Algorithmic Analysis of Polygonal Hybrid Systems

GERARDO SCHNEIDER

VERIMAG

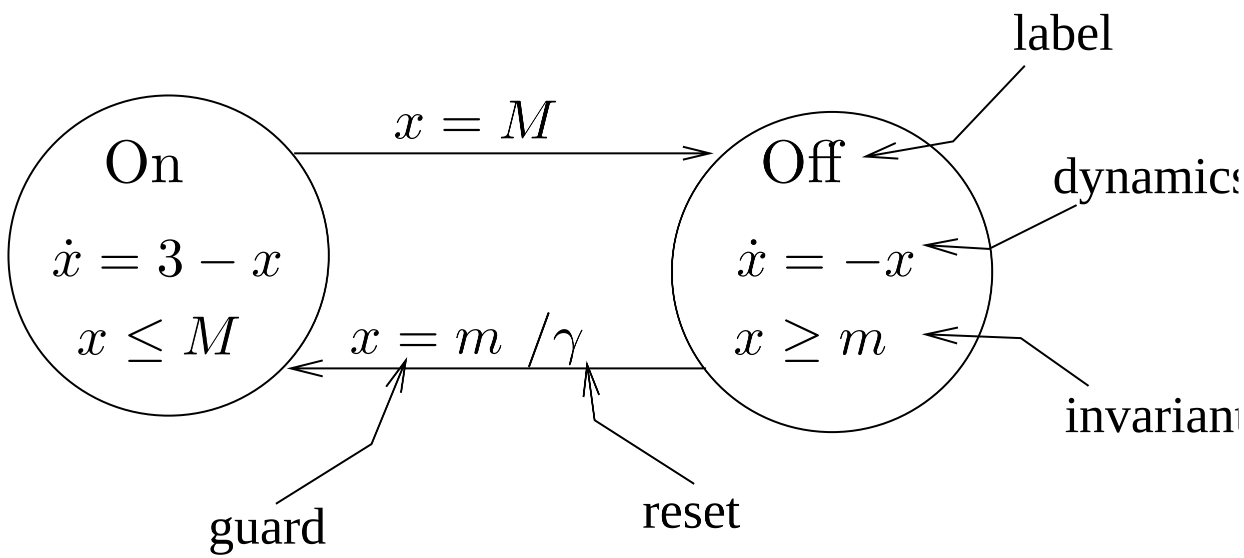
GRENOBLE

Hybrid Systems

- Hybrid Systems: interaction between discrete and continuous behaviors
- Examples: thermostat, automated highway systems, air traffic management systems, robotic systems, chemical plants, etc.

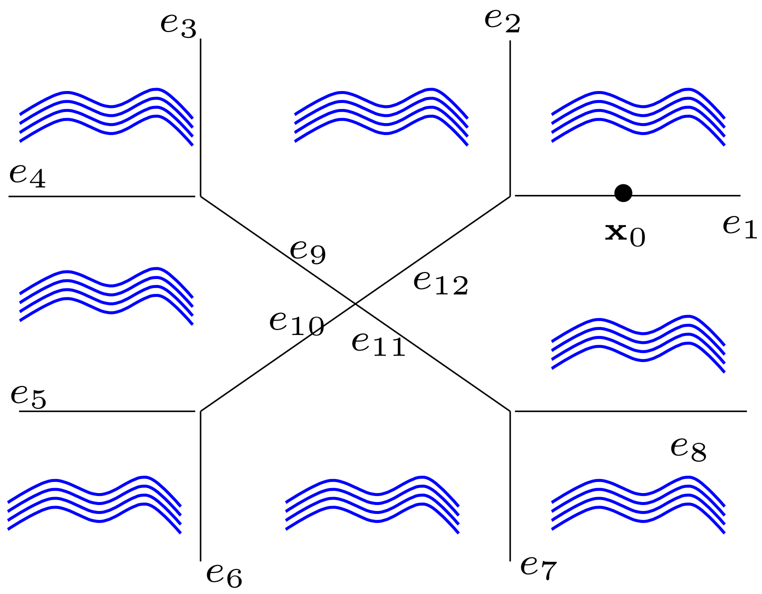
Hybrid Systems

Model: Hybrid Automata



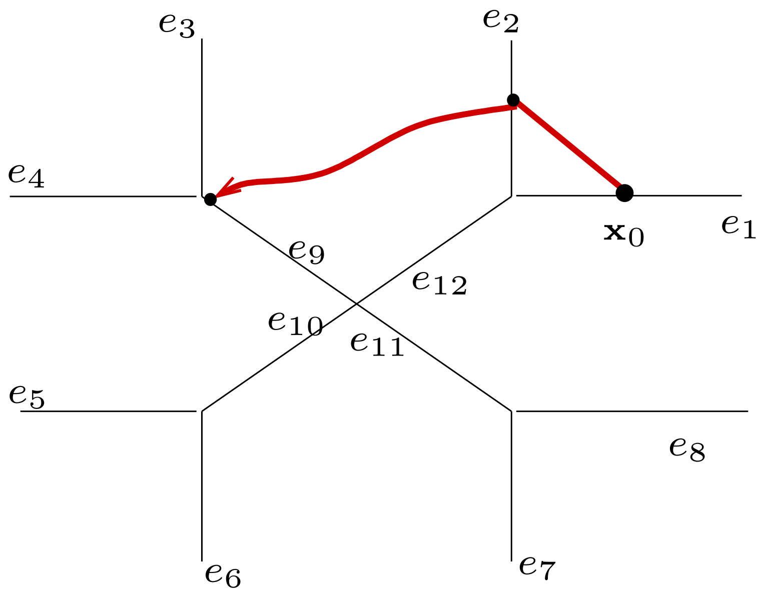
Hybrid Systems

Example: Swimmer in a whirlpool



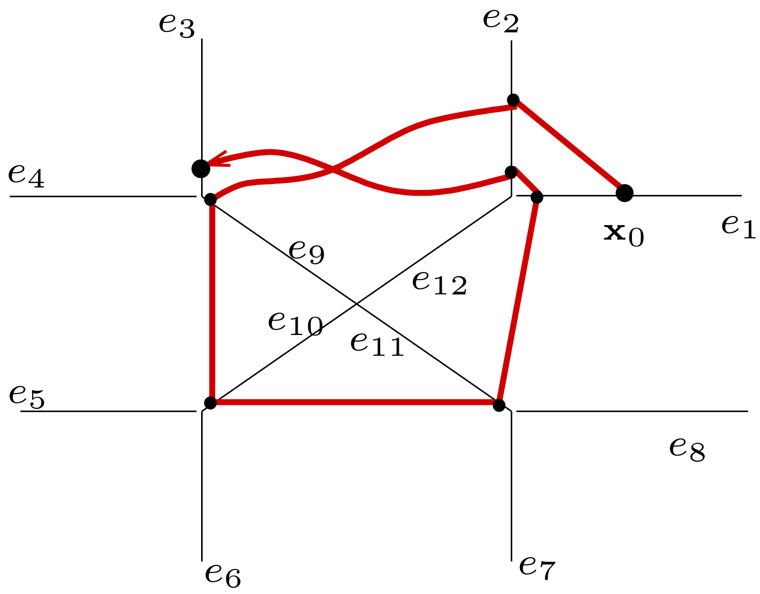
Hybrid Systems

Example: Swimmer in a whirlpool



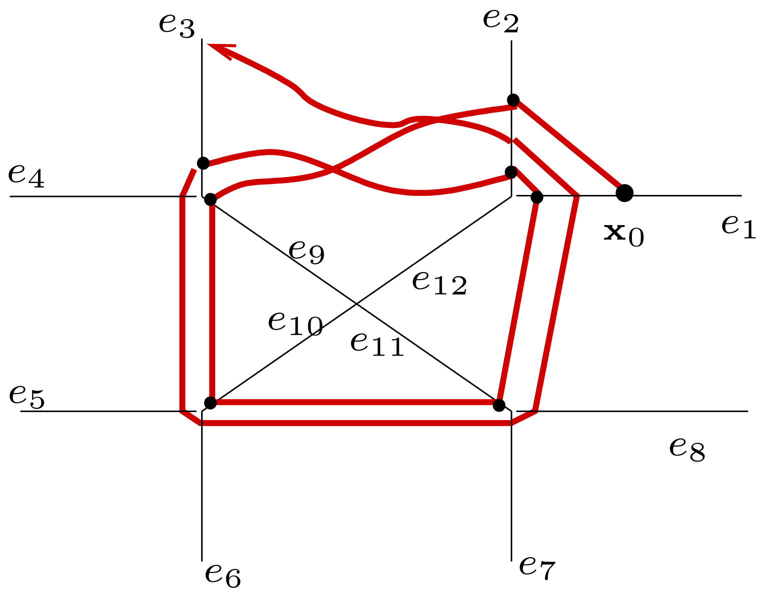
Hybrid Systems

Example: Swimmer in a whirlpool



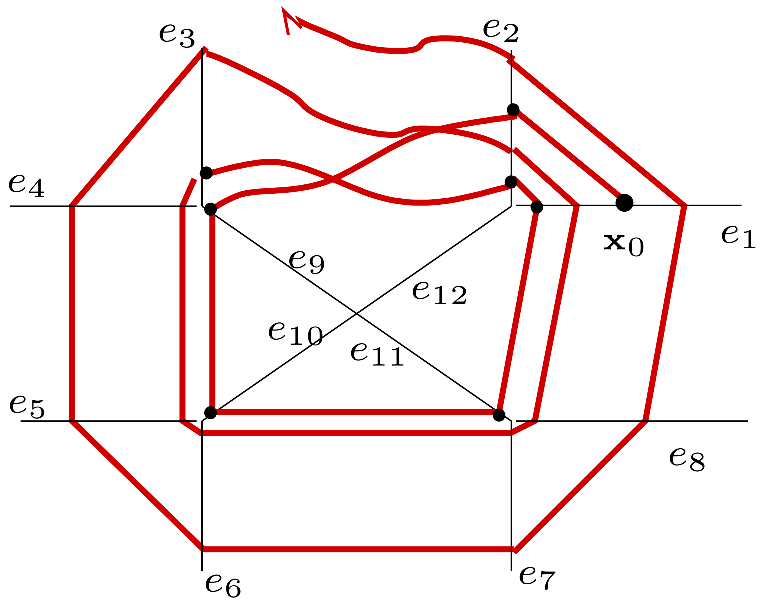
Hybrid Systems

Example: Swimmer in a whirlpool



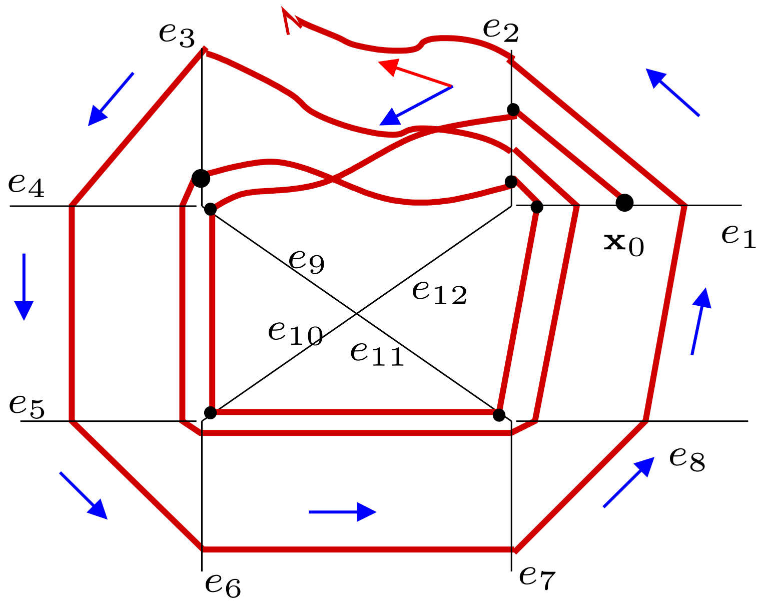
Hybrid Systems

Example: Swimmer in a whirlpool



Hybrid Systems

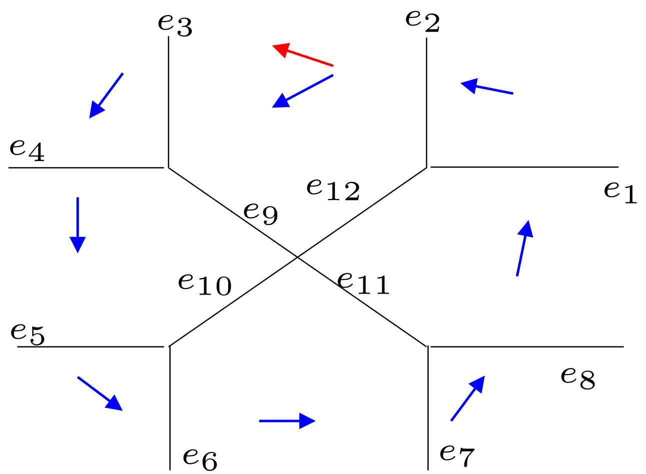
Example: Swimmer in a whirlpool



Polygonal Differential Inclusion Systems (SPDIs)

- A partition of the plane into convex polygonal regions
- A constant differential inclusion for each region

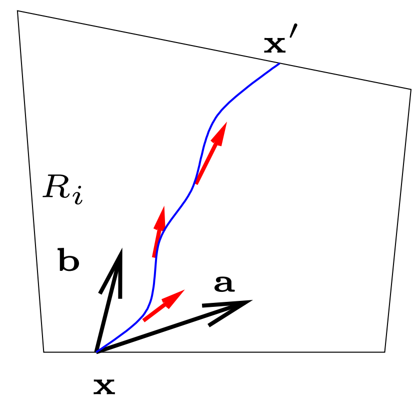
$$\dot{x} \in \angle_{\mathbf{a}}^{\mathbf{b}} \text{ if } \mathbf{x} \in R_i$$

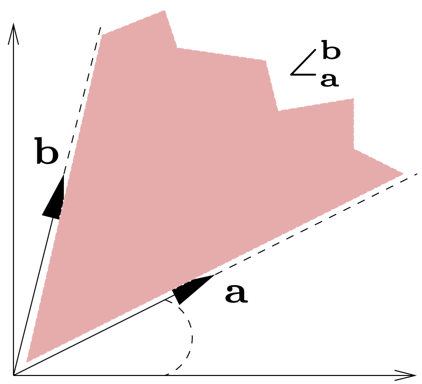


Polygonal Differential Inclusion Systems (SPDIs)

- A partition of the plane into convex polygonal regions
- A constant differential inclusion for each region

$$\dot{x} \in \angle_{\mathbf{a}}^{\mathbf{b}} \text{ if } \mathbf{x} \in R_i$$



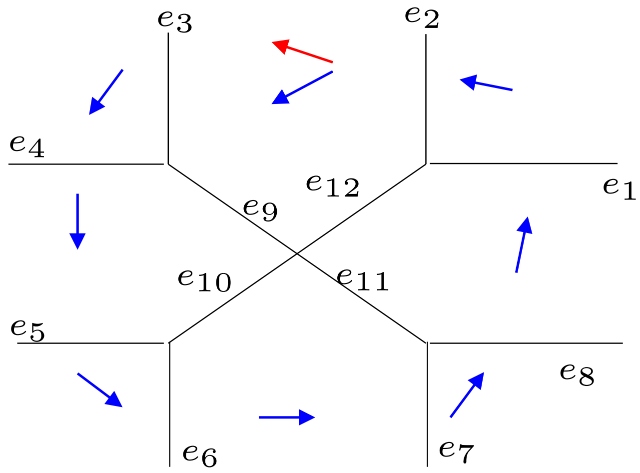


Polygonal Differential Inclusion Systems (SPDIs)

- The "swimmer" is a hybrid system
- Hybrid Automata?

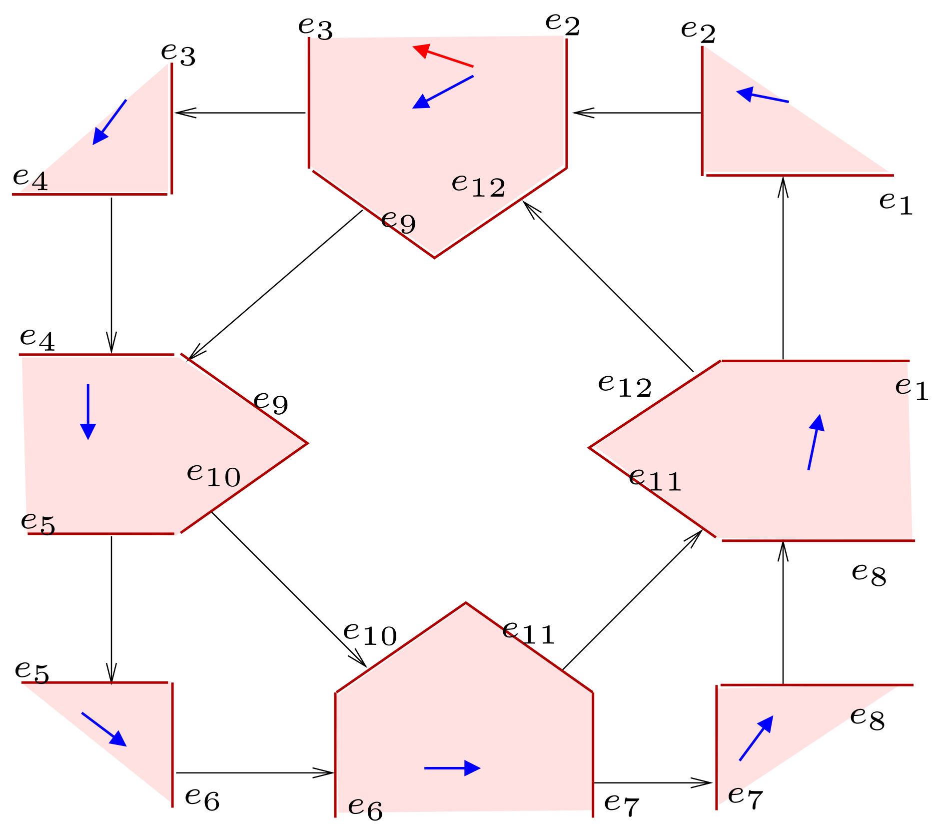
Polygonal Differential Inclusion Systems (SPDIs)

- The “swimmer” is a hybrid system
- Hybrid Automata?



