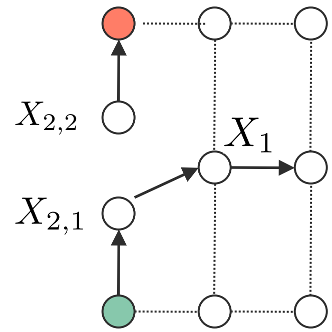
Finite quotients of continuous-space systems

Refinement is necessary.



simulation

$<$



$$Pre(X_1) = \{x \,|\, \exists t \geq 0 \, \exists x' \in X_1 \; s.t. \; x' = \phi(x,t)\}$$

$$X_{2,1} = Pre(X_1) \cap X_2$$

$$X_{2,2} = X_2 \setminus X_{2,1}$$

Finite quotients of continuous-space systems

Iterative refinement (bisimulation) algorithm

While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

$$X_{i,1} = X_i \cap Pre(X_j)$$

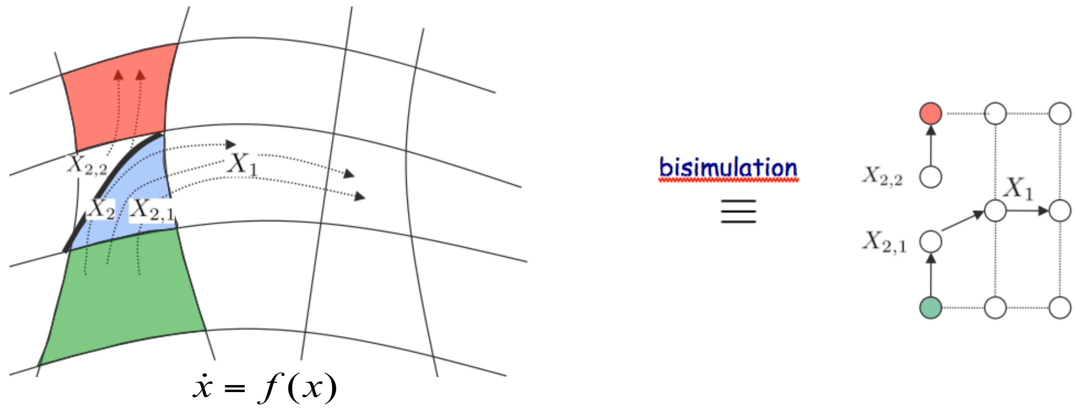
$$X_{i,2} = X_i \setminus X_{i,1}$$

remove X_i

add $X_{i,1}$, $X_{i,2}$

endwhile

A. Bouajjani, J.-C. Fernandez, and N. Halbwachs, 1991.



If the algorithm terminates, the finite quotient and the original system are called bisimilar, and the quotient can be used in lieu of the original system for verification from very general specs

Challenges:

Computability: set representation, computation of Pre, set intersection and difference, emptyness of sets

Termination: finite number of iterations

Decidability = Computability & Termination -> very restrictive classes of systems (e.g., timed automata, multi-rate automata, o-minimal systems)

R. Alur and D. L. Dill, 1994; R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho, 1993; G. Lafferriere, G. J. Pappas, and S. Sastry, 2000.

Finite quotients of continuous-space systems

Give up termination

While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

$X_{i,1} = X_i \cap Pre(X_j)$

$X_{i,2} = X_i \setminus X_{i,1}$

remove X_i

add $X_{i,1}$, $X_{i,2}$

construct the quotient

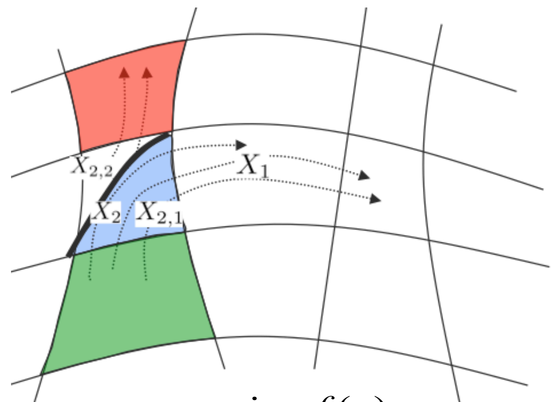
model check the quotient

if the spec is satisfied

break

endif

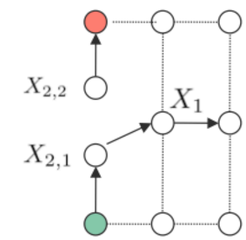
endwhile



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

simulation

<



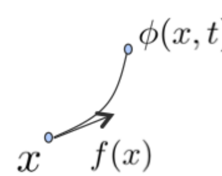
A. Chutinan and B. H. Krogh, 2001.

Verification only against universal properties, i.e., if all the trajectories of the quotient satisfy a spec, then all the trajectories of the original system satisfy the spec.

Computability:

- Still limited to very restrictive classes (should allow for quantifier elimination)
- Computation is very expensive

$$Pre(X_1) = \{x \mid \exists t \geq 0\, \exists x' \in X_1\ s.t.\ x' = \phi(x,t)\}$$

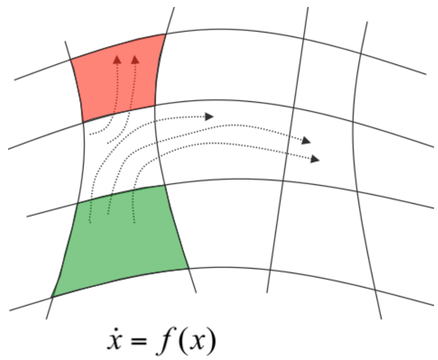


G. Lafferriere, G. J. Pappas, and S. Yovine, 2001.

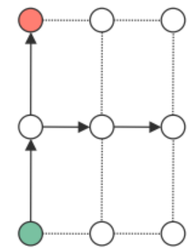
Finite quotients of continuous-space systems

Give up computation of Pre

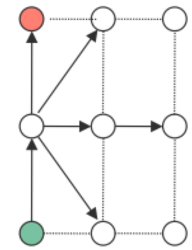
While TRUE
construct (an over-approximation of) the quotient
model check the quotient
if the spec is satisfied
break;
endif
refine (using some
partitioning scheme)
endwhile



simulation
<



simulation
<



$$\overline{Post}(X) \supseteq Post(X) = \{x' \mid \exists x \in X \, \exists t > 0 \; s.t. \; x' = \phi(x,t)\}$$

Continuous-time continuous-space polynomial dynamics and semi-algebraic regions (still requires quantifier elimination)

A. Tiwari and G. Khanna, 2002.

Continuous-time continuous-space affine and multi-affine dynamics and polytopic / rectangular / regions

L.C.G.J.M. Habets and J.H. van Schuppen, 2004; C. Belta and L.C.G.J.M. Habets, 2006

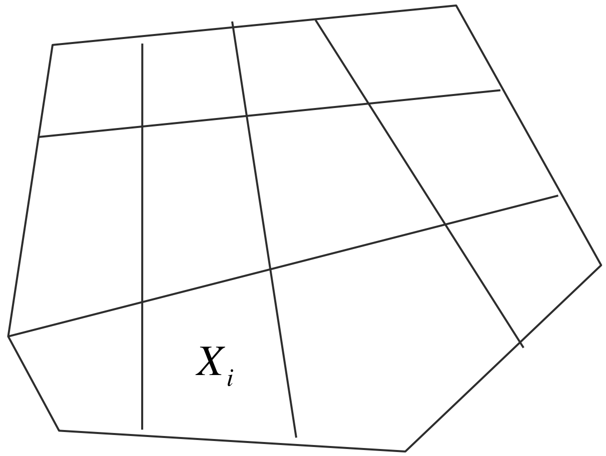
M. Kloetzer and C. Belta, HSCC 2006, TIMC 2012