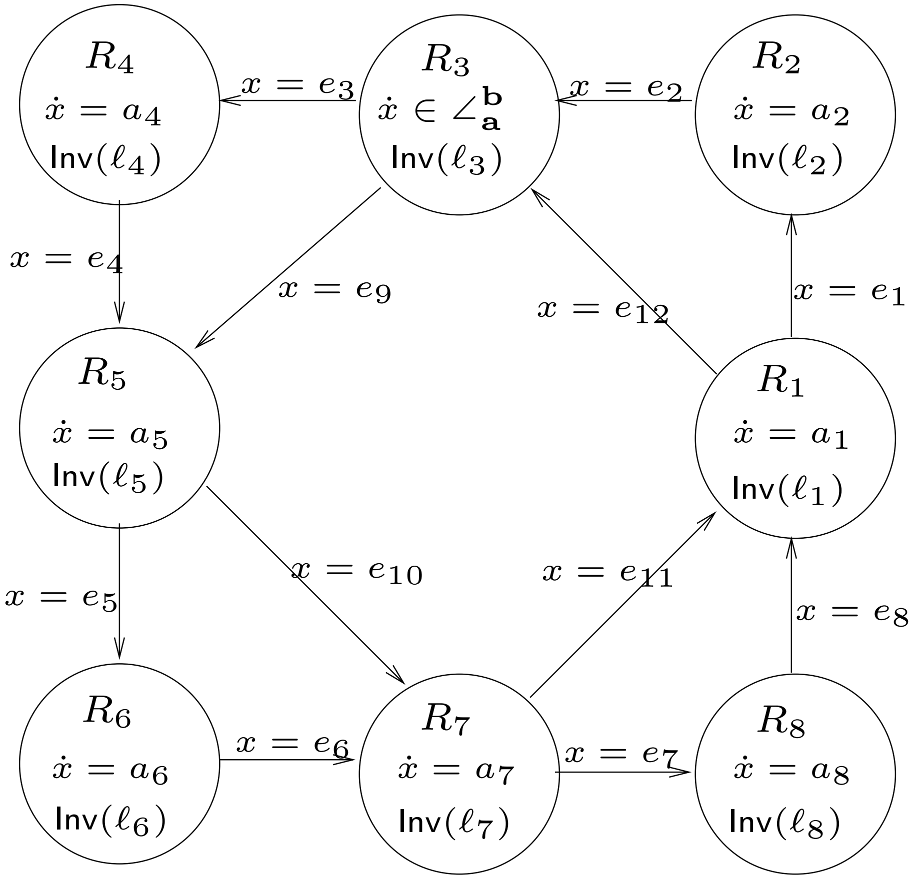
Polygonal Differential Inclusion Systems (SPDIs)

- The “swimmer” is a hybrid system
- Hybrid Automata?



Polygonal Differential Inclusion Systems (SPDIs)

- The "swimmer" is a hybrid system
- Hybrid Automata?



Polygonal Differential Inclusion Systems (SPDIs)

- The “swimmer” is a hybrid system
- Hybrid Automata?

We will use the “geometric” representation instead of the hybrid automata

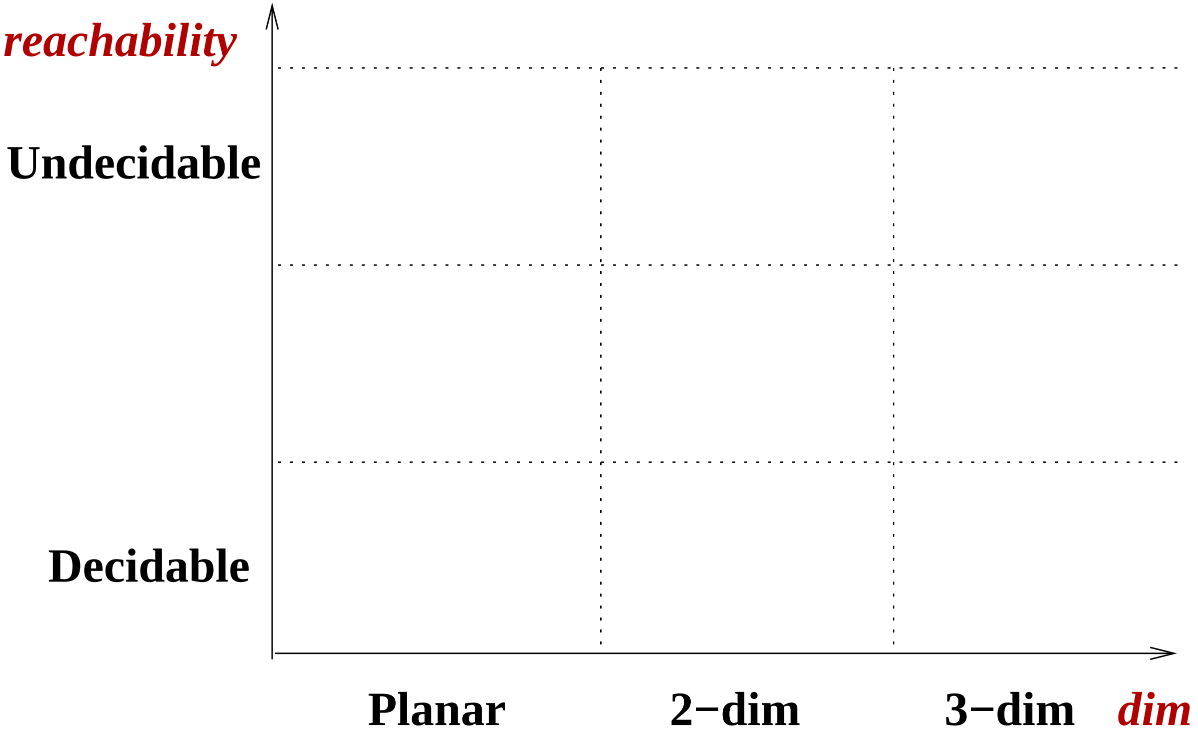
Overview of the presentation

- Motivation and Contributions
- Algorithm for Reachability Problem (SPDIs)
- Implementation – SPeeDI
- Algorithm for Phase Portrait construction (SPDIs)
- Other 2 dim Hybrid Systems
  - Between Decidability and Undecidability
  - Undecidability results
- Summary of Results and Perspectives

Motivation and Contributions

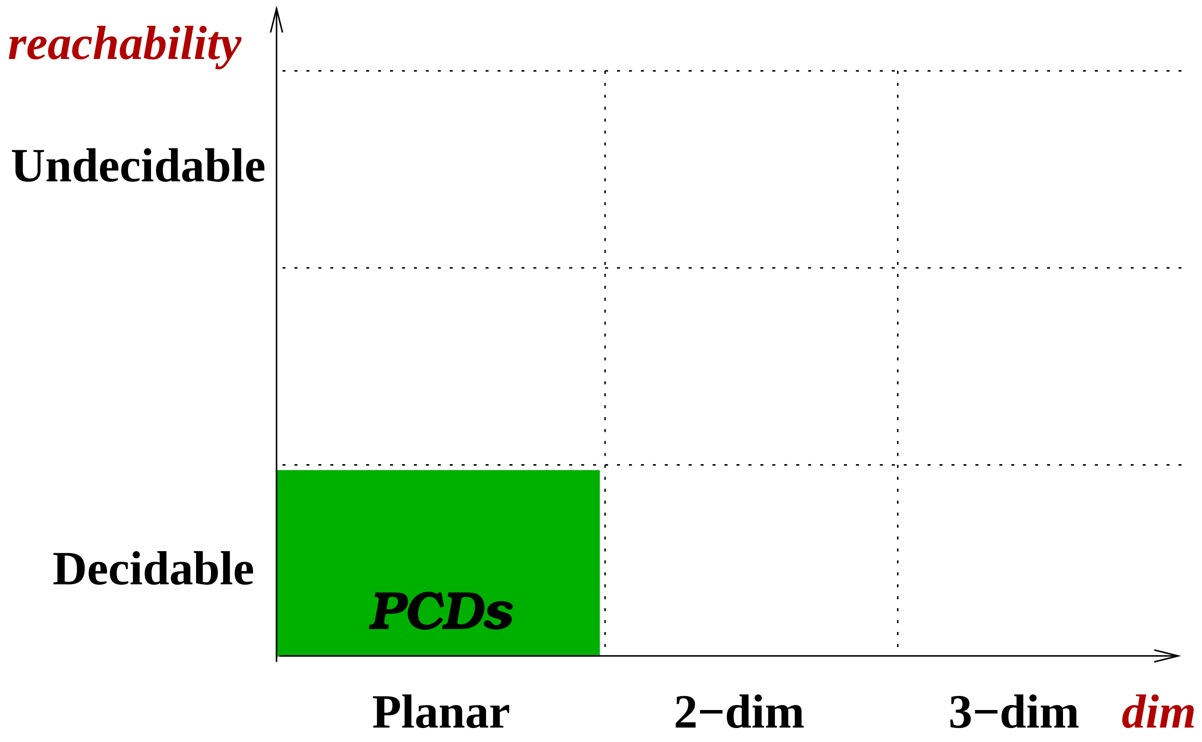
Motivation and Contributions

Scientific Context (Reachability)



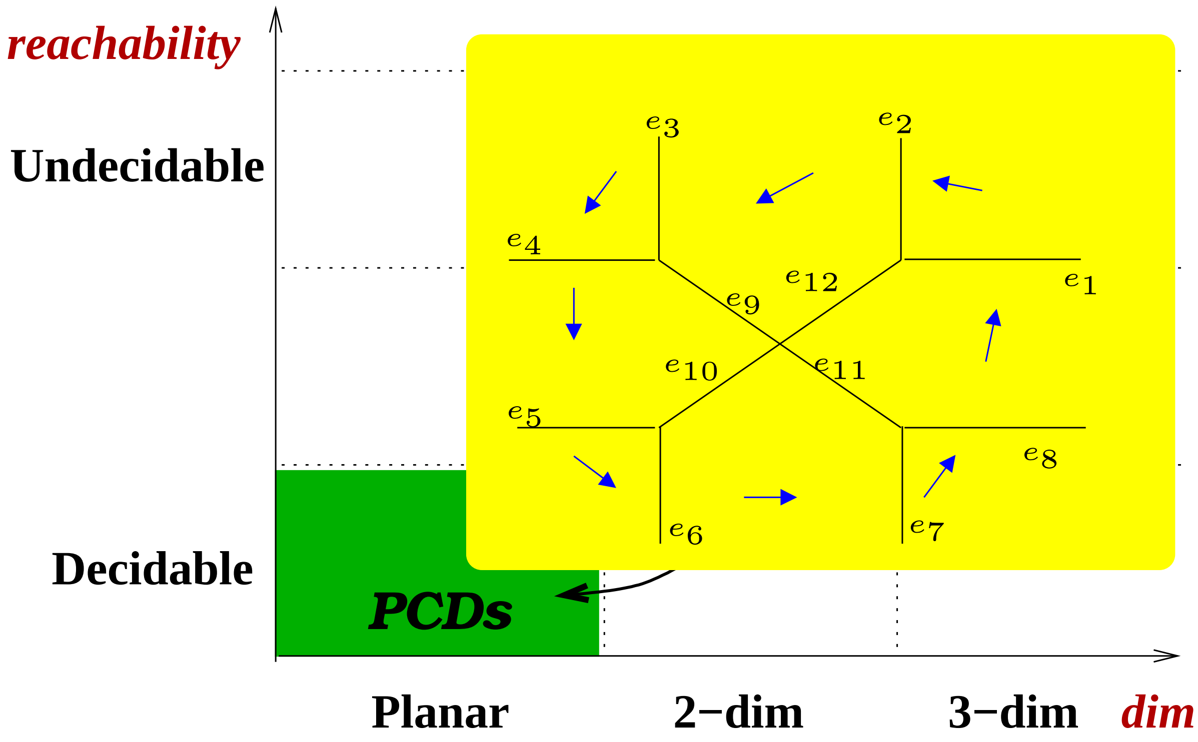
Motivation and Contributions

Scientific Context (Reachability)



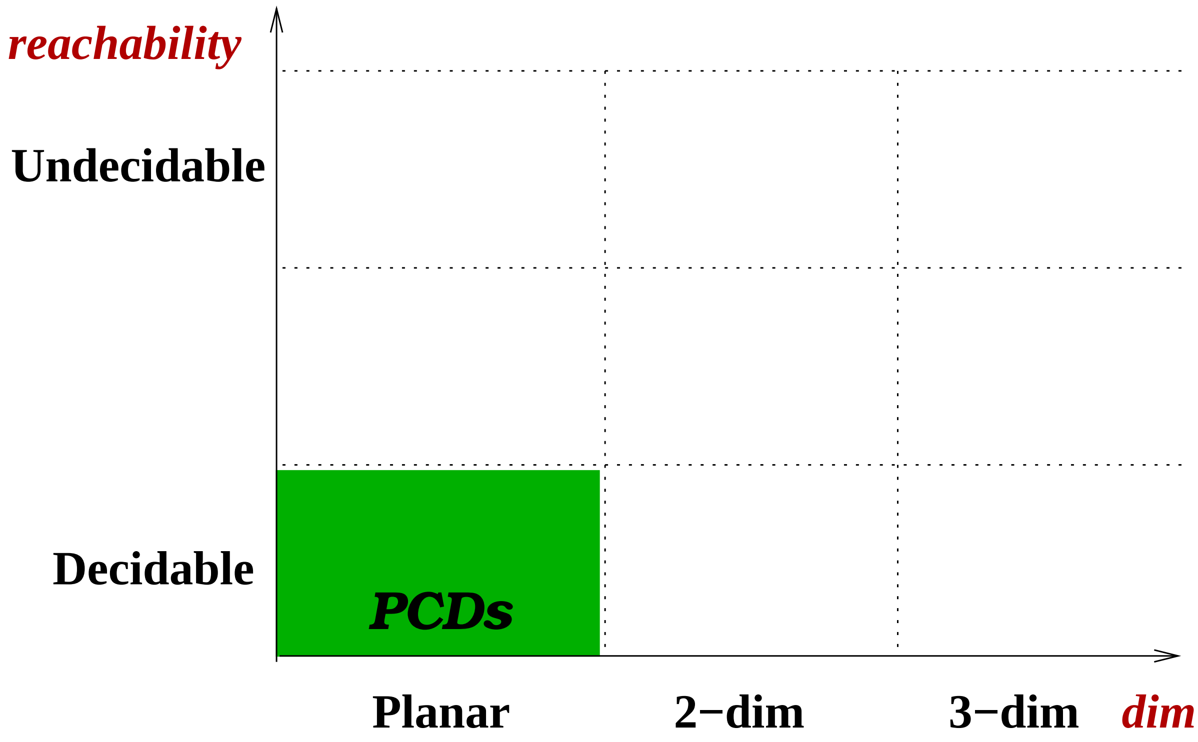
Motivation and Contributions

Scientific Context (Reachability)



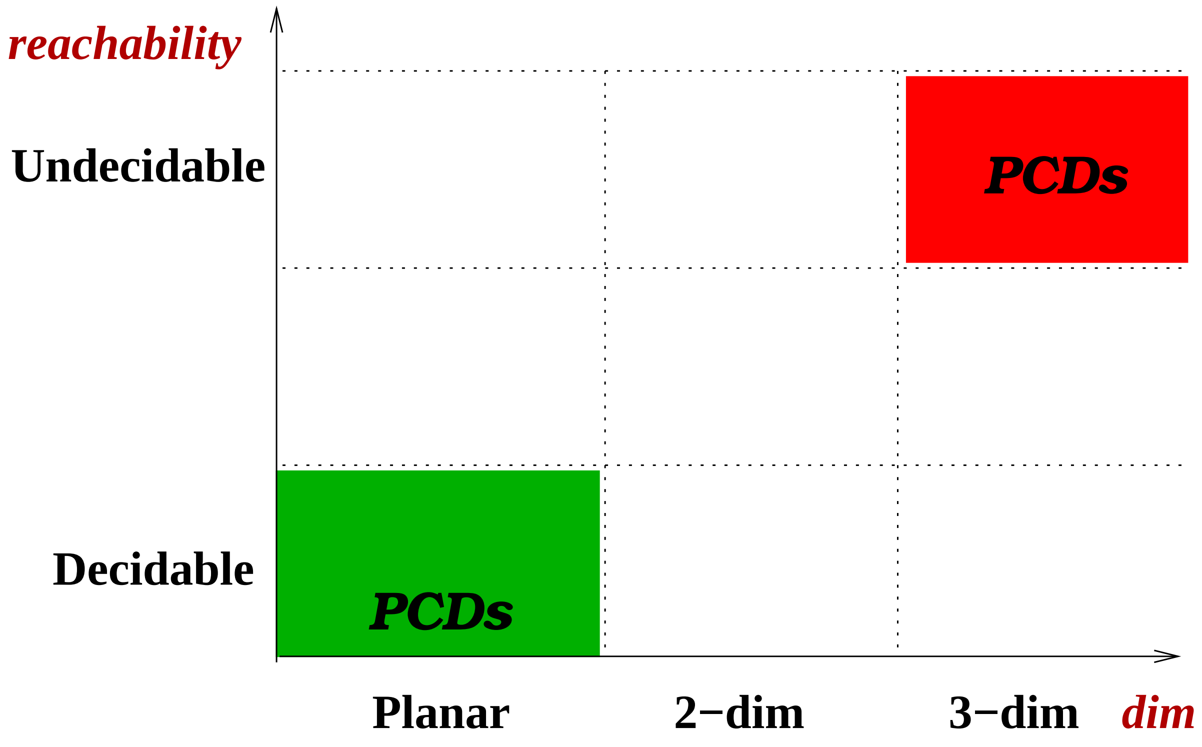
Motivation and Contributions

Scientific Context (Reachability)



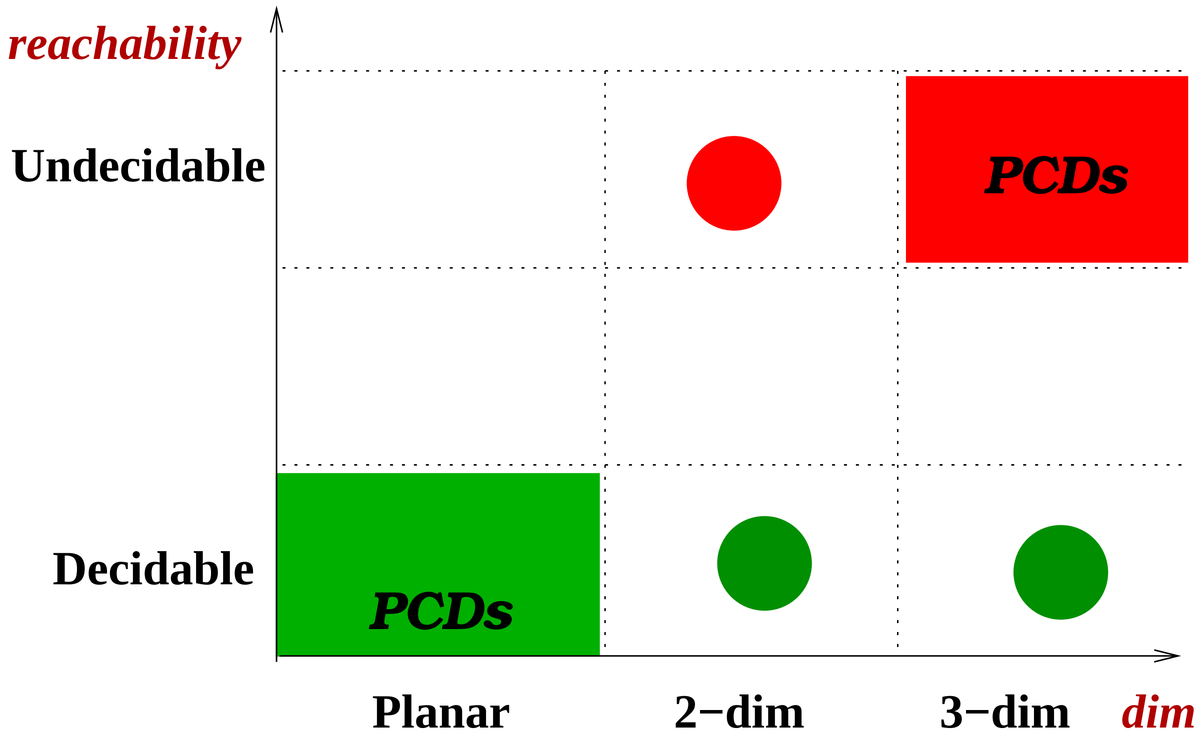
Motivation and Contributions

Scientific Context (Reachability)



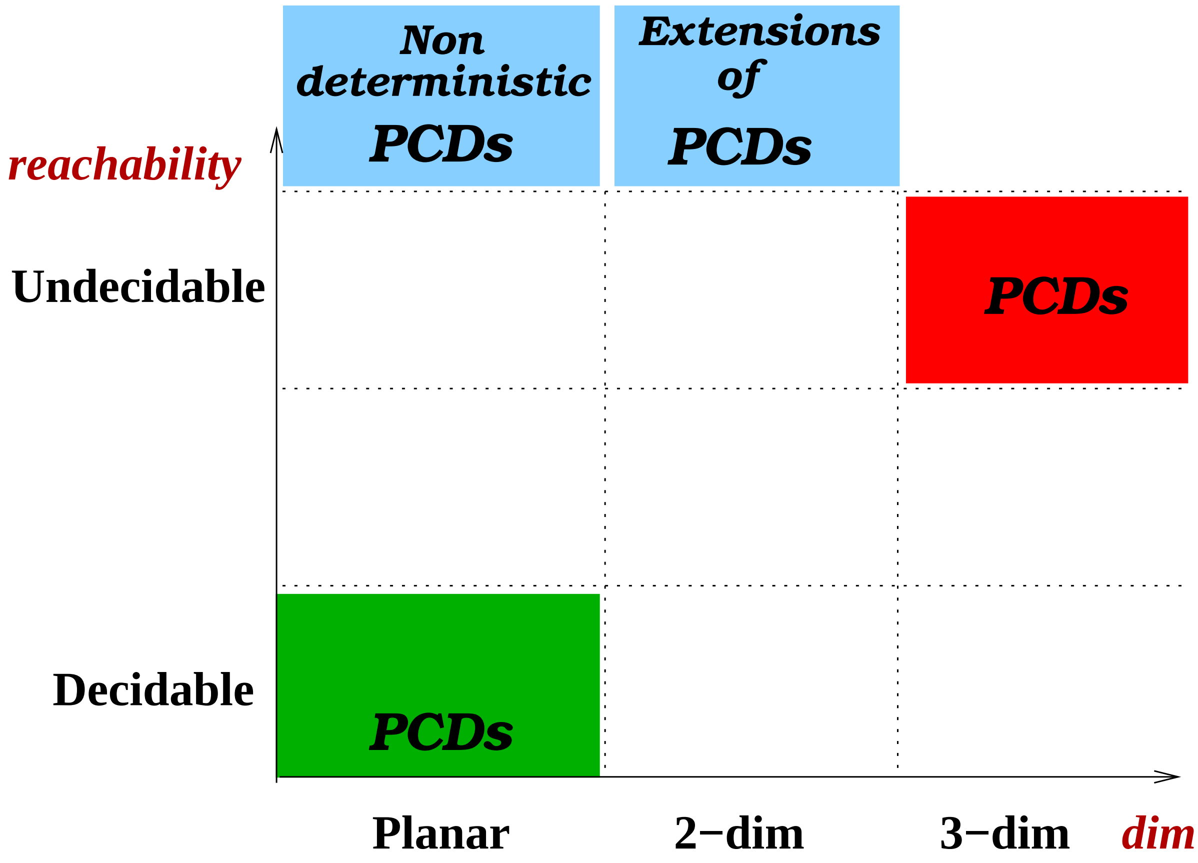
Motivation and Contributions

Scientific Context (Reachability)



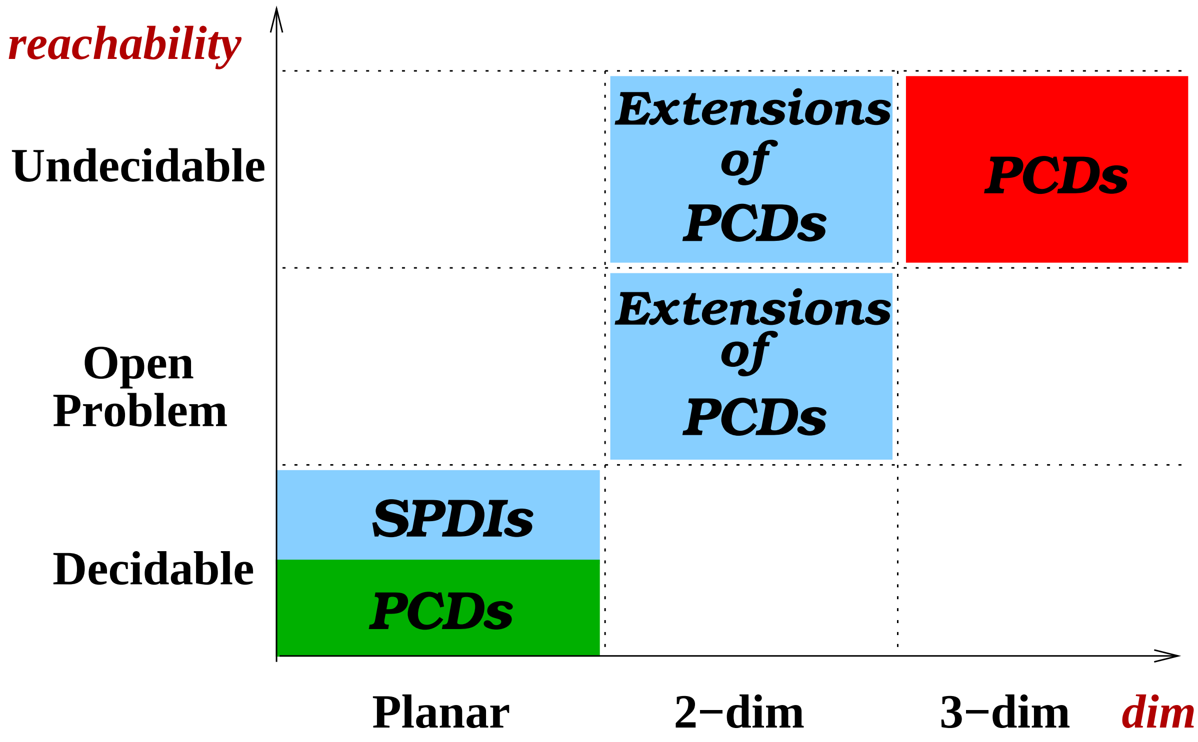
Motivation and Contributions

Challenge



Motivation and Contributions

Contributions (Reachability)



Motivation and Contributions

Scientific Context (Phase Portrait)

- Phase Portrait for PCDs
- Numerical algorithms for computing Viability Kernels

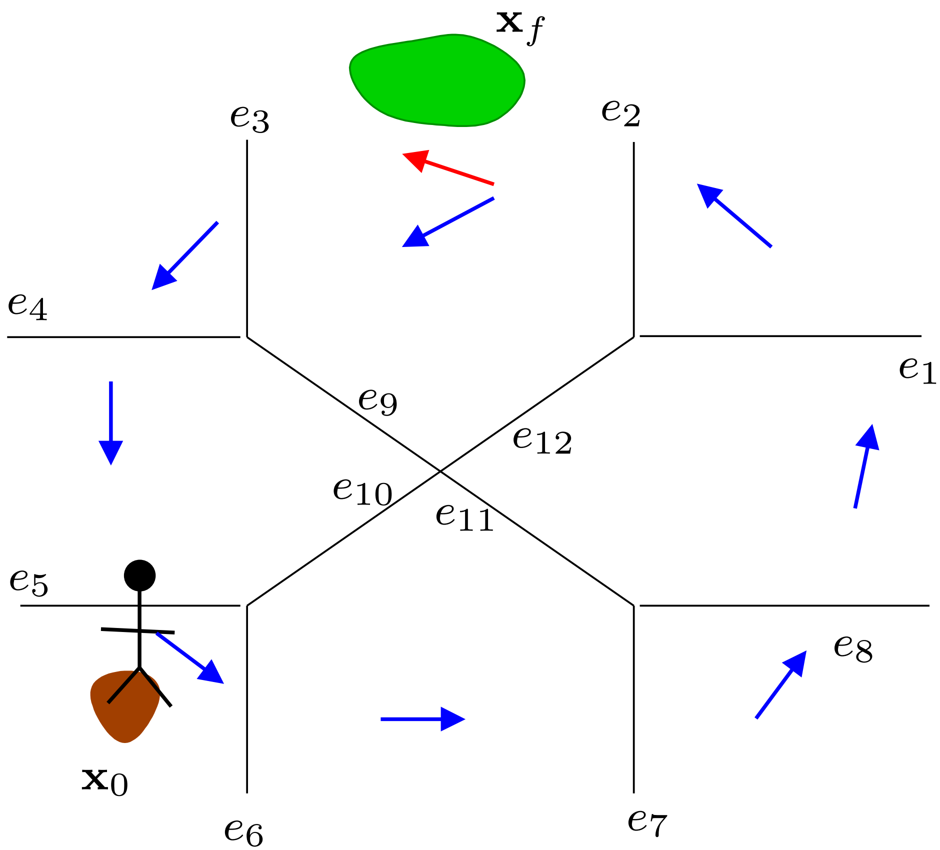
Motivation and Contributions

Contributions (Phase Portrait)

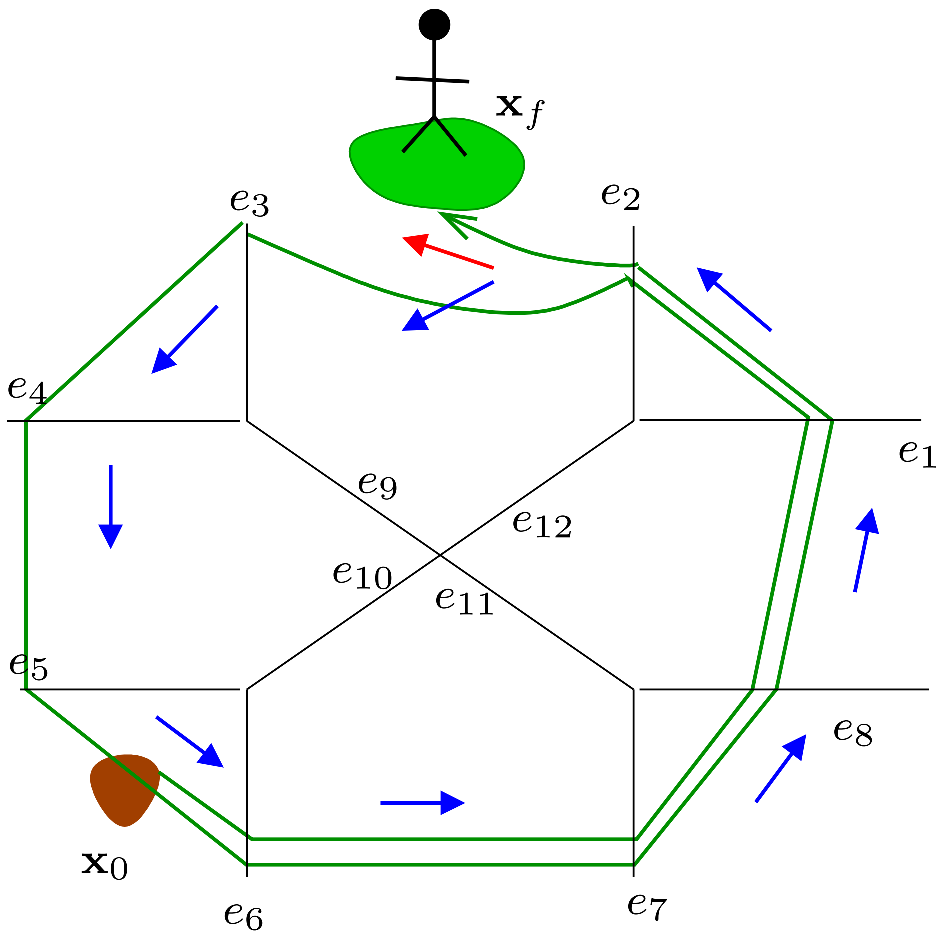
- Phase Portrait for SPDIs
  - Viability Kernel
  - Controllability Kernel

Reachability Analysis for SPDIs

The Reachability Problem for SPDIs



The Reachability Problem for SPDIs

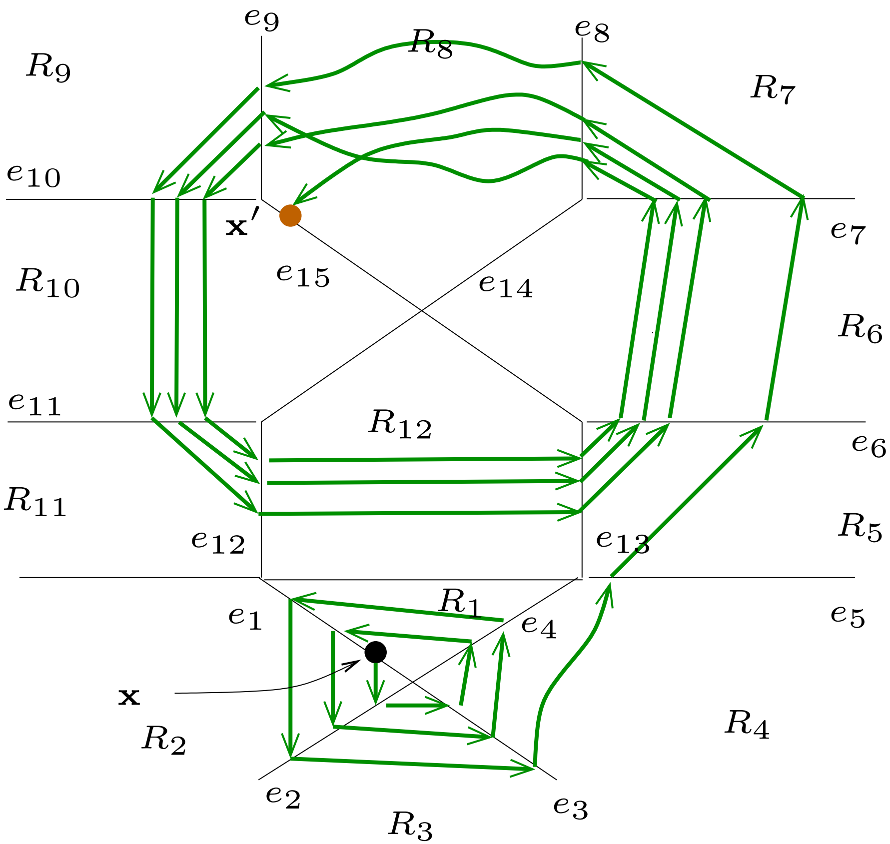


Reachability problem: Is there a trajectory from $\mathbf{x}_0$ to $\mathbf{x}_f$?