Verification for discrete-time PWA systems

$$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$$

$X_i, i \in I$ polytopes

$A_i, i \in I$ invertible

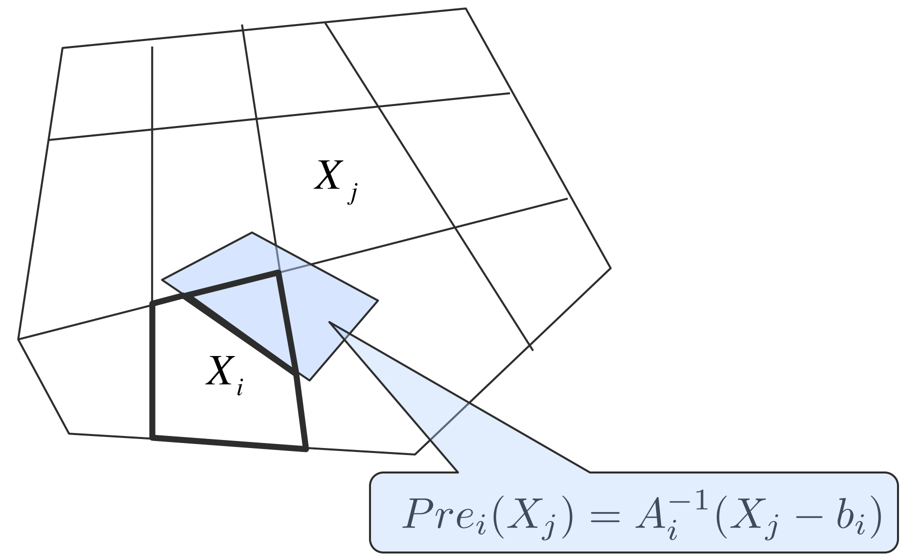


Verification for discrete-time PWA systems

$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$

$X_i, i \in I$ polytopes

$A_i, i \in I$ invertible



While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

$X_{i,1} = X_i \cap Pre(X_j)$

$X_{i,2} = X_i \setminus X_{i,1}$

remove X_i
add $X_{i,1}$, $X_{i,2}$
construct the quotient
model check the quotient
if the spec is satisfied
break
endif
endwhile

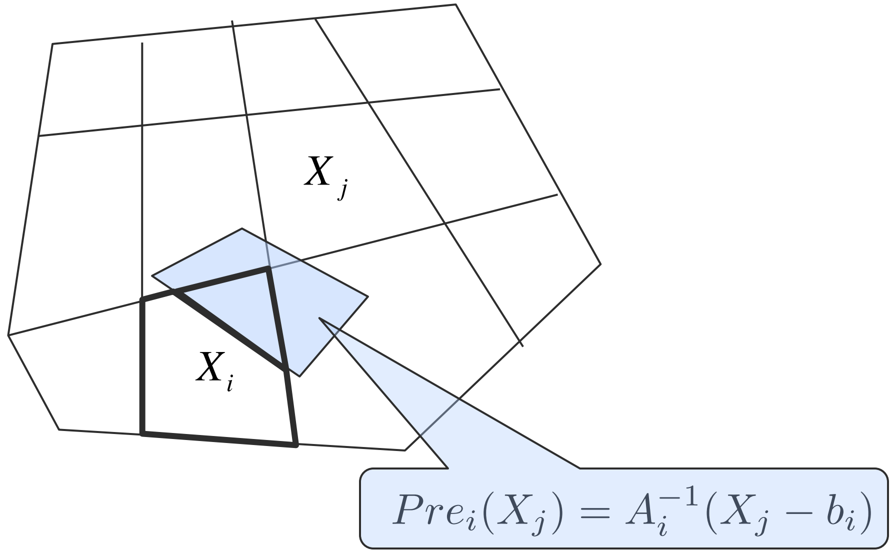
Everything is computable!

Verification for discrete-time PWA systems

$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$

$X_i, i \in I$ polytopes

$A_i, i \in I$ invertible



While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

$X_{i,1} = X_i \cap Pre(X_j)$

$X_{i,2} = X_i \setminus X_{i,1}$

remove X_i
add $X_{i,1}$, $X_{i,2}$
construct the quotient
model check the quotient
if the spec is satisfied
break
endif
endwhile

Everything is computable!

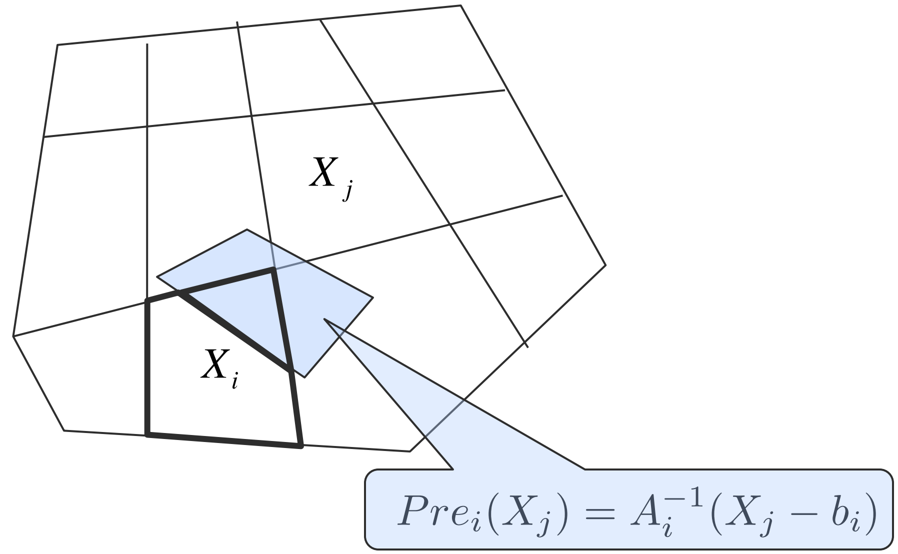
Problem Formulation: Find the largest subset of $\bigcup_{i \in I} X_i$ such that all the trajectories originating there satisfy an LTL formula ϕ over I .

Verification for discrete-time PWA systems

$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$

$X_i, i \in I$ polytopes

$A_i, i \in I$ invertible



While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

$X_{i,1} = X_i \cap Pre(X_j)$

$X_{i,2} = X_i \setminus X_{i,1}$

remove X_i
add $X_{i,1}$, $X_{i,2}$
construct the quotient
model check the quotient
if the spec is satisfied
break
endif
endwhile

Everything is computable!

Can be optimized by checking with both ϕ and $\neg\phi$ and partitioning only if necessary (no need to refine regions where the formula or its negation is satisfied at the corresponding state of the quotient).

Problem Formulation: Find the largest subset of $\bigcup_{i \in I} X_i$ such that all the trajectories originating there satisfy an LTL formula ϕ over I .