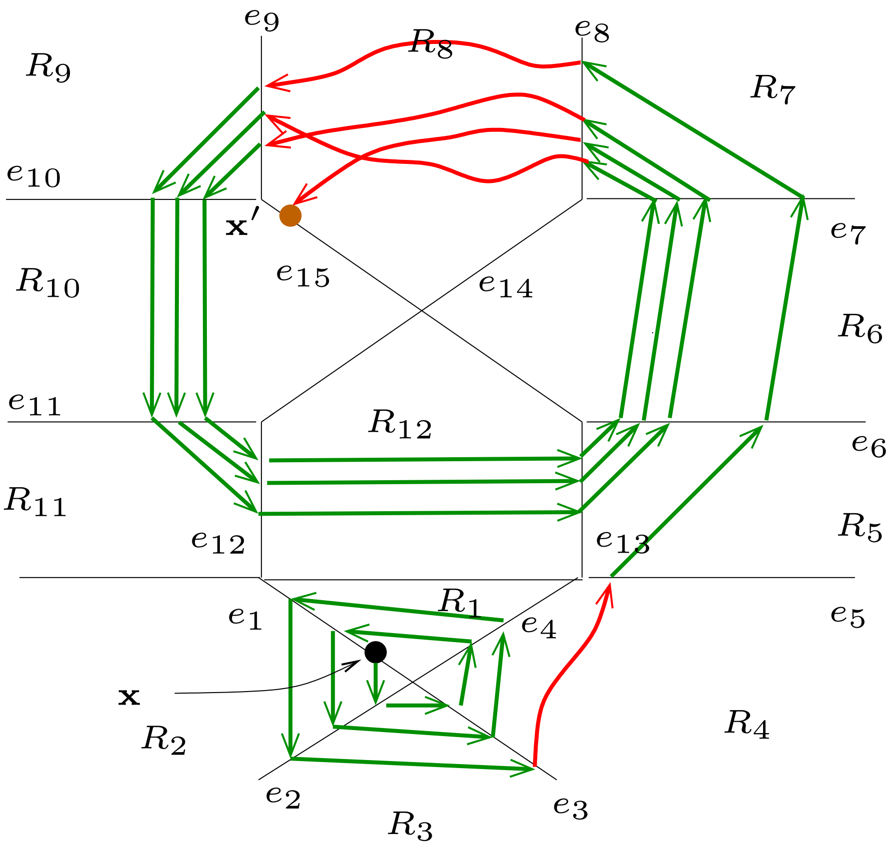
Solving the Reachability Problem

1. From trajectories to *simplified trajectories*

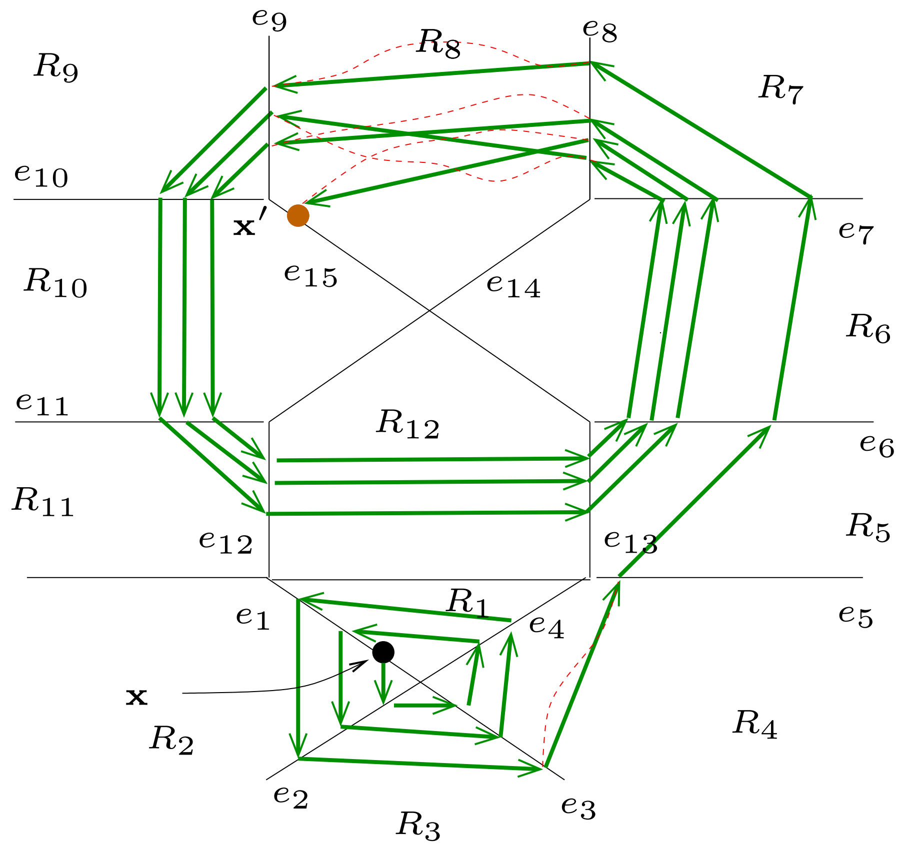
1. Simplification of trajectories



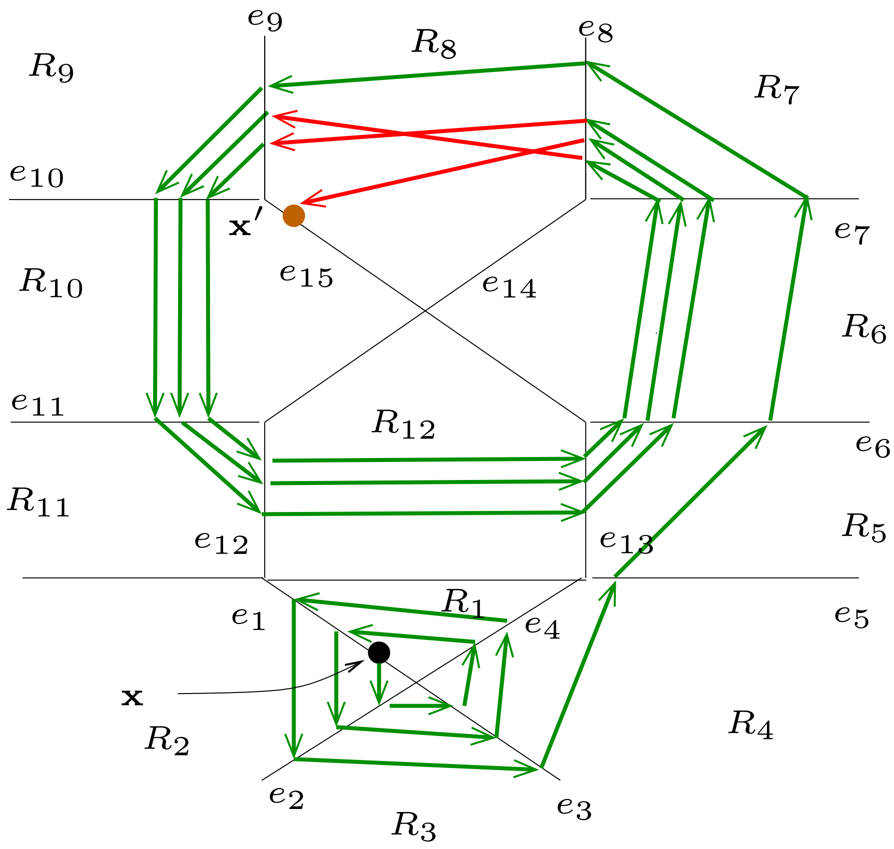
1. Simplification of trajectories



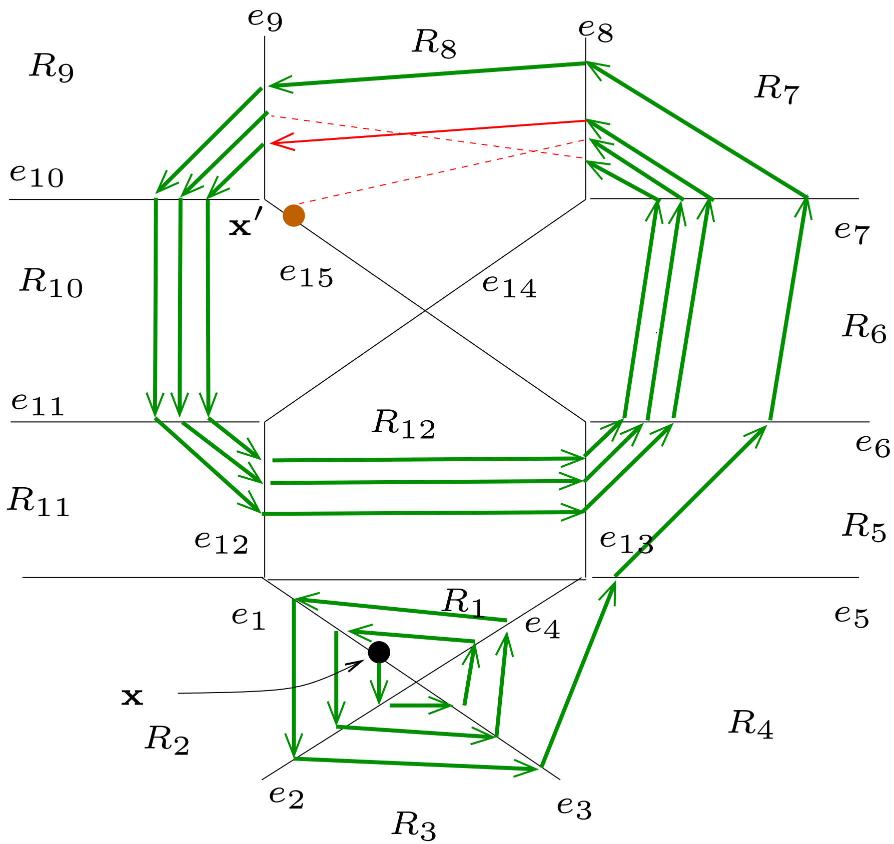
1. Simplification of trajectories



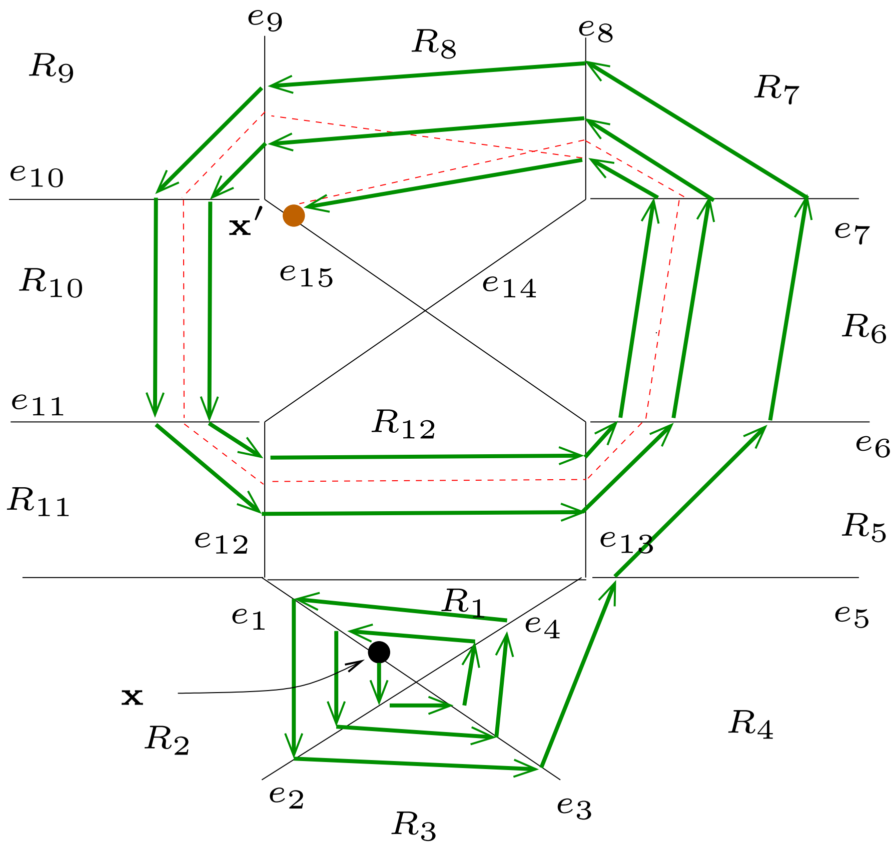
1. Simplification of trajectories



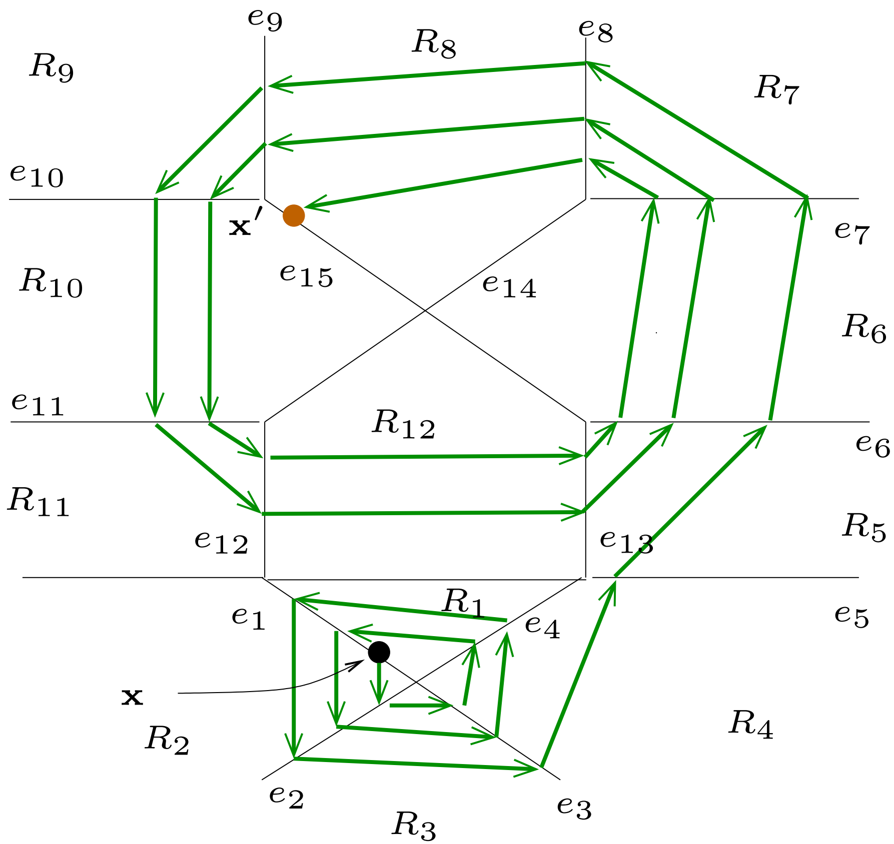
1. Simplification of trajectories



1. Simplification of trajectories



1. Simplification of trajectories

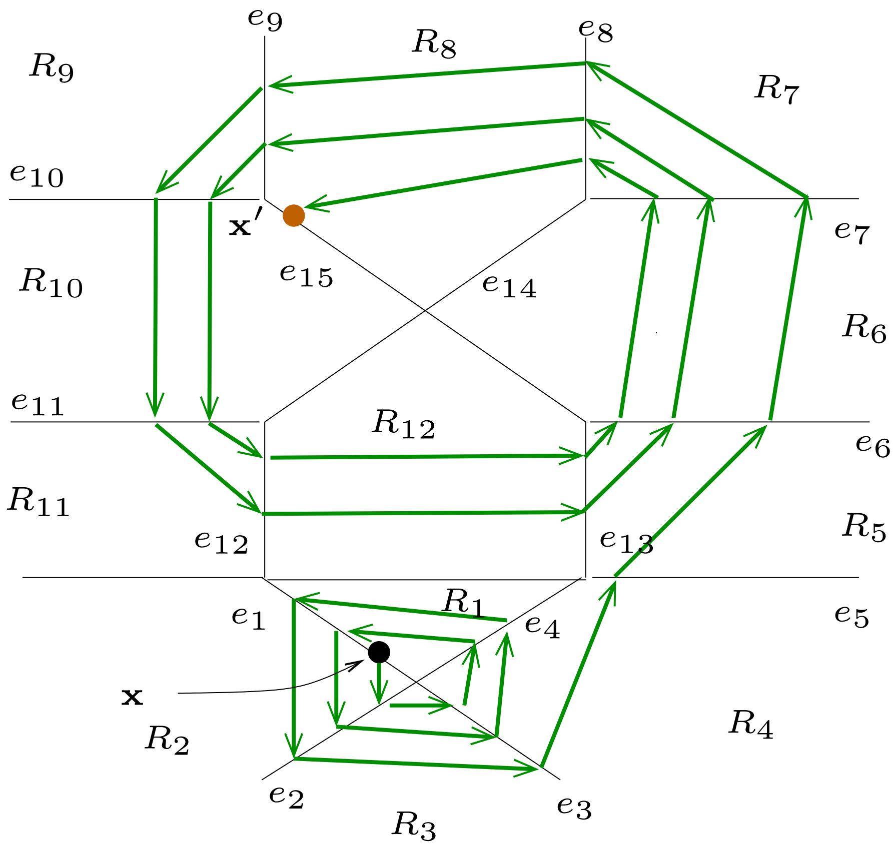


Theorem: If there is an arbitrary trajectory between two points then it always exists a straightened non–crossing trajectory between them