B. Yordanov and C. Belta, IEEE TAC 2010

Verification for discrete-time PWA systems

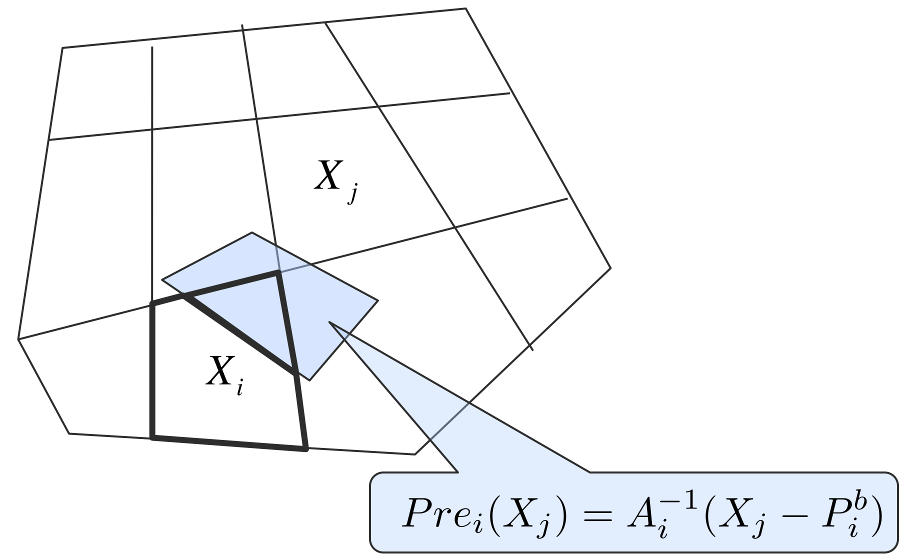
$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$

$X_i, i \in I$ polytopes $P_i^b, i \in I$ polytopes

$A_i, i \in I$ invertible

What if $b_i \in P_i^b, i \in I$?

Everything still works with extra computational overhead.



While there exist X_i , X_j such that $\emptyset \subset X_i \cap Pre(X_j) \subset X_i$

  $X_{i,1} = X_i \cap Pre(X_j)$

  $X_{i,2} = X_i \setminus X_{i,1}$

  remove X_i
  add $X_{i,1}$, $X_{i,2}$
  construct the quotient
  model check the quotient
  if the spec is satisfied
    break
  endif
endwhile

Everything is computable!

Can be optimized by checking with both ϕ and $\neg\phi$ and partitioning only if necessary (no need to refine regions where the formula or its negation is satisfied at the corresponding state of the quotient).

Problem Formulation: Find the largest subset of $\bigcup_{i \in I} X_i$ such that all the trajectories originating there satisfy an LTL formula ϕ over I .

B. Yordanov and C. Belta, IEEE TAC 2010

Verification for discrete-time PWA systems

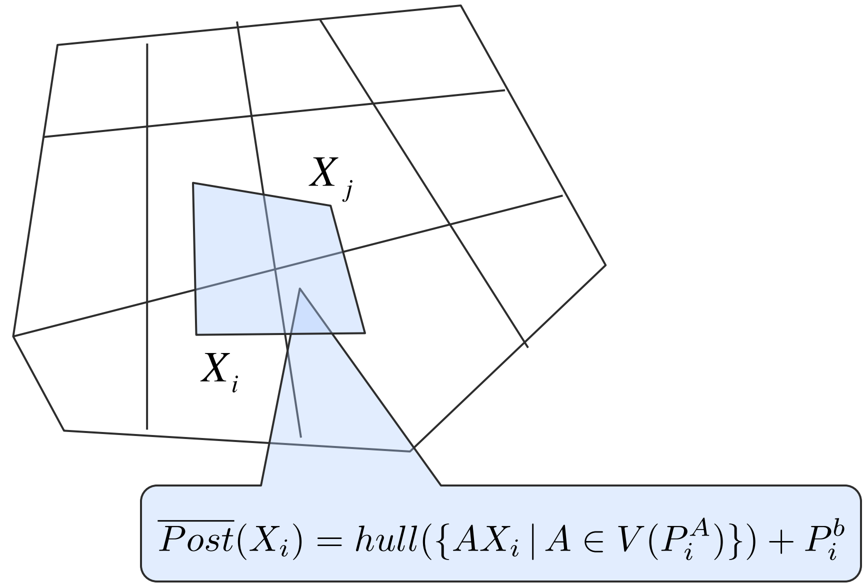
$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$

$X_i, i \in I$ polytopes $\quad P_i^b, i \in I$ polytopes

$A_i, i \in I$ invertible $\quad P_i^A, i \in I$ polytopes

What if $b_i \in P_i^b, i \in I$ and $A_i \in P_i^A, i \in I$?

Pre is not computable anymore. A polyhedral over-approximation of Post is computable.



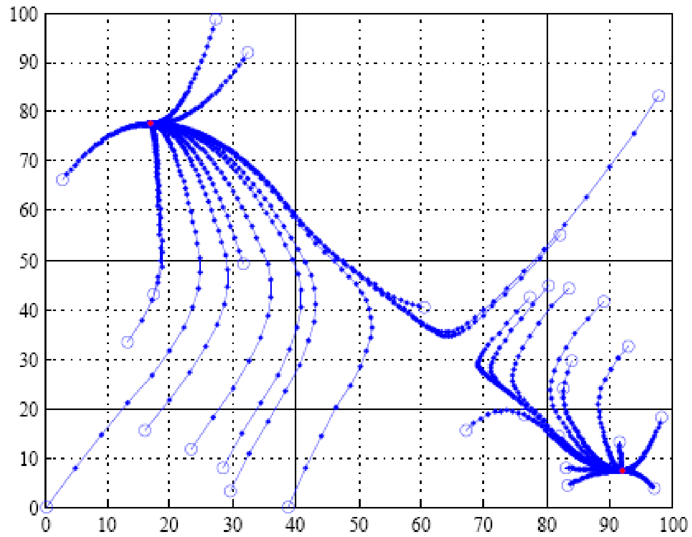

```
While TRUE
        construct (an over-approximation of) the quotient
        model check the quotient
        if the spec is satisfied
                break;
        endif
        refine (using arbitrary partitioning schemes)
endwhile
```

Problem Formulation: Find the largest subset of $\bigcup_{i \in I} X_i$ such that all the trajectories originating there satisfy an LTL formula ϕ over I .

B. Yordanov and C. Belta, IEEE TAC 2010

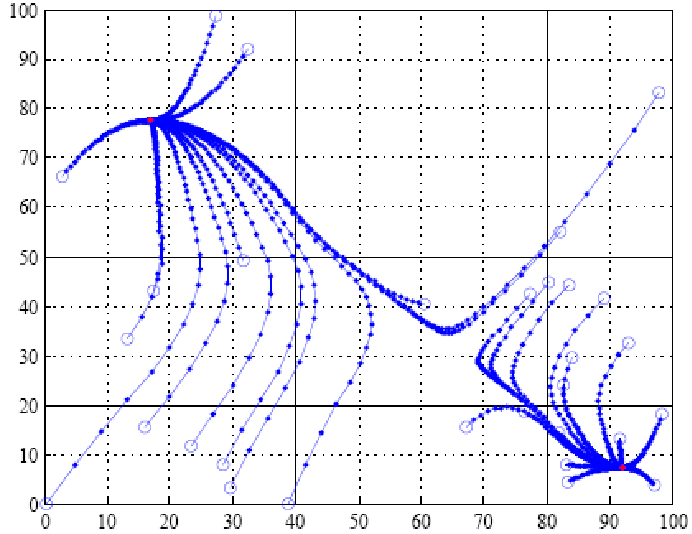
Verification for discrete-time PWA systems