Solving the Reachability Problem

1. From trajectories to *simplified trajectories*
2. From simplified trajectories to *signatures*

2. Abstraction into *signatures*



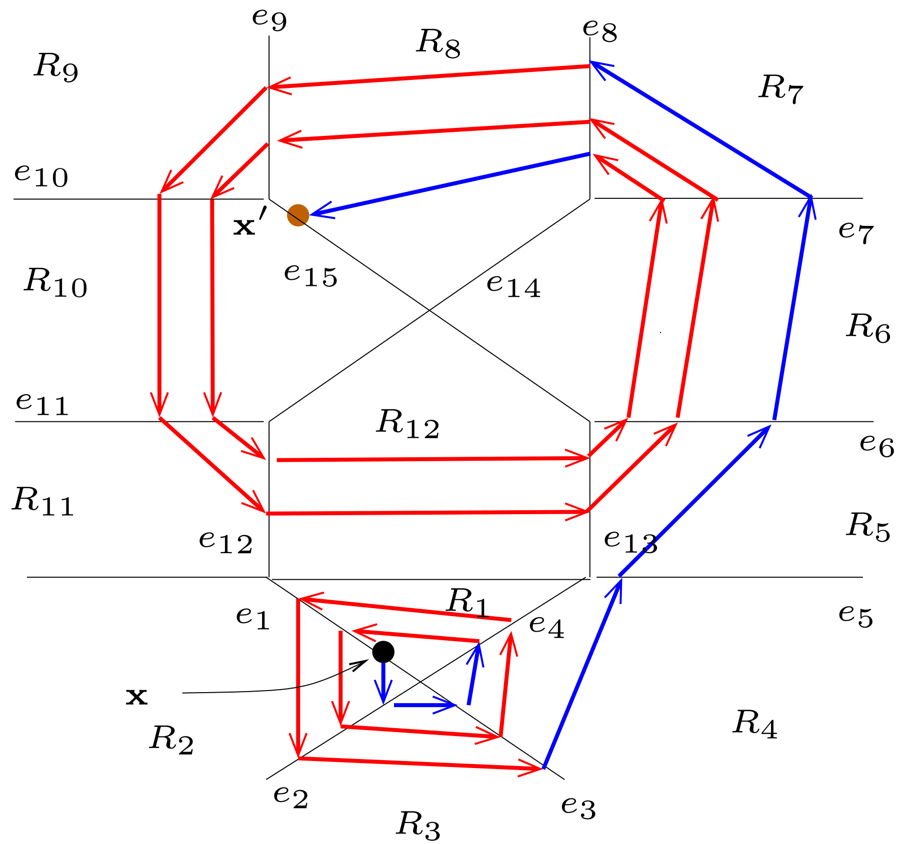
$$\sigma = e_1 e_2 e_3 \ldots e_5 e_6 e_7 \ldots e_{13} e_6 e_7 e_8 e_{15}$$

Solving the Reachability Problem

1. From trajectories to *simplified trajectories*
2. From simplified trajectories to *signatures*
3. From signatures to *factorized signatures*

3. Factorization of Signatures

For $\sigma = e_1 e_2 e_3 \ldots e_5 e_6 e_7 \ldots e_{13} e_6 e_7 e_8 e_{15}$



We obtain the representation:

$\sigma = e_1 e_2 e_3 \left(e_4 e_1 e_2 e_3\right)^2 e_5 e_6 e_7 e_8 \left(e_9 \cdots e_{13} e_6 e_7 e_8\right)^2 e_{15}$

Verimag

3. Canonical Factorization of Signatures

Representation Theorem: Any edge signature $\sigma = e_1, e_2, \ldots, e_n$ can be represented as

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

3. Canonical Factorization of Signatures

Representation Theorem: Any edge signature $\sigma = e_1, e_2, \ldots, e_n$ can be represented as

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

- Properties:
  - r_i is a seq. of pairwise different edges;
  - s_i is a simple cycle;
  - r_i and r_j are disjoint
  - s_i and s_j are different

Proof based on topological properties of the plane

Solving the Reachability Problem

1. From trajectories to *simplified trajectories*
2. From simplified trajectories to *signatures*
3. From signatures to *factorized signatures*
4. From factorized signatures to *types of signatures*

4. Types of Signatures

Abstraction: Any edge signature

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

belongs to a type

$$type(\sigma) = r_1, s_1, r_2, s_2, \ldots r_n, s_n, r_{n+1}$$

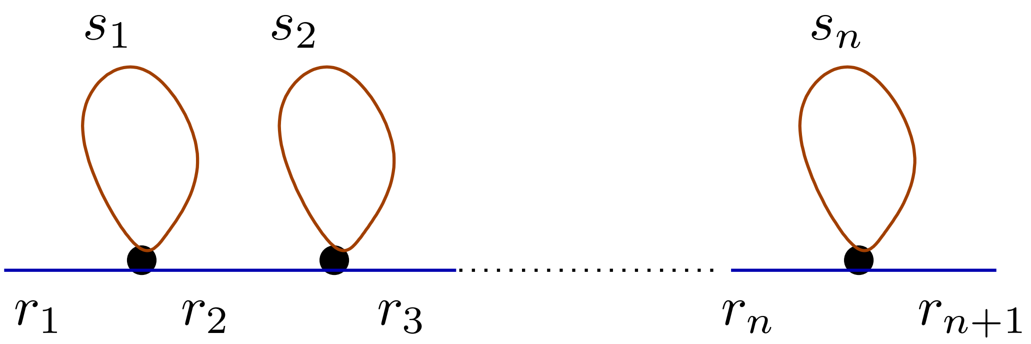
4. Types of Signatures

Abstraction: Any edge signature

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

belongs to a type

$$type(\sigma) = r_1, s_1, r_2, s_2, \ldots r_n, s_n, r_{n+1}$$



4. Types of Signatures

Abstraction: Any edge signature

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

belongs to a type

$$type(\sigma) = r_1, s_1, r_2, s_2, \ldots r_n, s_n, r_{n+1}$$

In the previous example:

$type(\sigma) =$

$$e_1 e_2 e_3,\ e_4 e_1 e_2 e_3,\ e_5 e_6 e_7 e_8,\ e_9 \cdots e_{13} e_6 e_7 e_8,\ e_{15}$$

4. Types of Signatures

Abstraction: Any edge signature

$$\sigma = r_1(s_1)^{k_1} r_2(s_2)^{k_2} \ldots r_n(s_n)^{k_n} r_{n+1}$$

belongs to a type

$$type(\sigma) = r_1, s_1, r_2, s_2, \ldots r_n, s_n, r_{n+1}$$

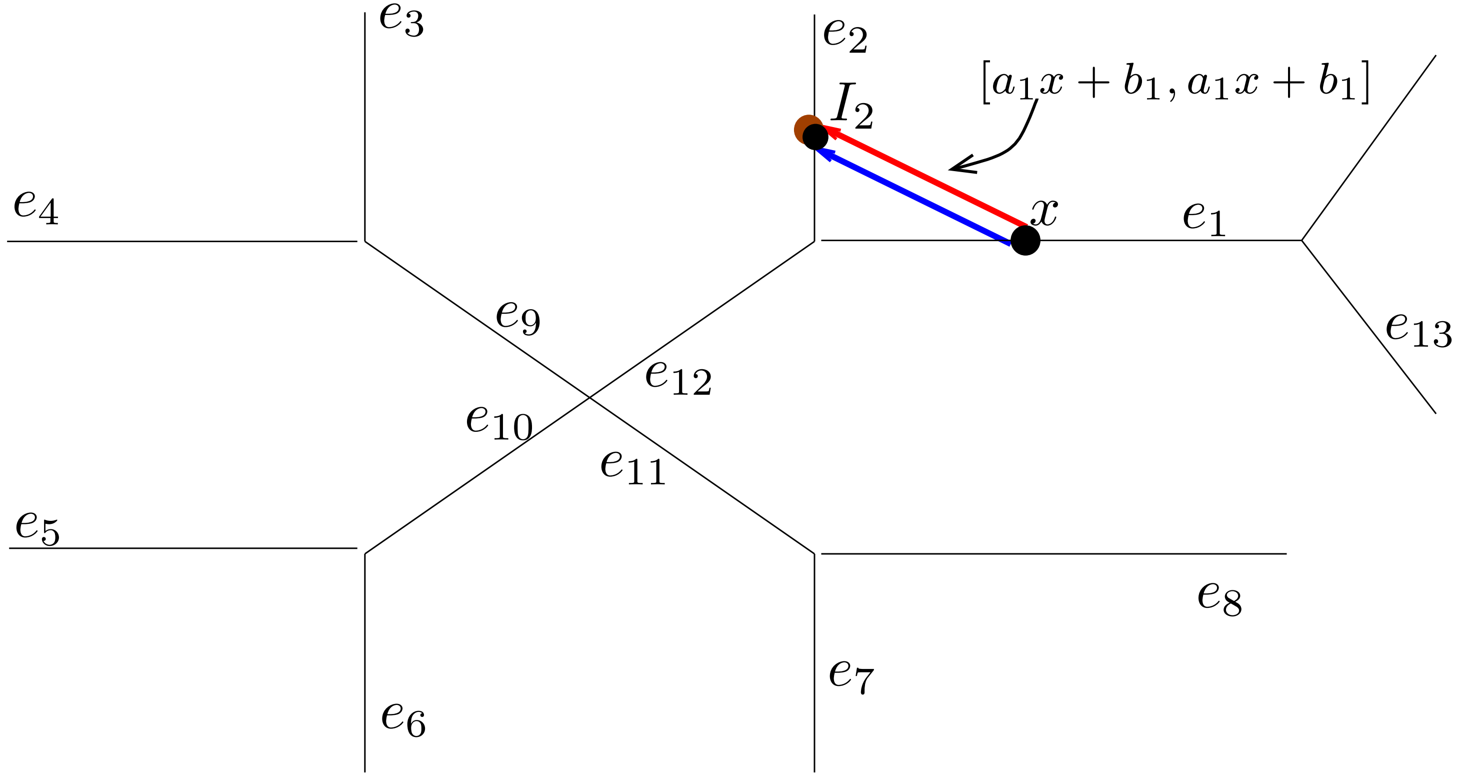
Prop. The set of types of signatures is finite

Solving the Reachability Problem

1. From trajectories to *simplified trajectories*
2. From simplified trajectories to *signatures*
3. From signatures to *factorized signatures*
4. From factorized signatures to *types of signatures*
5. Analysis of each type of signature (computing successors)

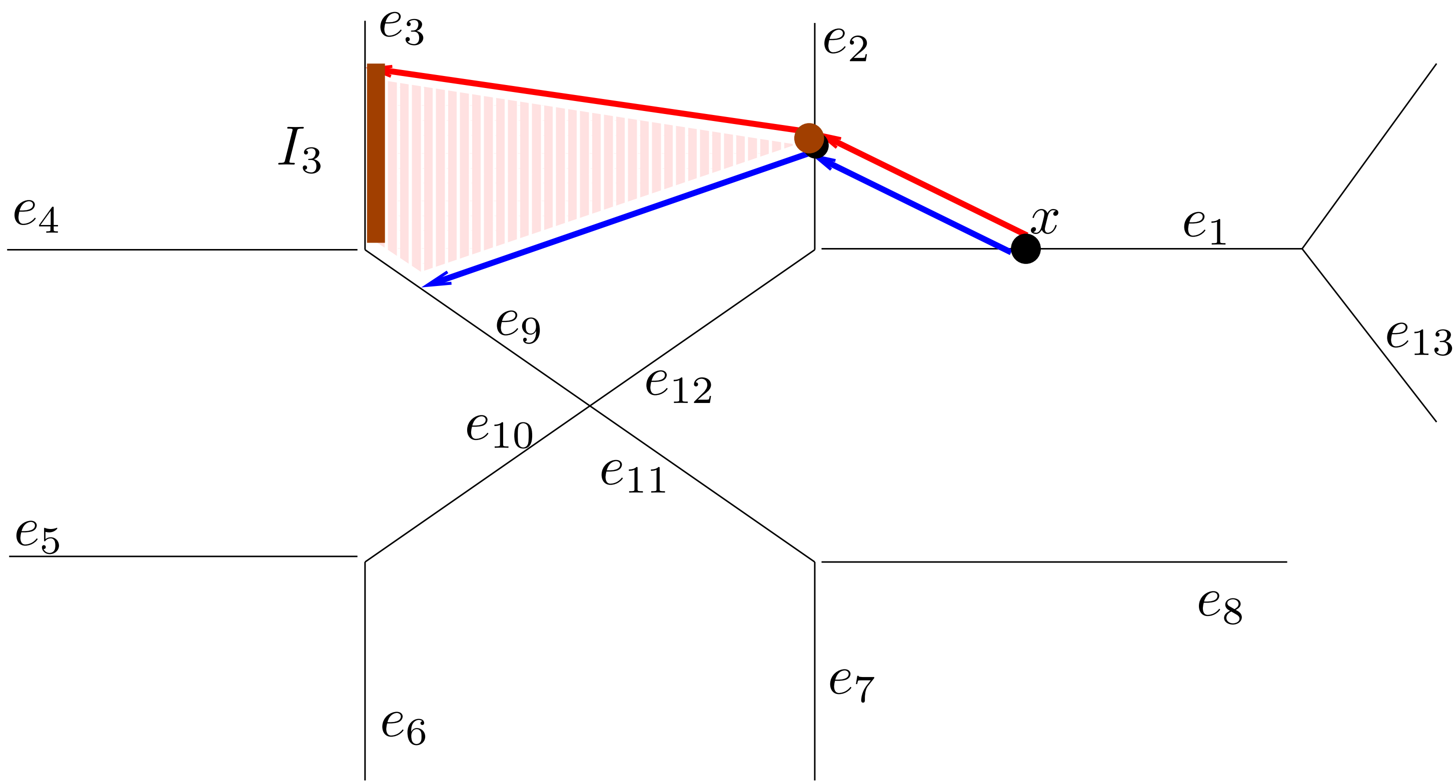
Computing Successors (for σ)

One step ($\sigma = e_1 e_2$)



Computing Successors (for σ)

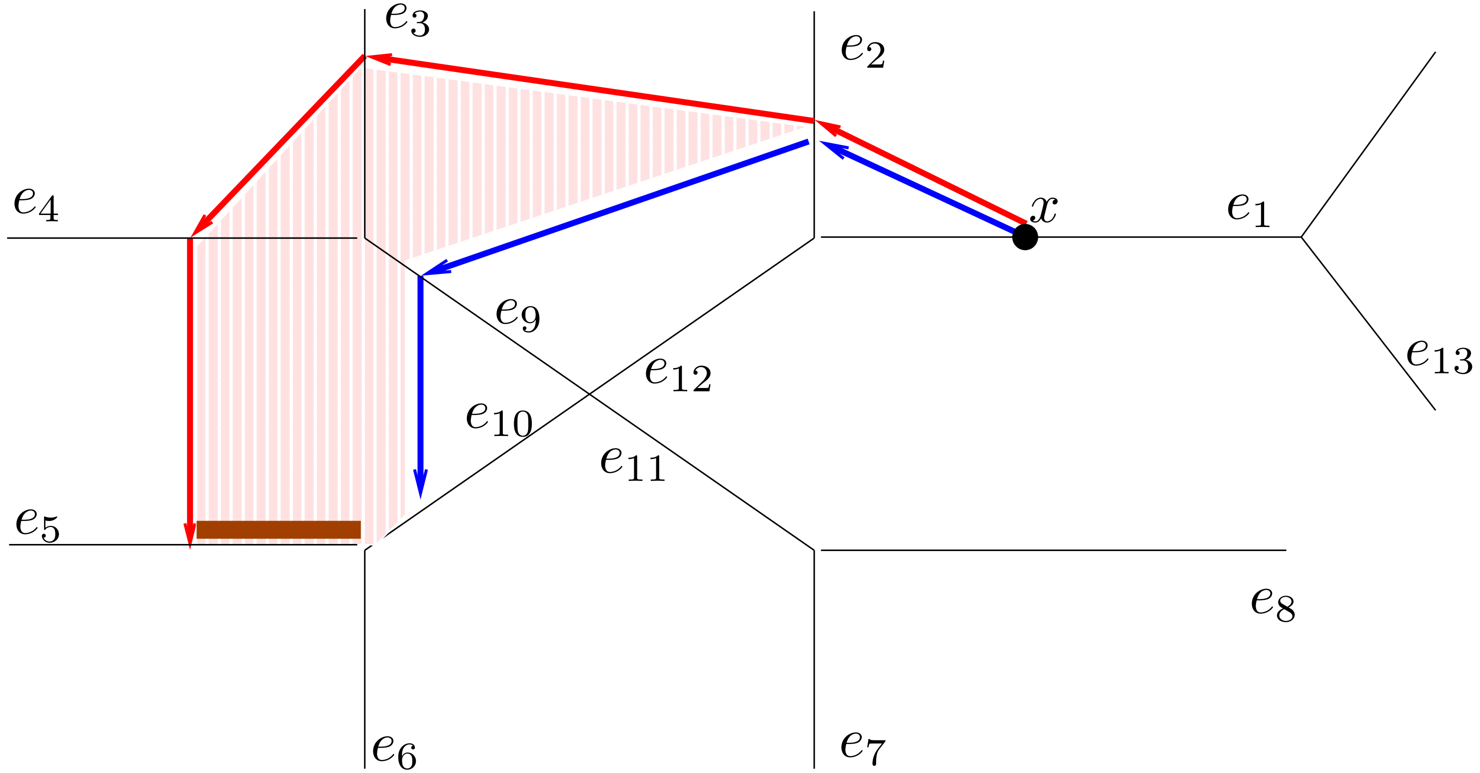
Several steps ($\sigma = e_1 e_2 e_3$)



$$I_3 = \mathsf{Succ}_\sigma(x) = [a_2 x + b_2, a_2' x + b_2'] \cap e_3$$

Computing Successors (for σ)