Example: toggle switch

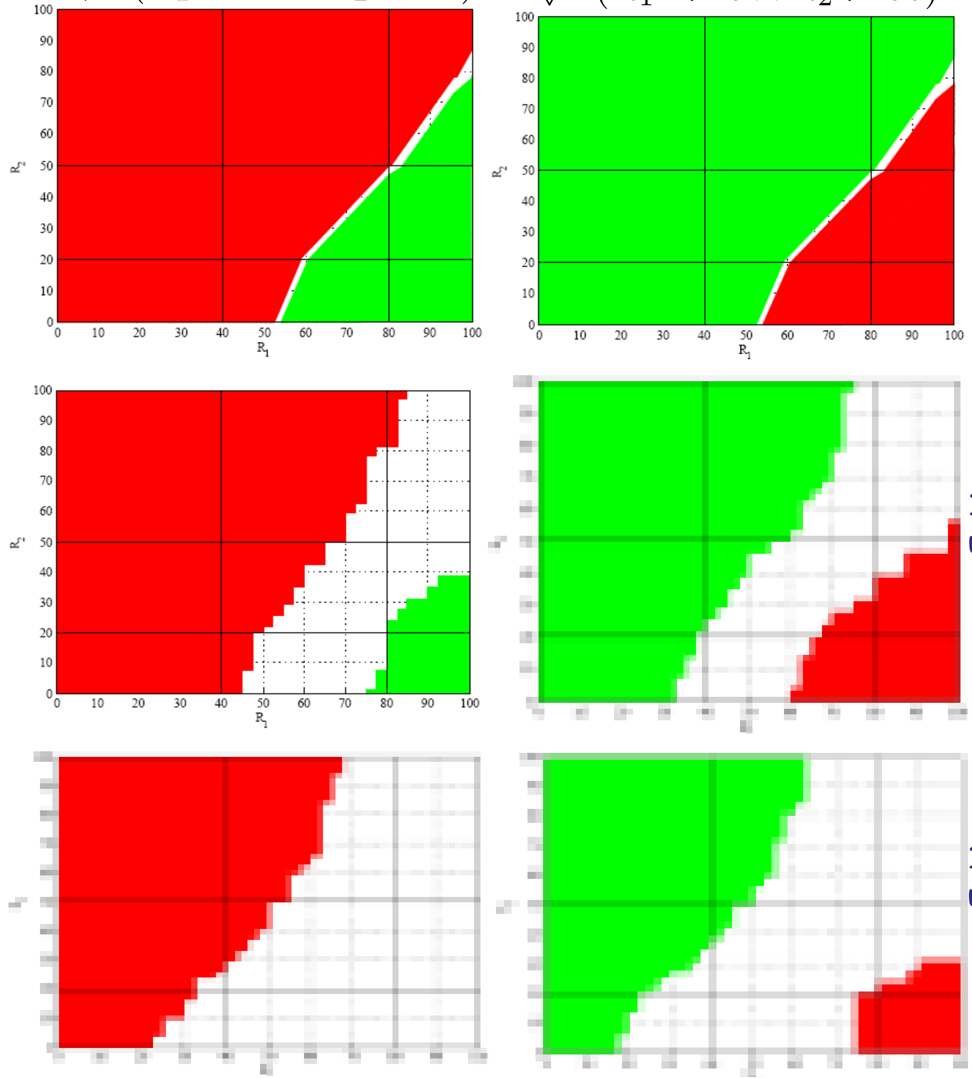


Verification for discrete-time PWA systems

Example: toggle switch



$\Diamond\Box(R_1 > 80 \wedge R_2 < 20)$ $\Diamond\Box(R_1 < 40 \wedge R_2 > 50)$