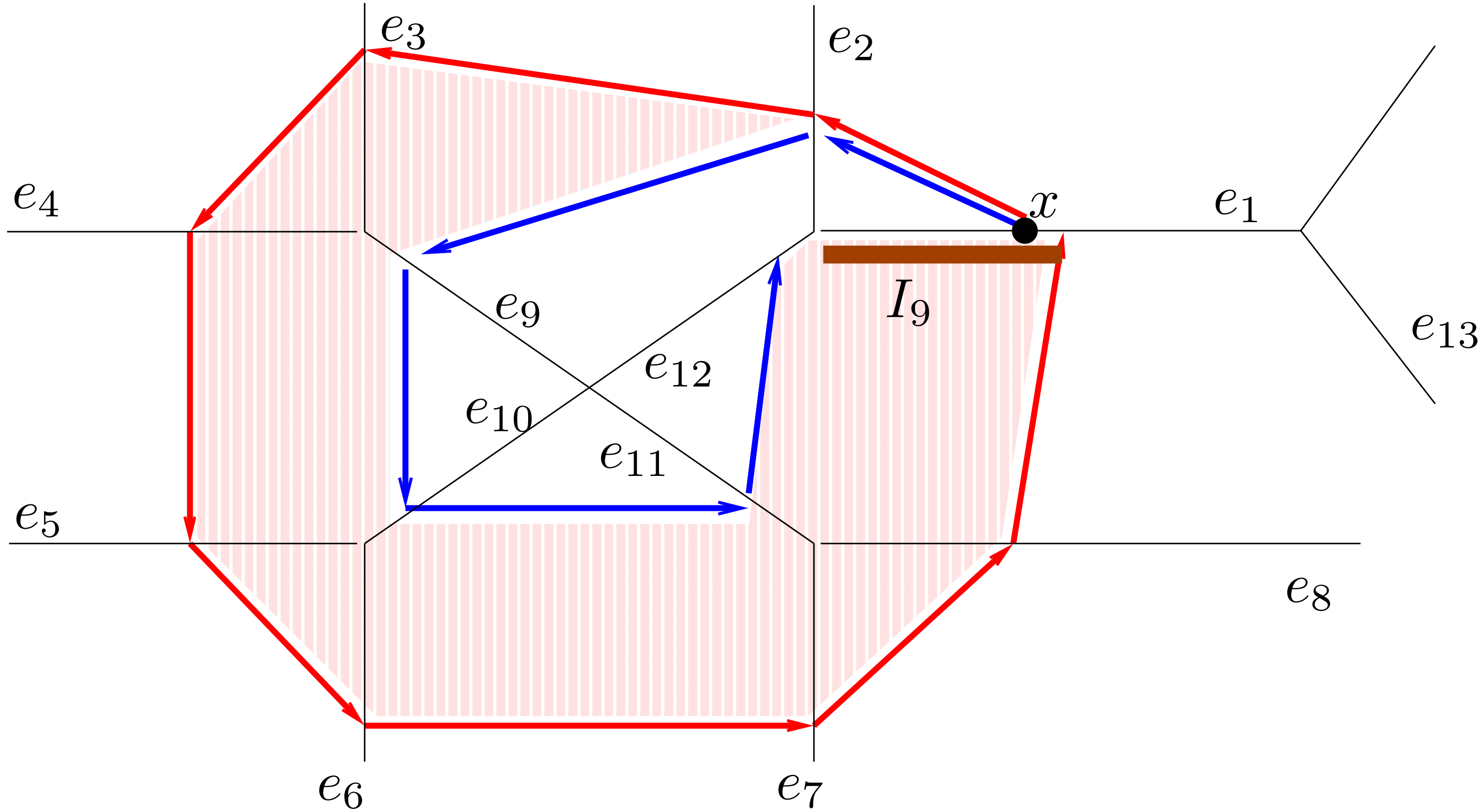
Several steps ($\sigma = e_1 e_2 e_3 e_4 e_5$)



$$I_5 = \mathsf{Succ}_\sigma(x) = [a_4 x + b_4, a'_4 x + b'_4] \cap e_5$$

Computing Successors (for σ)

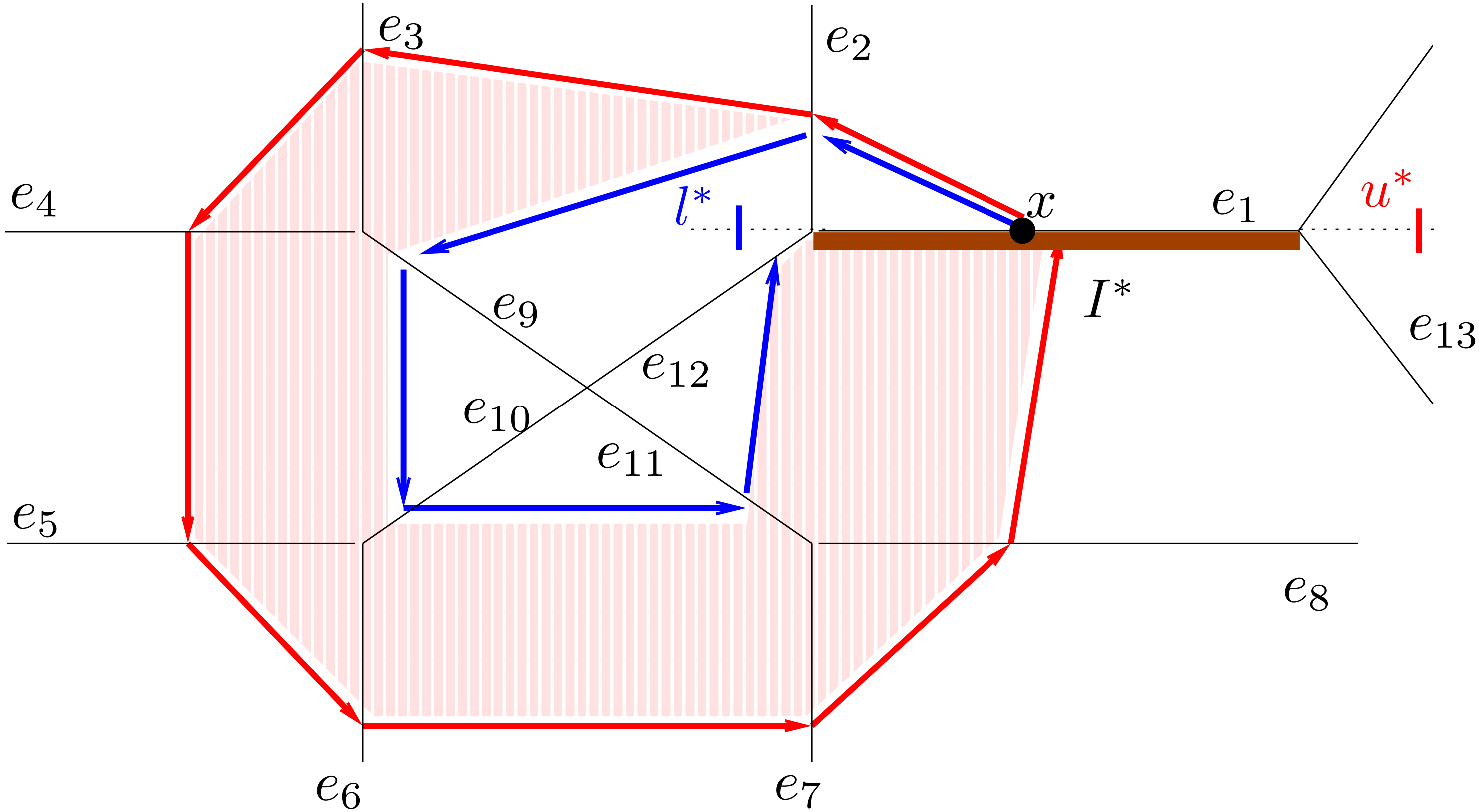
One cycle ($\sigma = s = e_1 e_2 \cdots e_8 e_1$)



$$I_9 = \mathsf{Succ}_\sigma(x) = [a_8 x + b_8, a_8' x + b_8'] \cap e_1$$

Computing Successors (for σ)

One cycle ($\sigma = s = e_1 e_2 \cdots e_8 e_1$)

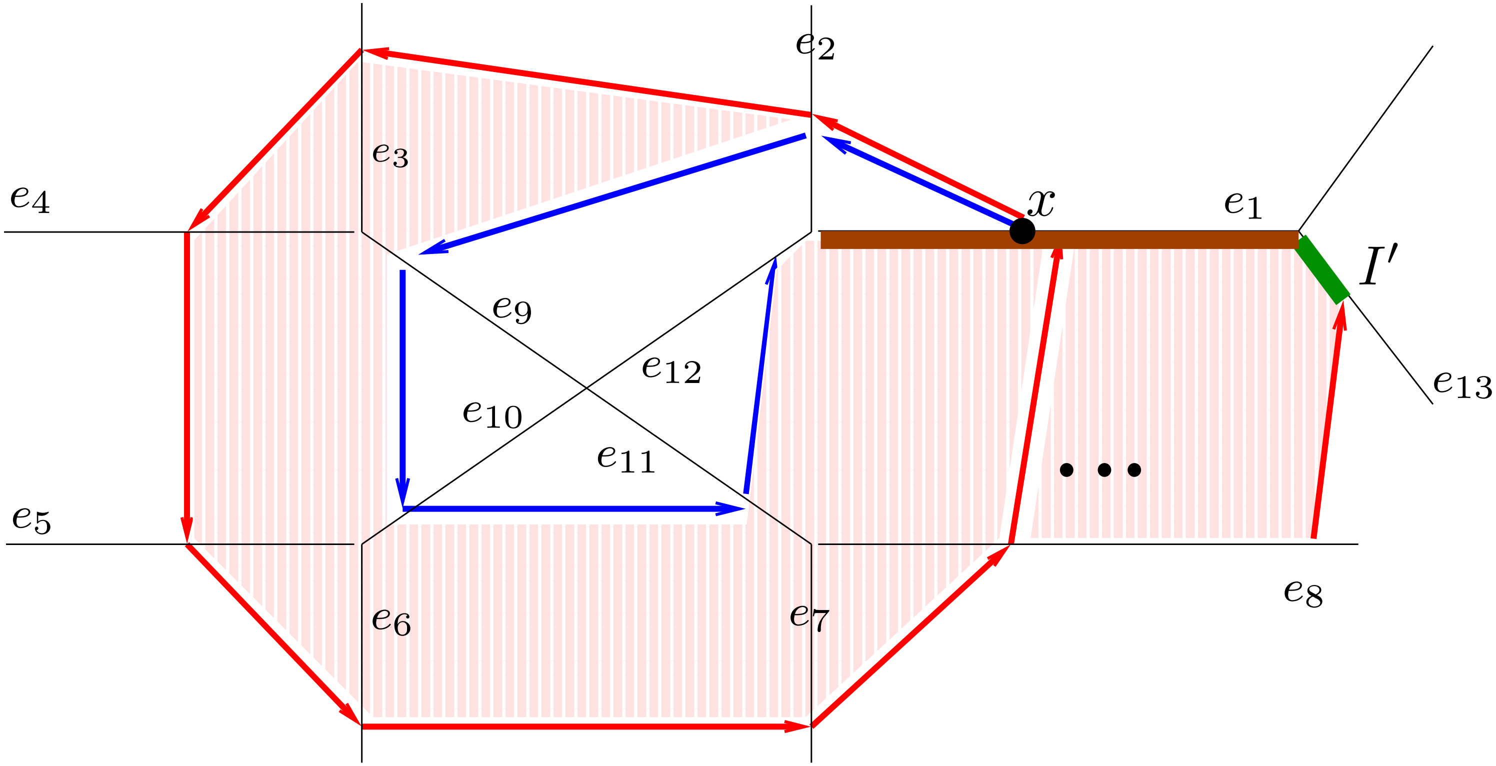


$$l^* = a_1 l^* + b_1 \qquad u^* = a_2 u^* + b_2$$

$$I^* = \mathsf{Succ}^*_\sigma(x) = [l^*, u^*] \cap e_1$$

Computing Successors (for σ)

$\sigma = (s)^* e_{13} \;\; (s = e_1 e_2 \cdots e_8 e_1)$



One cycle iterated: *solution of fixpoint equation (acceleration)*: $I' = \mathsf{Succ}_{e_8 e_{13}} \circ \mathsf{Succ}_{e_1 \cdots e_8} \circ \mathsf{Succ}_s^*(x)$

Computing Successors

Lemma: Successors have the form

$$\mathsf{Succ}_\sigma(l, u) = [a_1 l + b_1, a_2 u + b_2] \cap J \text{ if } [l, u] \subseteq S$$

Lemma: Fixpoint equations

$$[a_1 l^* + b_1, a_2 u^* + b_2] = [l^*, u^*]$$

can be explicitely solved (without iterating).
We have that ($I = [l, u]$):

$$\mathsf{Succ}^*_\sigma(I) = [l^*, u^*] \cap J$$

Reachability Algorithm

for each type of signature τ **do**
check whether $Reach_\tau(x_0, x_f)$

To test whether $Reach_\tau(x_0, x_f)$ for

$$\tau = r_1(s_1)^* \cdots (s_n)^* r_{n+1}$$

Compute Succ_r
Accelerate $(\mathsf{Succ}_s)^*$

Reachability: Main Result

- The capability of computing fixpoints for simple cycles (acceleration)
- The set of types of signatures is finite

Reachability is decidable for SPDI

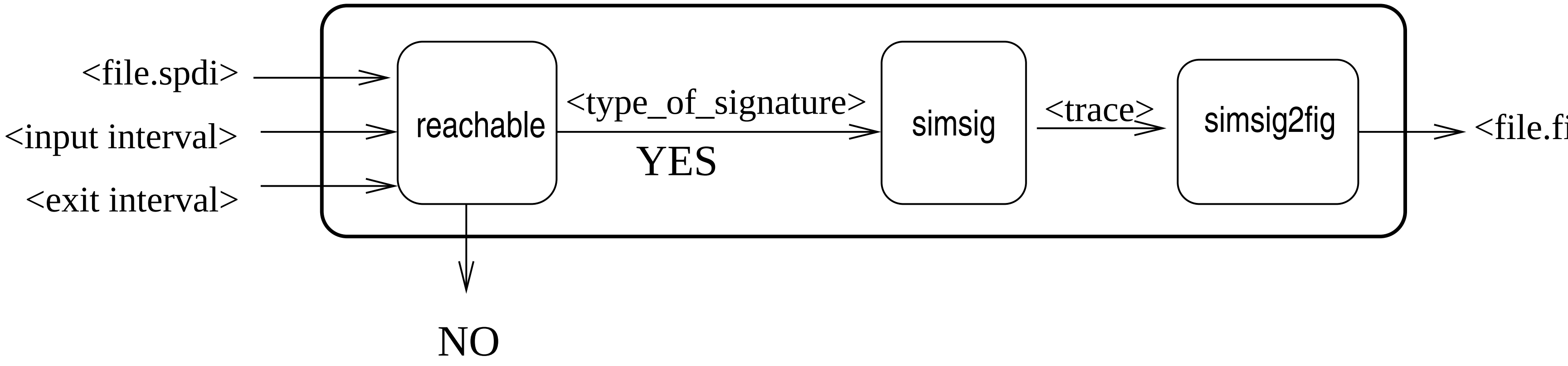
SPeeDI: a Tool for SPDIs

Implementation: SPeeDI

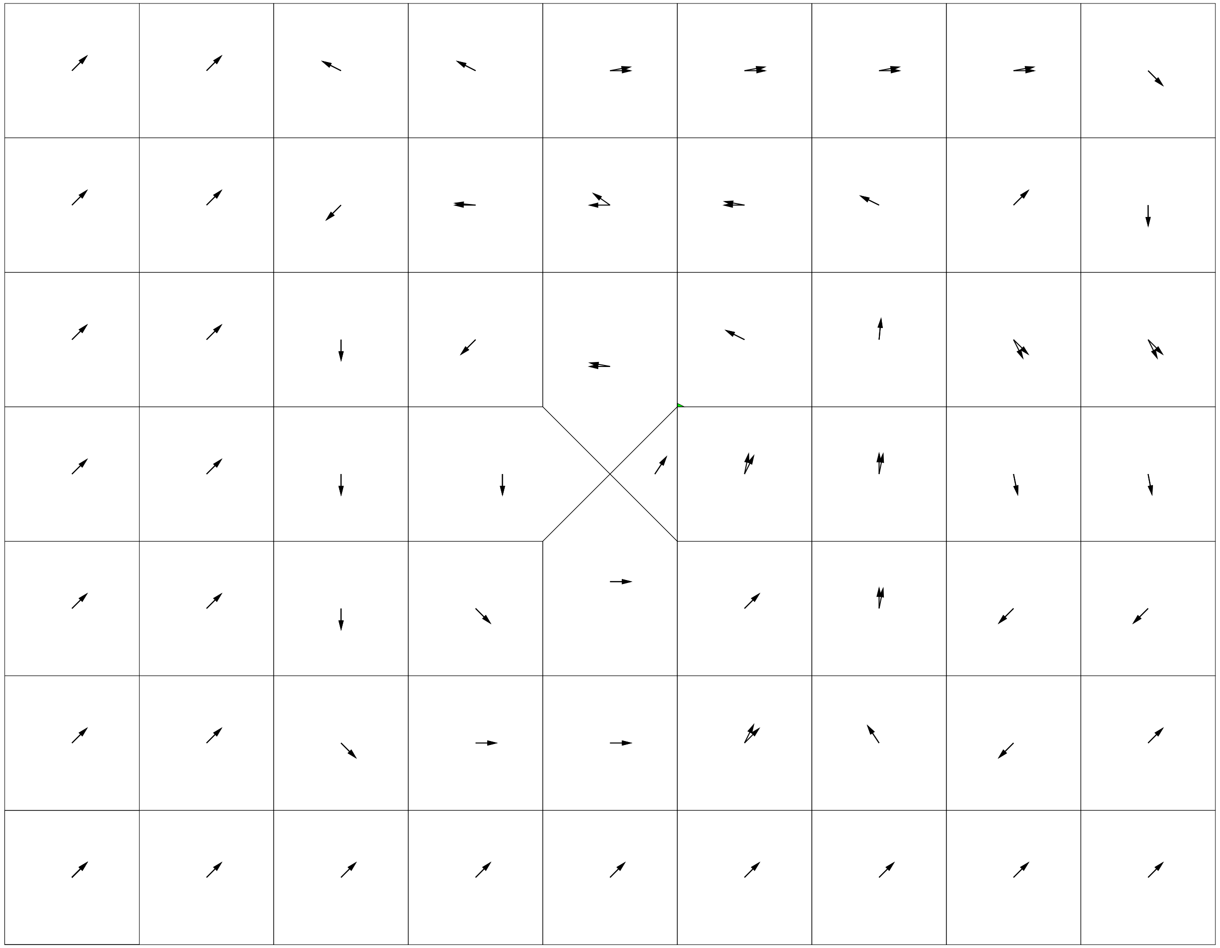
- We have implemented the reachability algorithm for SPDIs: SPeeDI (joint work with Gordon Pace)
- Language: Haskell