Fixed parameters

1% param uncertainty

10% param uncertainty

Initial satisfying states

Initial violating states

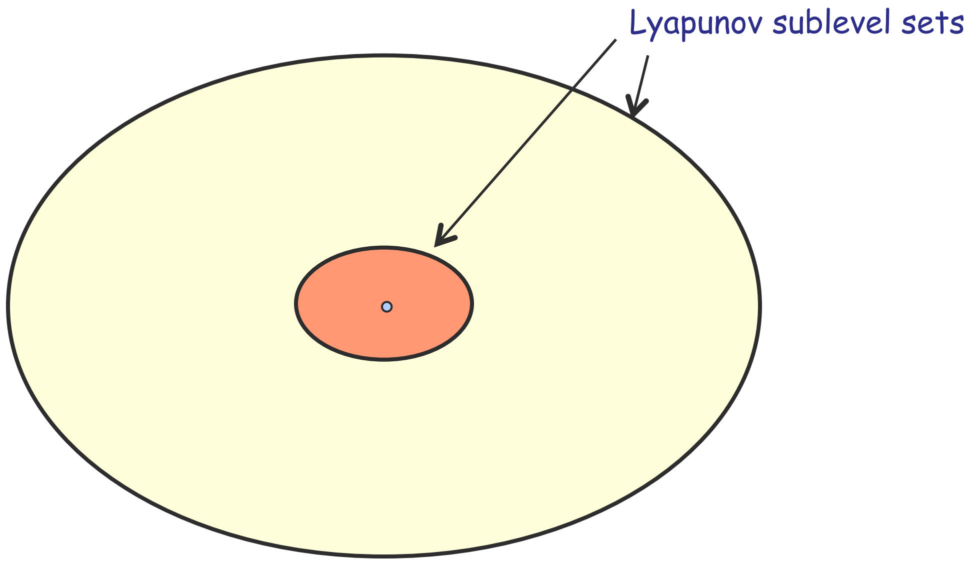
B. Yordanov and C. Belta, IEEE TAC 2010

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

Verification for discrete-time systems

Using Lyapunov functions to construct finite bisimulations

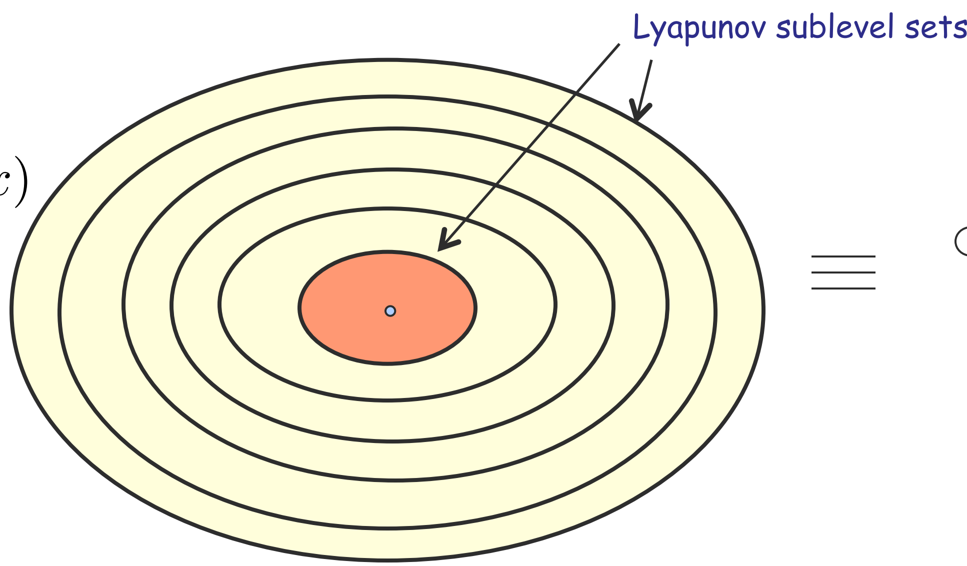
$$x_{k+1} = \Phi(x_k)$$



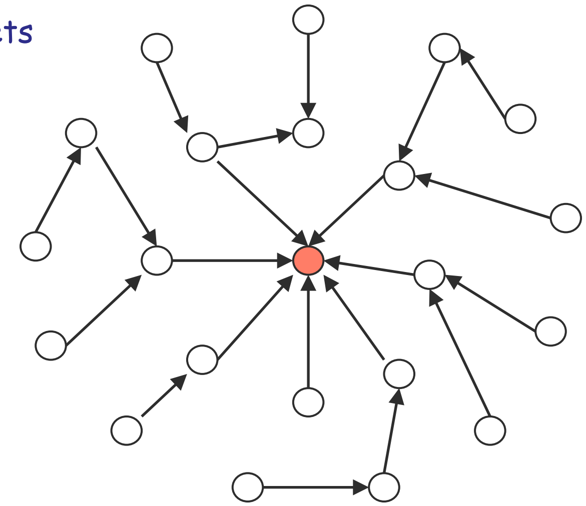
Verification for discrete-time systems

Using Lyapunov functions to construct finite bisimulations

$$x_{k+1} = \Phi(x_k)$$
$$V(\Phi(x)) \leq \rho V(x)$$



$\equiv$



Algorithm: Slice the space in between two sublevel sets into N slices (N determined by the contraction rate); Starting from the inner-most slice, compute the pre-image of the slice and intersect it with all the other slices.

Theorem: At the ith iteration, the partition of the inner region bounded by the ith slice is a bisimulation. As a result, a bisimulation for the whole region is obtained in N steps

Applicability:
- we can only reason about the behavior of the system in between two sublevel sets (we should not mind that all trajectories of the system eventually disappear in the region closest to the origin)
- need to be able to compute the pre-image of a slice through the dynamics of the system and the intersections with other slices

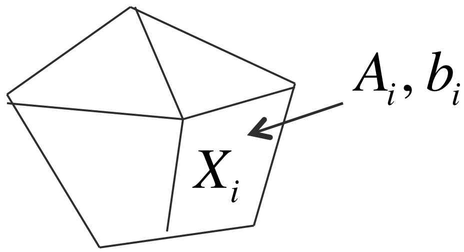
E. Aydin Gol, X.C. Ding,, M. Lazar, C. Belta ADHS 2012, CDC 2012

Verification for discrete-time systems

Using Lyapunov functions to construct finite bisimulations

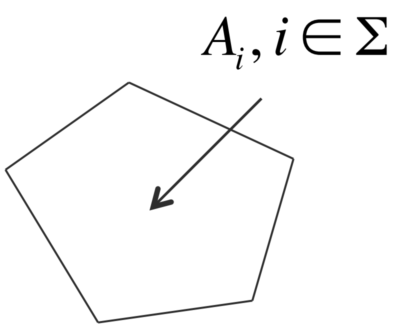
Computability

Discrete-time PWA systems



$$x_{k+1} = A_i x_k + b_i, x_k \in X_i, i \in I$$

Discrete-time switched linear systems



$$x_{k+1} = A_{\sigma(k)} x_k, \; \sigma(k) \in \Sigma$$

Lyapunov functions with polytopic sublevel sets can be constructed

$$V(x) = \|Lx\|_\infty$$

Blanchini 1994, Lazar 2010

Verification for discrete-time linear systems

Using Lyapunov functions to construct finite bisimulations

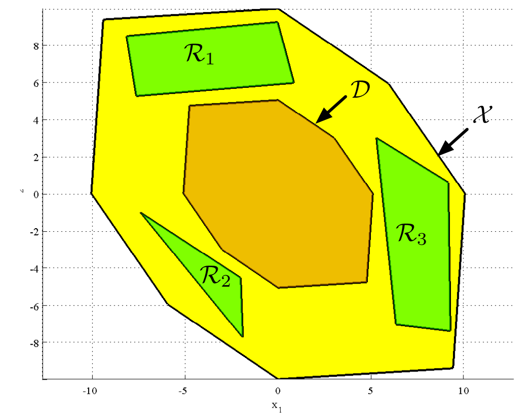
Example:

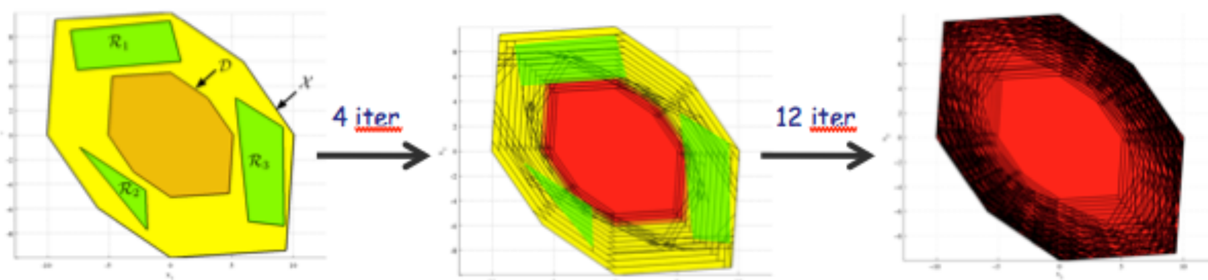
$$\Sigma = \{1, 2\} \qquad A_1 = \begin{pmatrix} -0.65 & 0.32 \\ -0.42 & -0.92 \end{pmatrix} \quad A_2 = \begin{pmatrix} 0.65 & 0.32 \\ -0.42 & -0.92 \end{pmatrix}$$

$$x_{k+1} = A_{\sigma(k)} x_k,\ \sigma(k) \in \Sigma$$

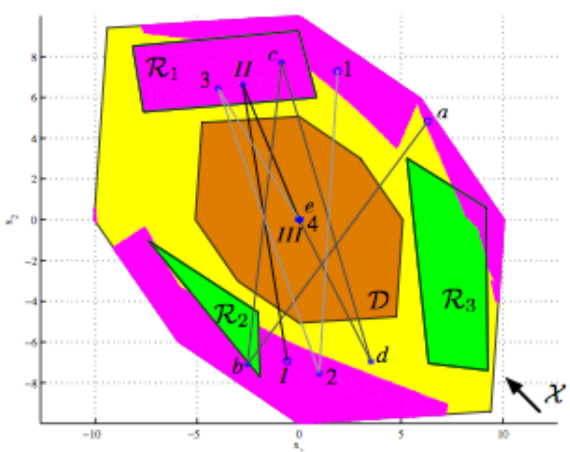
"A system trajectory never visits $\mathcal{R}_2$ and eventually visits $\mathcal{R}_1$. Moreover, if it visits $\mathcal{R}_3$ then it must not visit $\mathcal{R}_1$ at the next time step" can be translated to a scLTL formula:

$$\phi := (\neg \mathcal{R}_2 \,\mathsf{U}\, \Pi_{\mathcal{D}}) \wedge \mathsf{F}\, \mathcal{R}_1 \wedge ((\mathcal{R}_3 \Rightarrow \mathsf{X} \neg \mathcal{R}_1) \,\mathsf{U}\, \Pi_{\mathcal{D}})$$