Implementation: SPeeDI

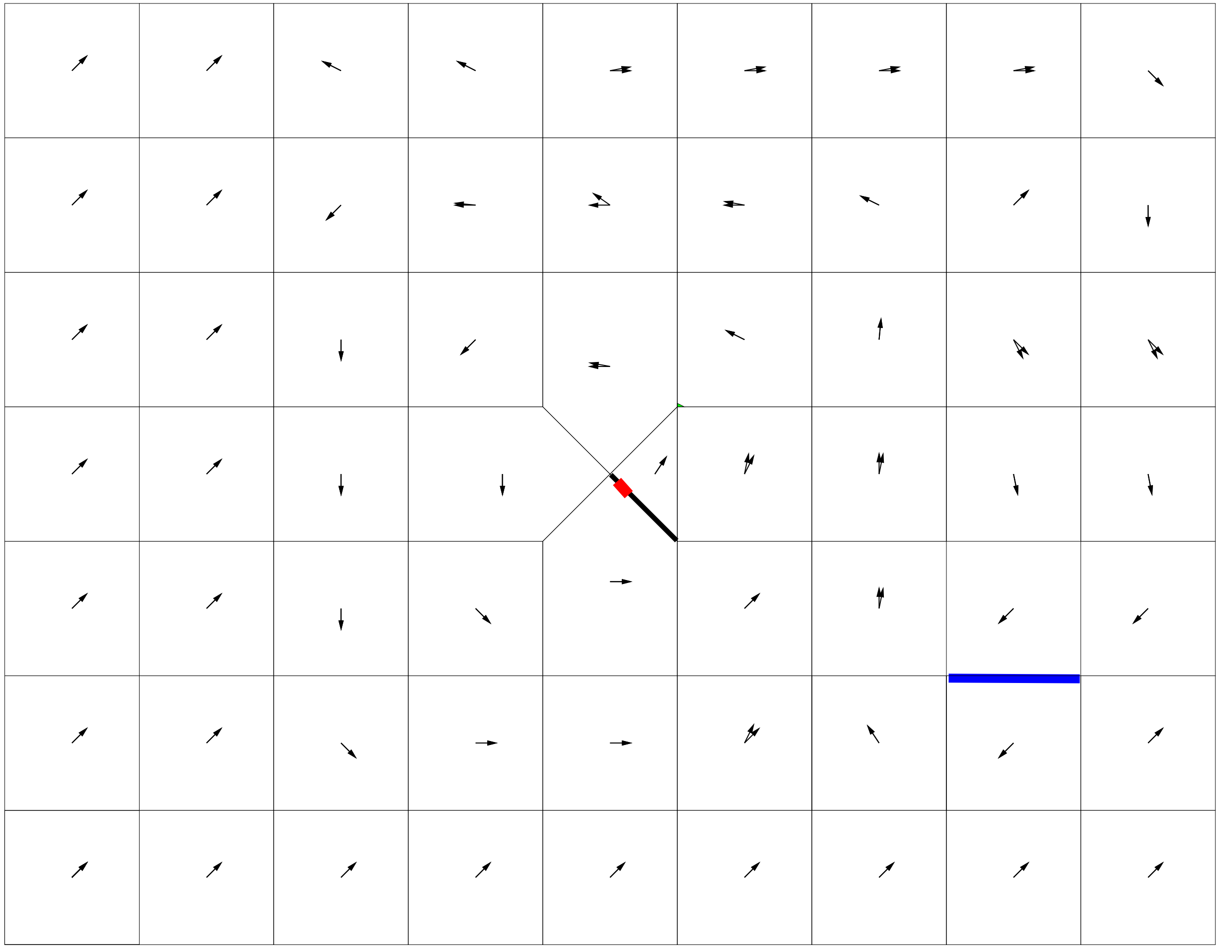
- We have implemented the reachability algorithm for SPDIs: SPeeDI (joint work with Gordon Pace)
- Language: Haskell