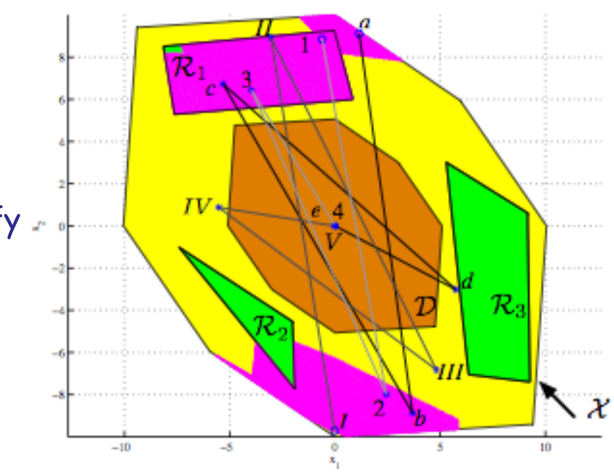




Purple: Sets of initial states for which there exists a switching strategy such that all trajectories satisfy the spec



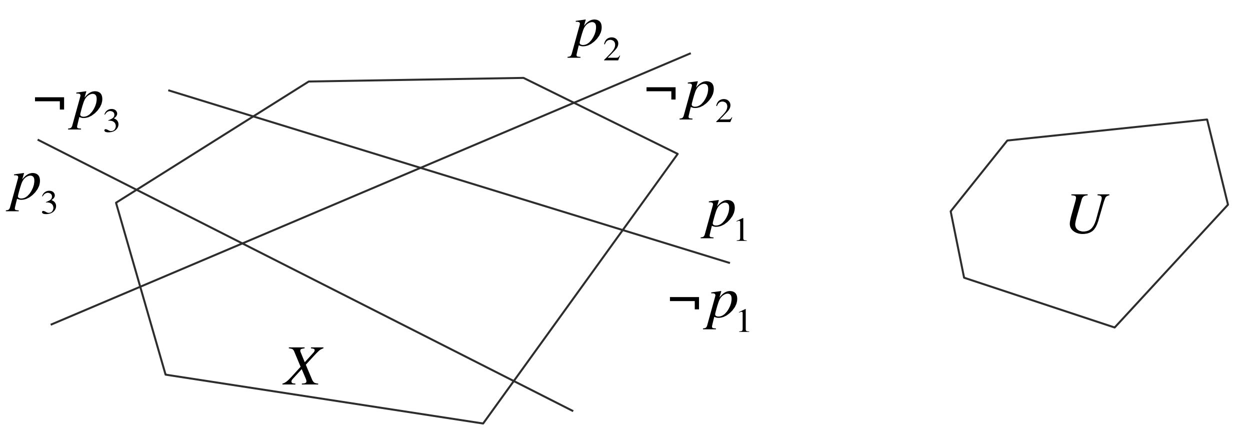
Purple: Sets of initial states for which all trajectories satisfy the spec under all possible switches



E. Aydin Gol, X.C. Ding, M. Lazar, C. Belta ADHS 2012, CDC 2012

TL control for discrete-time linear systems

$$x_{k+1} = Ax_k + Bu_k,\; x_k \in X,\; u_k \in U \qquad X, U \text{ polytopes}$$

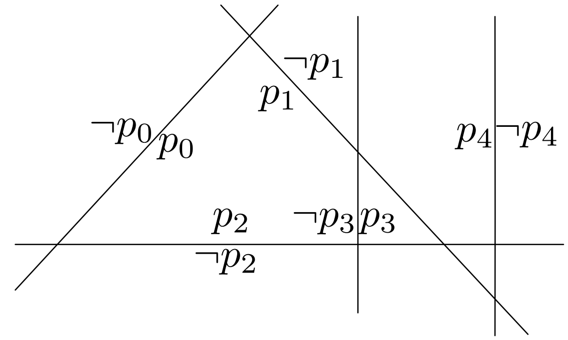


Problem Formulation: Find $X_0 \subseteq X$ and a state-feedback control strategy such that all trajectories of the closed loop system originating at X_0 satisfy an LTL formula ϕ over the linear predicates p_i

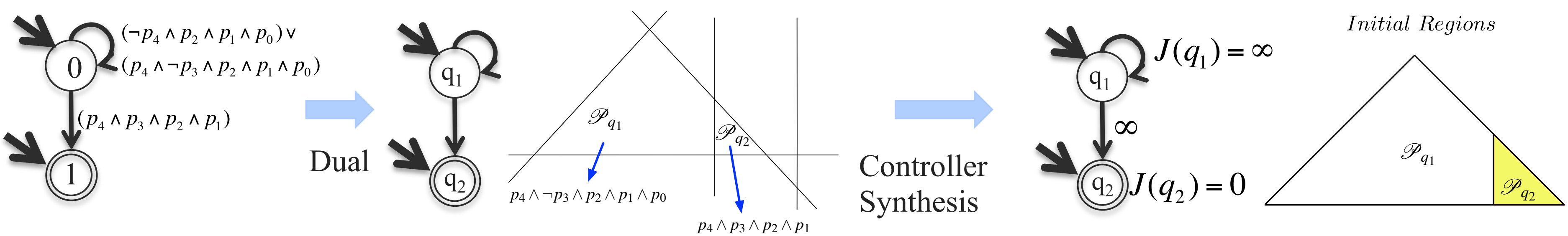
TL control for discrete-time linear systems

Approach: Language-guided controller synthesis and refinement

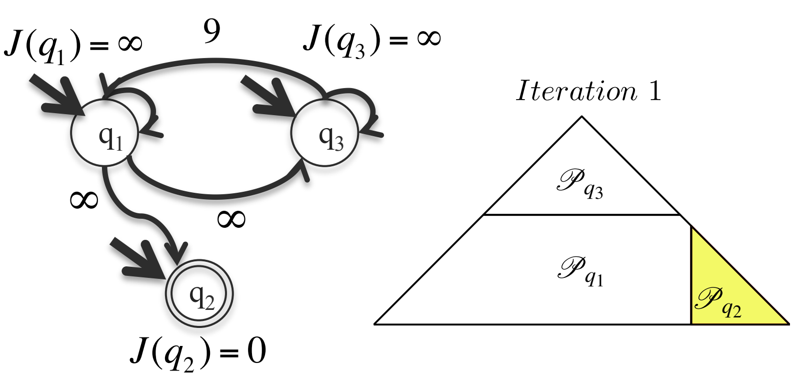
$$\Phi = (p_0 \wedge p_1 \wedge p_2) U (p_1 \wedge p_2 \wedge p_3 \wedge p_4)$$

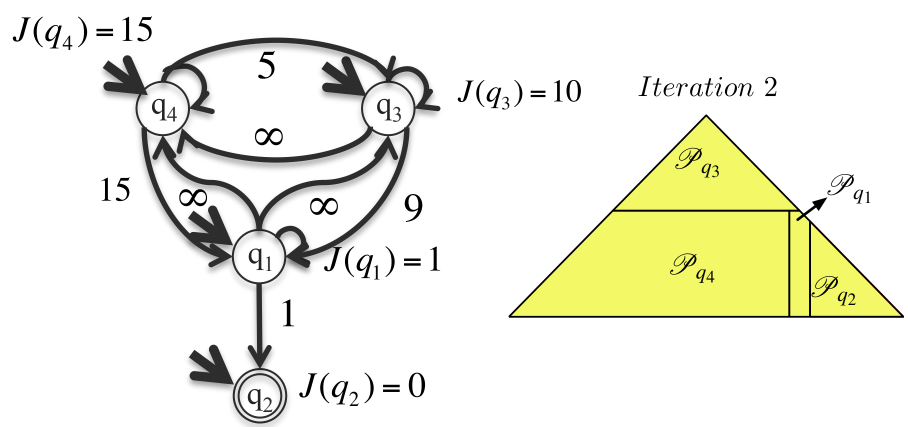


Latvala 2003



Refinement:





E. Aydin Gol, M. Lazar, and C. Belta, HSCC 2012

TL control for discrete-time linear systems

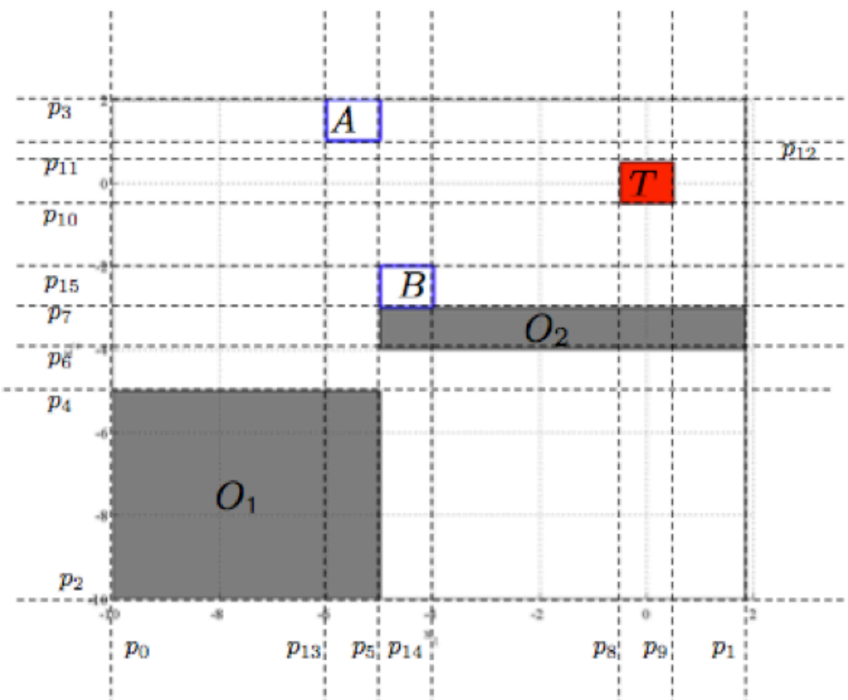
Example

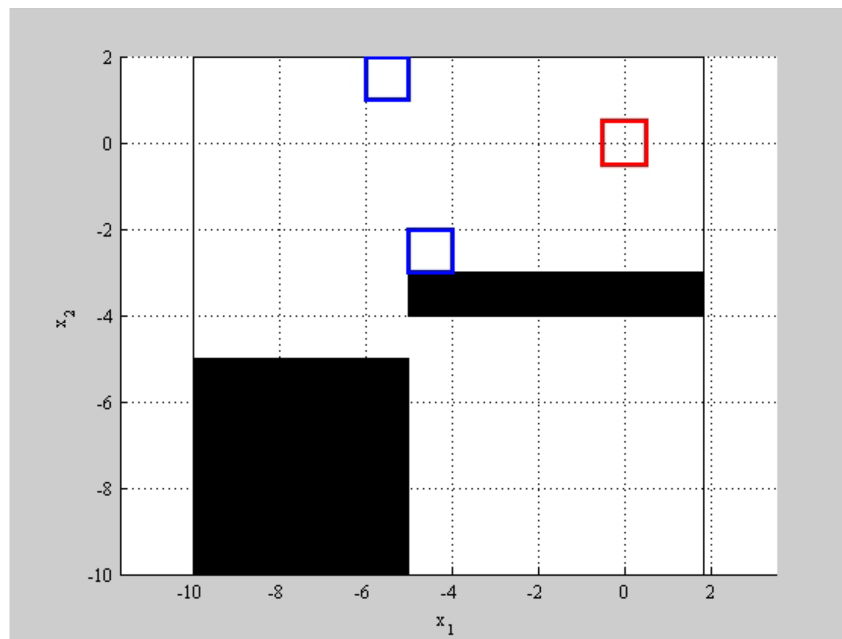
$$x_{k+1} = Ax_k + Bu_k, \quad x_k \in \mathbb{X},\ u_k \in \mathbb{U}$$

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

"Visit region A or region B before reaching the target while always avoiding the obstacles"

$$\Phi_2 = ((p_0 \wedge p_1 \wedge p_2 \wedge p_3 \wedge \neg(p_4 \wedge p_5) \wedge \neg(\neg p_5 \wedge \neg p_6 \wedge p_7)) \,\mathscr{U}\, (\neg p_8 \wedge p_9 \wedge \neg p_{10} \wedge p_{11})) \wedge (\neg(\neg p_8 \wedge p_9 \wedge \neg p_{10} \wedge p_{11}) \,\mathscr{U}\, ((p_5 \wedge \neg p_{12} \wedge \neg p_{13}) \vee (\neg p_5 \wedge \neg p_7 \wedge p_{14} \wedge p_{15})))$$

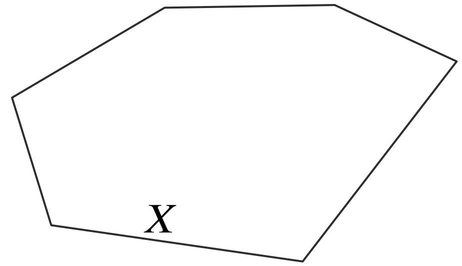


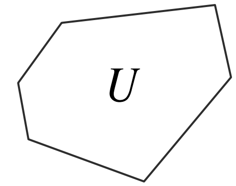


E. Aydin Gol, M. Lazar, and C. Belta, HSCC 2012

Optimal TL control for discrete-time linear systems

$$x_{k+1} = Ax_k + Bu_k, \quad x_k \in \mathbb{X},\ u_k \in \mathbb{U},$$





Initial state: x_0

Reference trajectories:

$x_0^r, x_1^r \ldots$

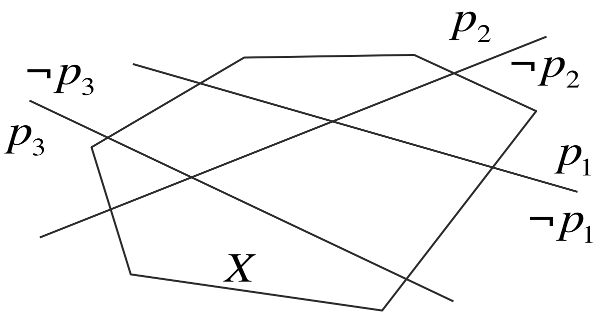
$u_0^r, u_1^r, \ldots$

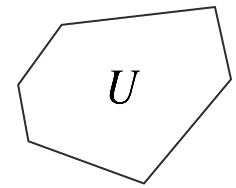
Observation horizon : N

$$\begin{aligned} C(x_k, \mathbf{u}_k) = &\ (x_{k+N} - x_{k+N}^r)^\top L_N (x_{k+N} - x_{k+N}^r) \\ &+ \sum_{i=0}^{N-1} \big\{ (x_{k+i} - x_{k+i}^r)^\top L (x_{k+i} - x_{k+i}^r) \\ &\quad + (u_{k+i} - u_{k+i}^r)^\top R (u_{k+i} - u_{k+i}^r) \big\}\,, \end{aligned}$$

Optimal TL control for discrete-time linear systems

$$x_{k+1} = Ax_k + Bu_k, \quad x_k \in \mathbb{X},\ u_k \in \mathbb{U},$$





Initial state: x_0

Reference trajectories:

$x_0^r, x_1^r \ldots$

$u_0^r, u_1^r, \ldots$

Observation horizon : N

$$C(x_k, \mathbf{u}_k) = (x_{k+N} - x_{k+N}^r)^\top L_N (x_{k+N} - x_{k+N}^r) + \sum_{i=0}^{N-1} \left\{ (x_{k+i} - x_{k+i}^r)^\top L (x_{k+i} - x_{k+i}^r) + (u_{k+i} - u_{k+i}^r)^\top R (u_{k+i} - u_{k+i}^r) \right\},$$

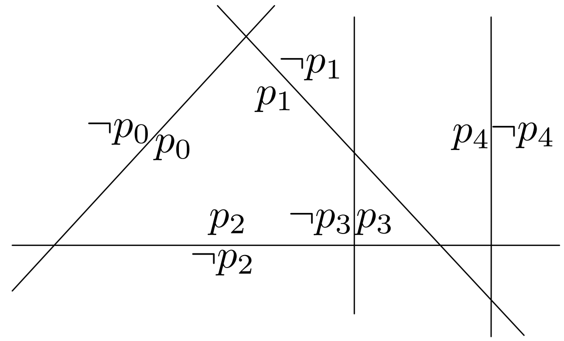
Syntactically co-safe LTL formula over linear predicates p_i

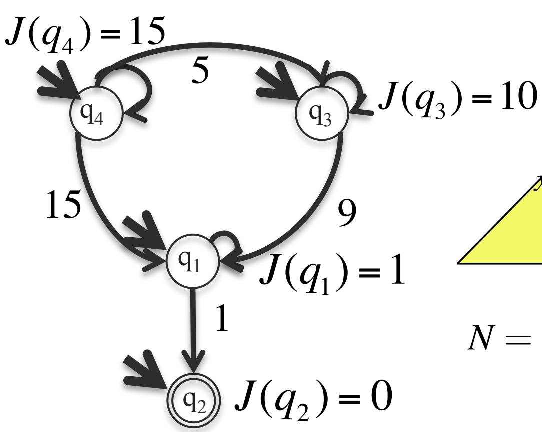
Problem Formulation: Find an optimal state-feedback control strategy such that the trajectory originating at x_0 satisfies the formula.

Optimal TL control for discrete-time linear systems

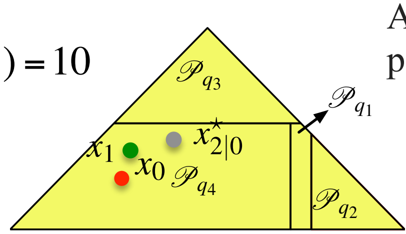
Approach

$$\Phi = (p_0 \wedge p_1 \wedge p_2) U (p_1 \wedge p_2 \wedge p_3 \wedge p_4)$$





Refined dual automaton



$N = 2, \quad x_0^r x_1^r x_2^r$

$u_0^r u_1^r$

Automaton paths: $q_4 q_4 q_4$

$q_4 q_4 q_3$

$q_4 q_3 q_3$

$q_4 q_3 q_1$

$q_4 q_4 q_1$

$q_4 q_1 q_1$

$q_4 q_1 q_2$

$\mathscr{P}_{q4} \mathscr{P}_{q4} \mathscr{P}_{q4}$

$\mathscr{P}_{q4} \mathscr{P}_{q4} \mathscr{P}_{q3}$

$\mathscr{P}_{q4} \mathscr{P}_{q3} \mathscr{P}_{q3}$

•
•
•

$$\min C(x_k, \mathbf{u}_k),$$

subject to

$u_{i|k} \in \mathbb{U}, \qquad i = 0, \ldots, N-1,$

$x_{i|k} \in \mathscr{P}_{q_{i|k}}, \qquad i = 1, \ldots, N,$

$V(q_{N|k}, x_{N|k}) < V(q^*_{N|k-1}, x^*_{N|k-1}).$

- Solve an optimization problem for each automaton path.(at each stage)
- Progress constraint: Distance to a satisfying automaton state eventually decreases.

E. Aydin Gol, et.al. HSCC 2013

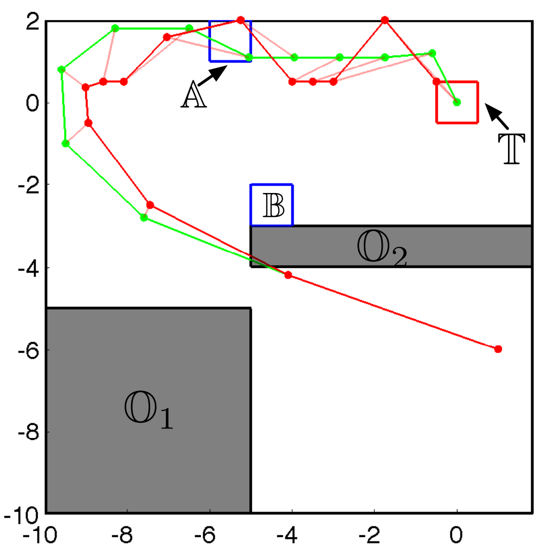
Optimal TL control for discrete-time linear systems

Example

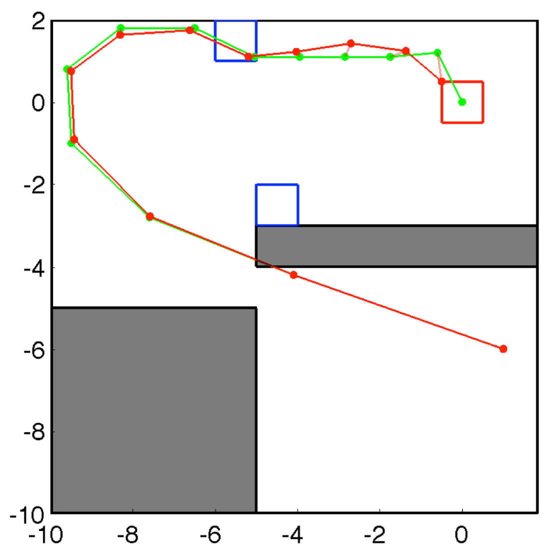
$$x_{k+1} = Ax_k + Bu_k, \quad x_k \in \mathbb{X},\ u_k \in \mathbb{U}$$

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$$

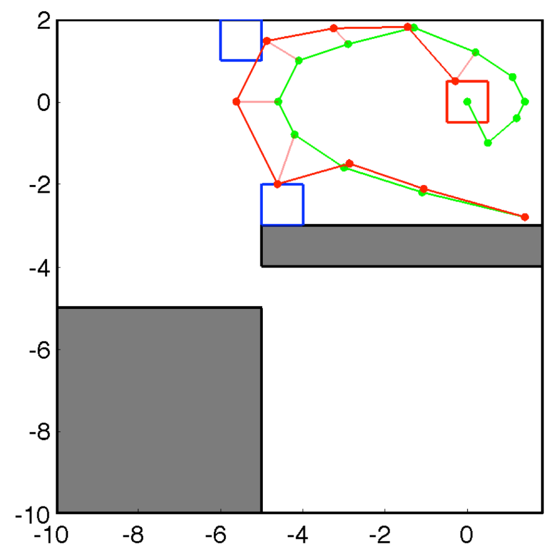
"Visit region A or region B before reaching the target while always avoiding the obstacles"



N = 2
total cost = 29.688



N = 4
total cost = 0.886



N = 6
total cost = 5.12

Reference trajectory
Controlled trajectory

Reference trajectory violates the specification

E. Aydin Gol, et.al., HSCC 2013

Acknowledgements

Ebru Aydin Gol

Boyan Yordanov
(now at Microsoft
Research)