Implementation: SPeeDI

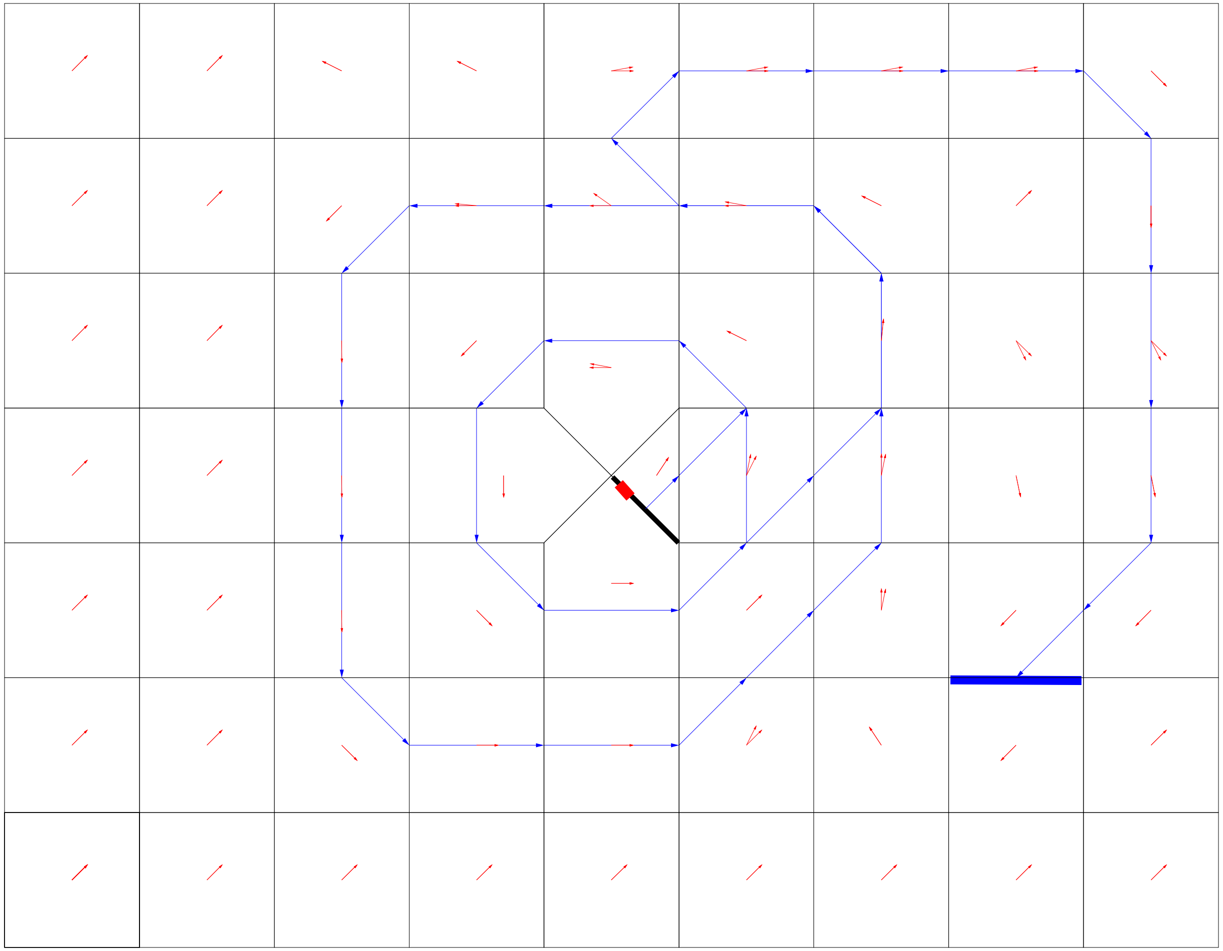


Implementation: SPeeDI



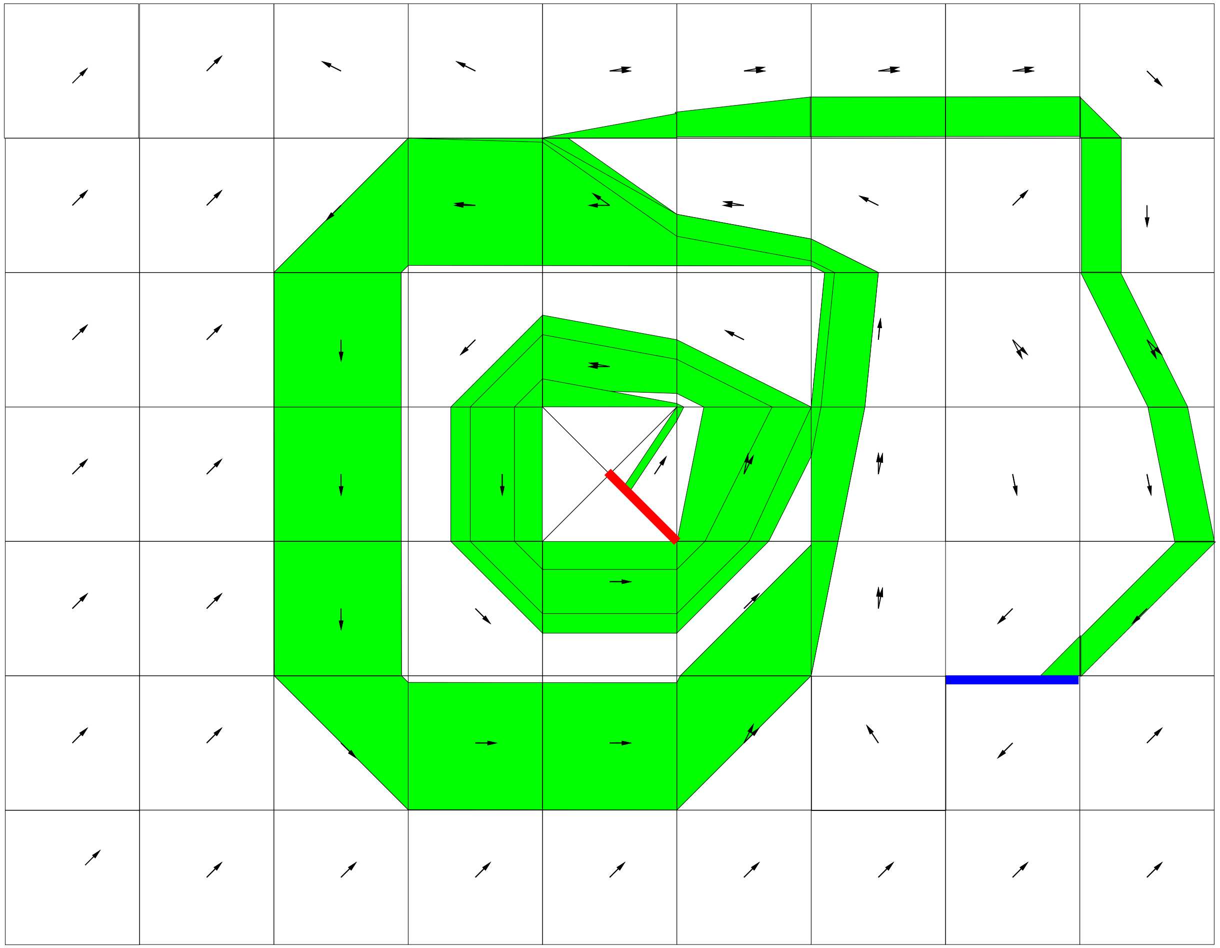
Animate

Implementation: SPeeDI



Animate

Implementation: SPeeDI



Animate

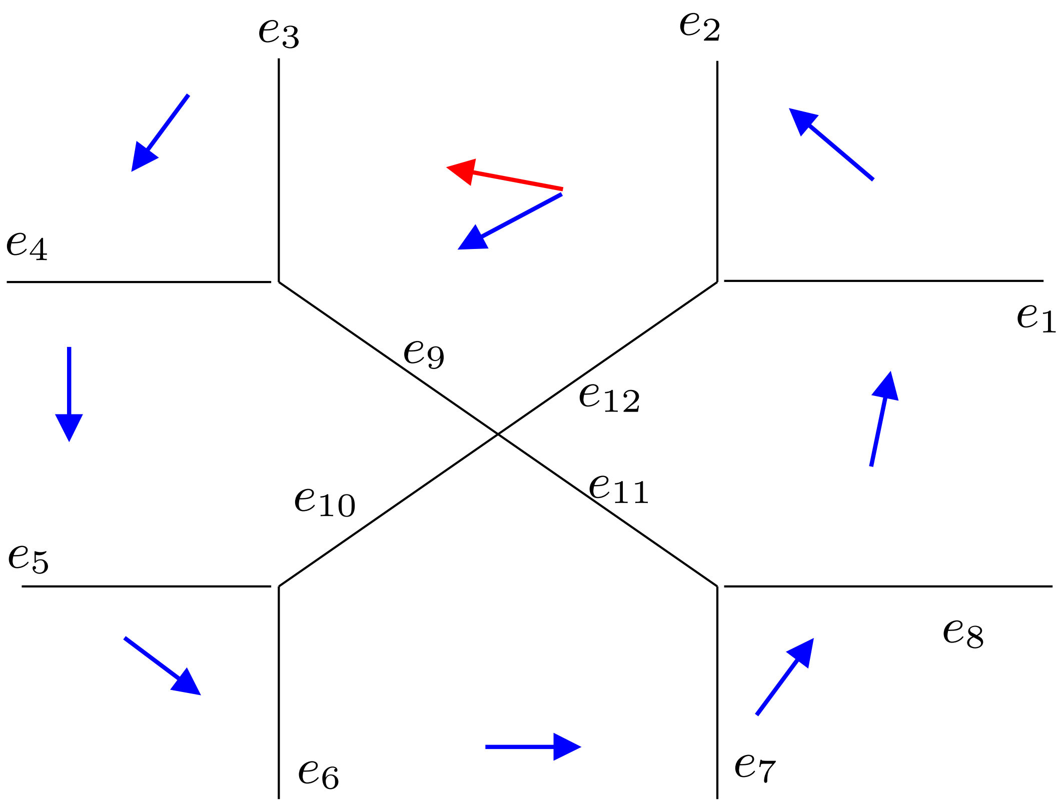
Phase Portrait of SPDIs

Phase Portrait

Phase Portrait: a picture of important objects of a dynamical system

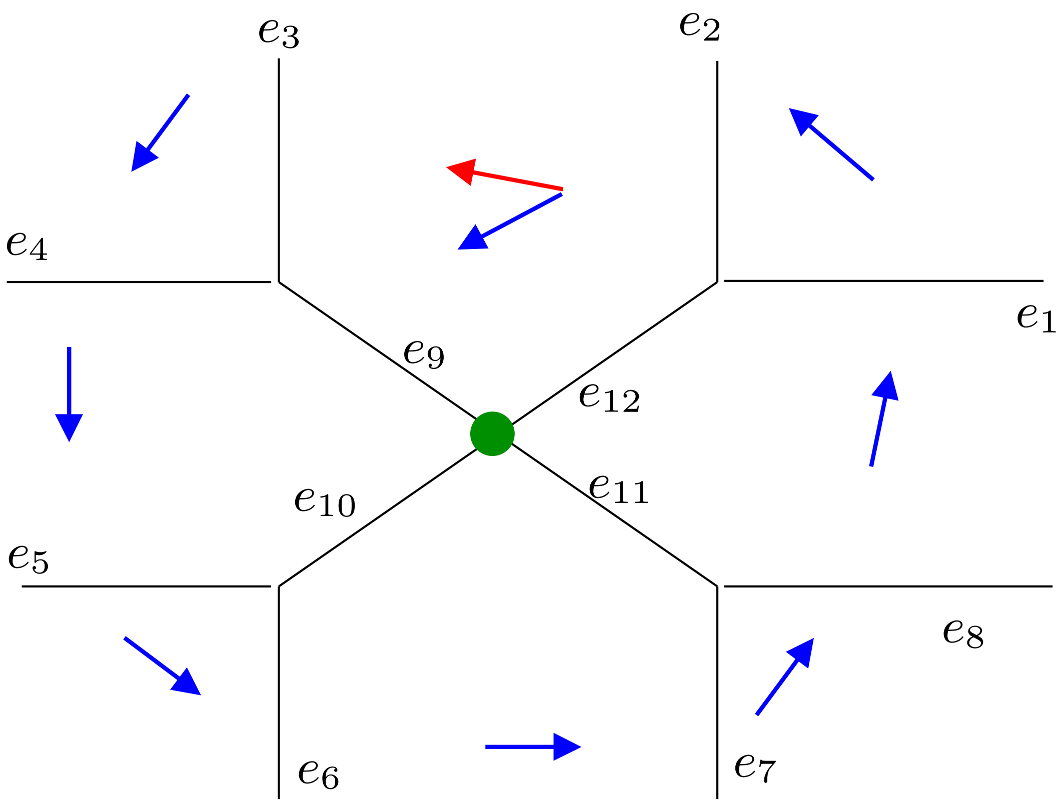
Phase Portrait

Phase Portrait: a picture of important objects of a dynamical system



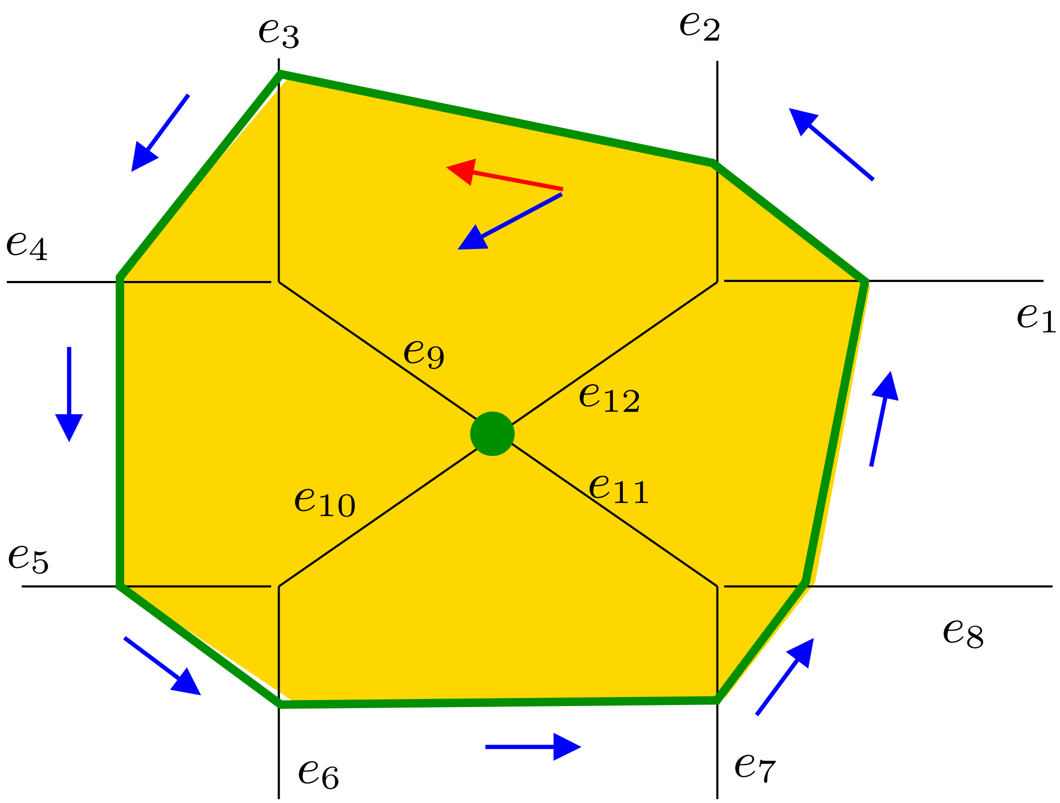
Phase Portrait

Phase Portrait: a picture of important objects of a dynamical system



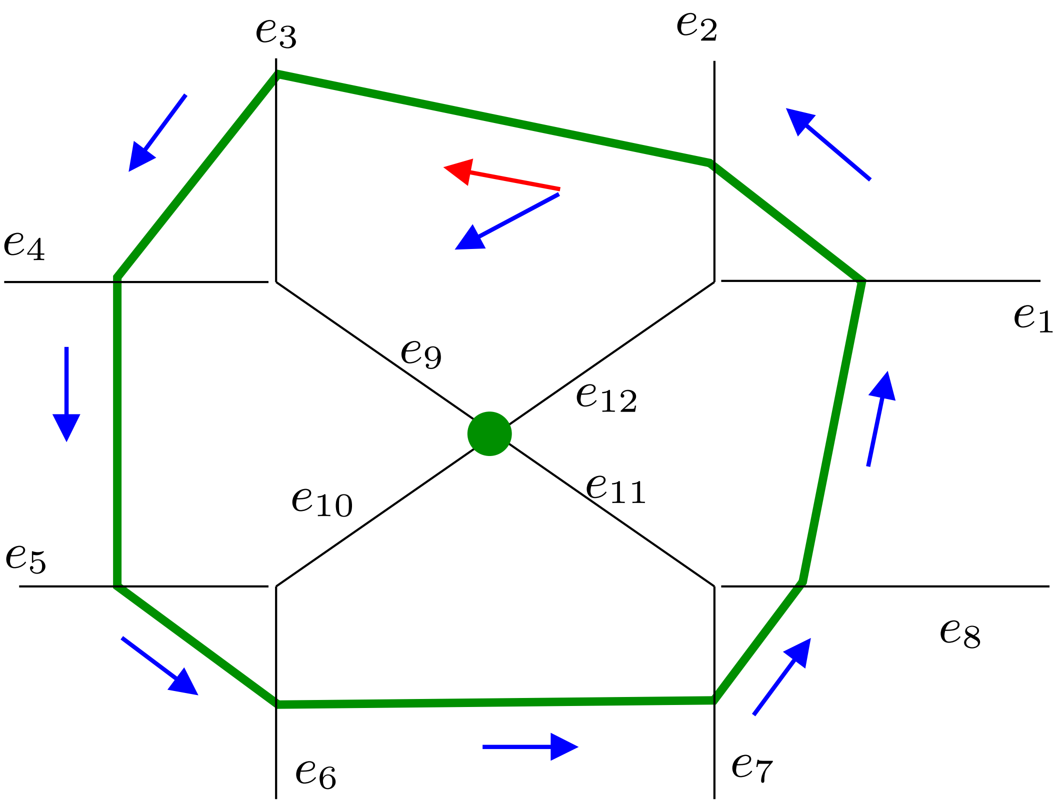
Phase Portrait

Phase Portrait: a picture of important objects of a dynamical system



Phase Portrait

Phase Portrait: a picture of important objects of a dynamical system



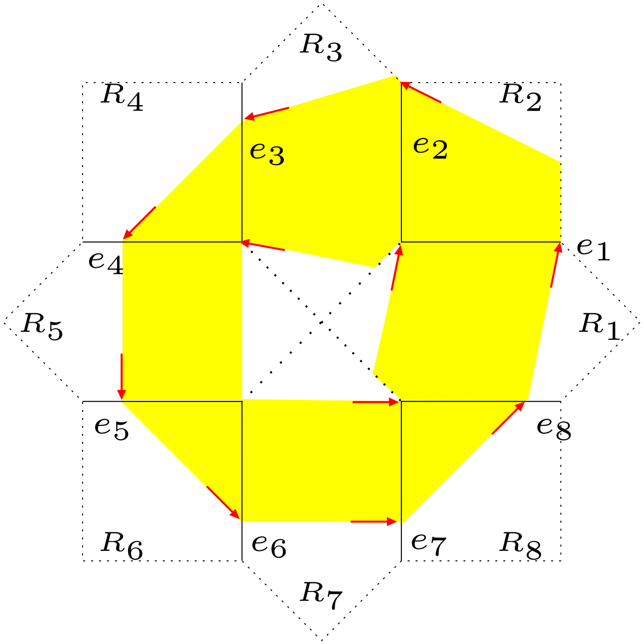
Viability Kernel

$\mathsf{Viab}(\sigma)$: Is the greatest set of initial points of trajectories which can cycle forever in σ

Viability Kernel

Viab(σ): Is the greatest set of initial points of trajectories which can cycle forever in σ

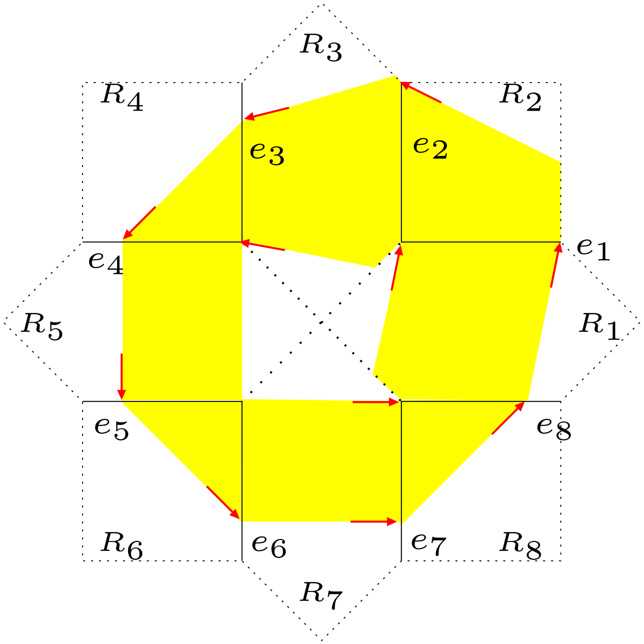
Example: $\sigma = e_1 e_2 \dots e_8 e_1$



Viability Kernel

Viab(σ): Is the greatest set of initial points of trajectories which can cycle forever in σ

Example: $\sigma = e_1 e_2 \ldots e_8 e_1$



Theorem: $\mathsf{Viab}(\sigma) = \overline{\mathsf{Pre}}_\sigma(\mathsf{Dom}(\mathsf{Succ}_\sigma))$

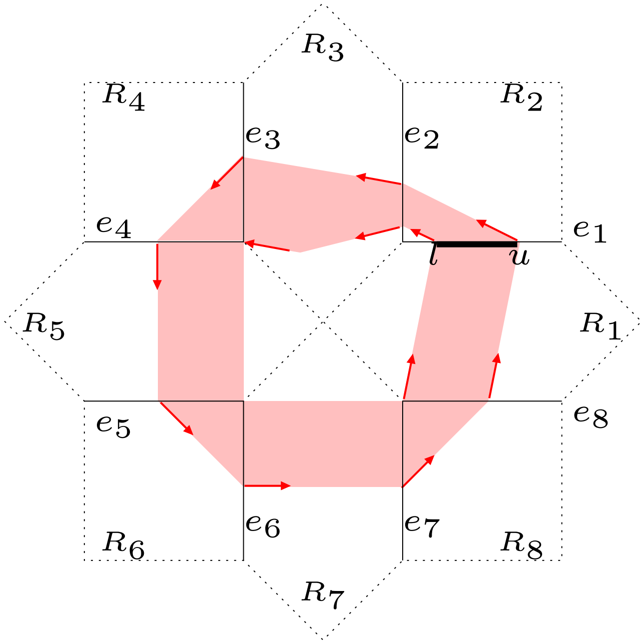
Controllability Kernel

$\mathsf{Cntr}(\sigma)$: Is the greatest set of mutually reachable points via trajectories that remain in the cycle

Controllability Kernel

Cntr(σ): Is the greatest set of mutually reachable points via trajectories that remain in the cycle

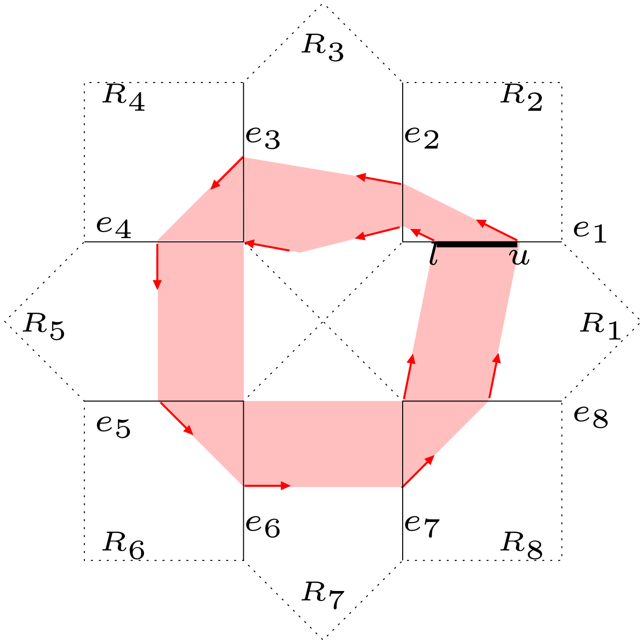
Example: $\sigma = e_1 e_2 \ldots e_8 e_1$



Controllability Kernel

$\mathsf{Cntr}(\sigma)$: Is the greatest set of mutually reachable points via trajectories that remain in the cycle

Example: $\sigma = e_1 e_2 \ldots e_8 e_1$



Theorem: $\mathsf{Cntr}(\sigma) = (\overline{\mathsf{Succ}}_\sigma \cap \overline{\mathsf{Pre}}_\sigma)(\mathcal{C}_{\mathcal{D}}(\sigma))$

Viability Kernel

Algorithm: phase portrait for SPDIs

for each simple cycle σ **do**
Compute $\mathsf{Viab}(\sigma)$ (*viability kernel*)
Compute $\mathsf{Cntr}(\sigma)$ (*controllability kernel*)

Viability Kernel

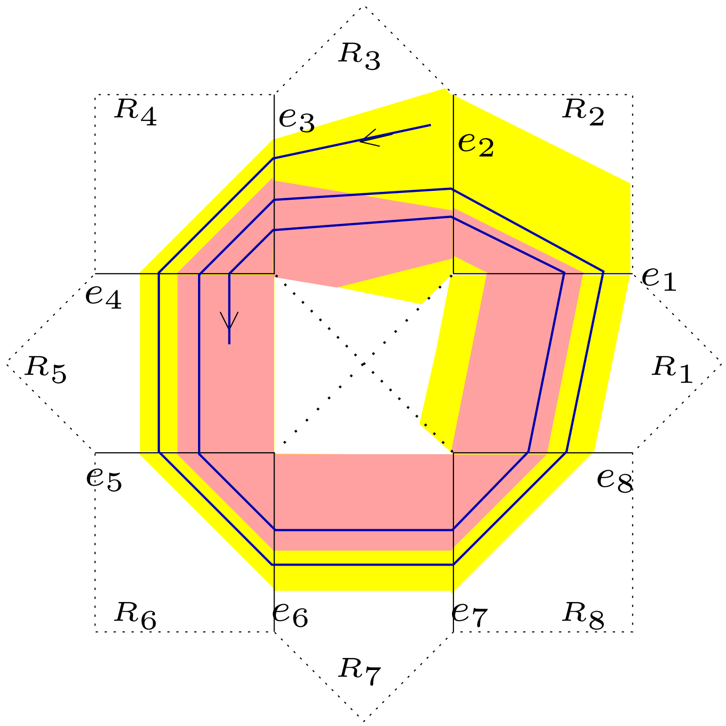
Algorithm: phase portrait for SPDIs

for each simple cycle σ **do**
Compute $\mathsf{Viab}(\sigma)$ (*viability kernel*)
Compute $\mathsf{Cntr}(\sigma)$ (*controllability kernel*)

Both kernels are exactly computed by non-iterative algorithms!

Properties of the Kernels

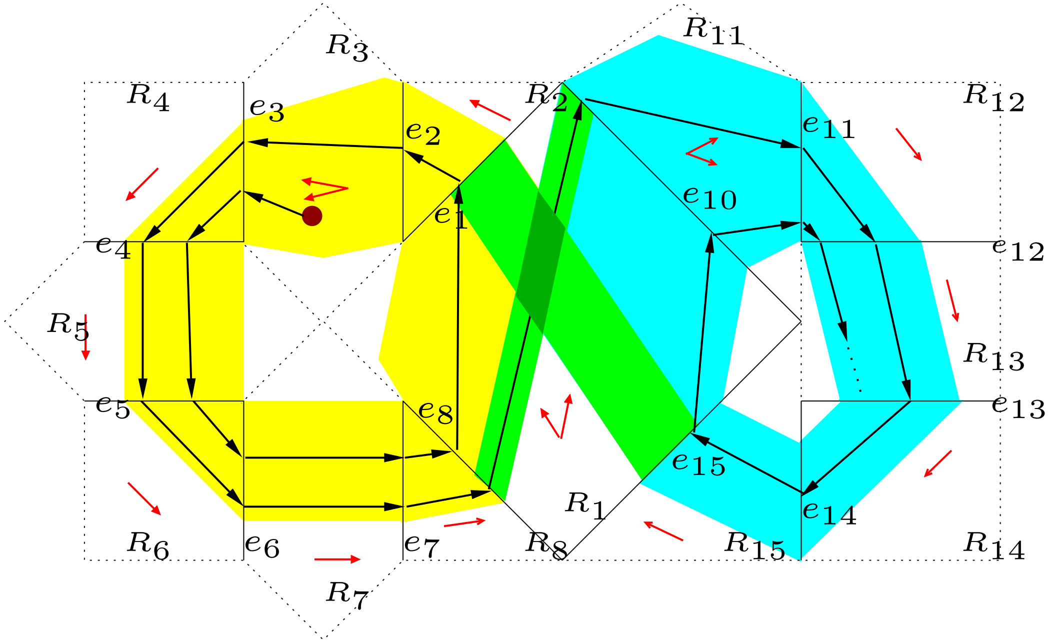
Theorem: Any viable trajectory in σ converges to $\mathsf{Cntr}(K_\sigma)$



- Controllability Kernel: "Weak" analog of limit cycle
- Viability Kernel: Its "local" attraction basin

Convergence Properties

Every trajectory with infinite signature without self-crossings converges to the controllability kernel of some simple edge-cycle



Between Decidable and Undecidable

More complex 2-dim systems

What happens if …

- … we allow jumps?
- … the PCD is on a 2-dim surface/manifold?
- … ?

More complex 2-dim systems

What happens if . . .

- . . . we allow jumps?
- . . . the PCD is on a 2-dim surface/manifold?
- . . . ?

Answer: Reachability is equivalent to a well known open problem

Our Reference Model

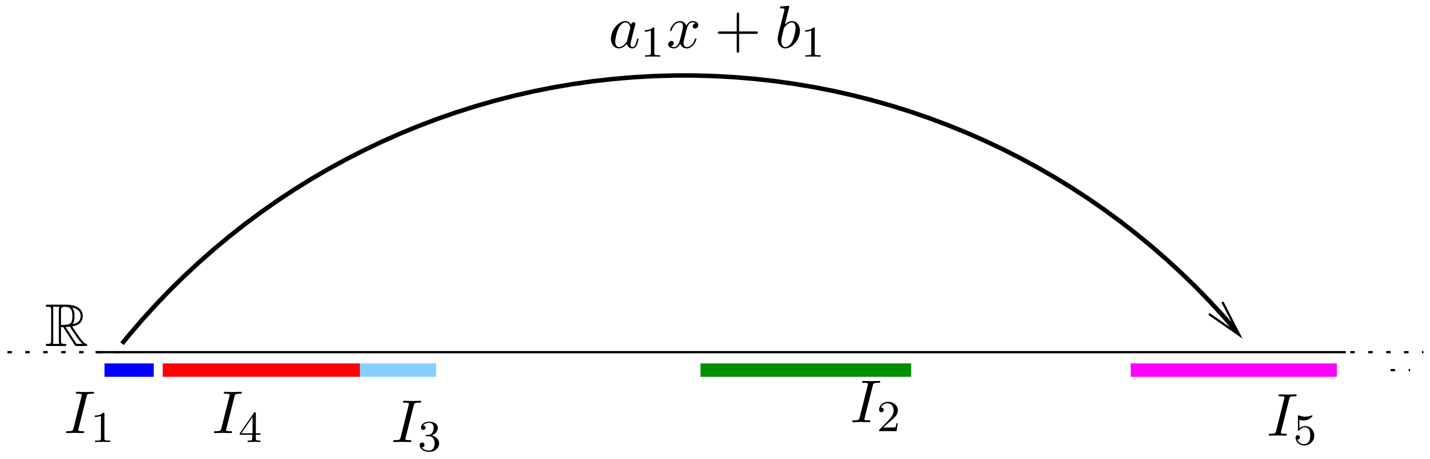
1-dim Piecewise Affine Maps (PAMs):

$f : \mathbb{R} \to \mathbb{R}$, $f(x) = a_i x + b_i$ for $x \in I_i$

Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

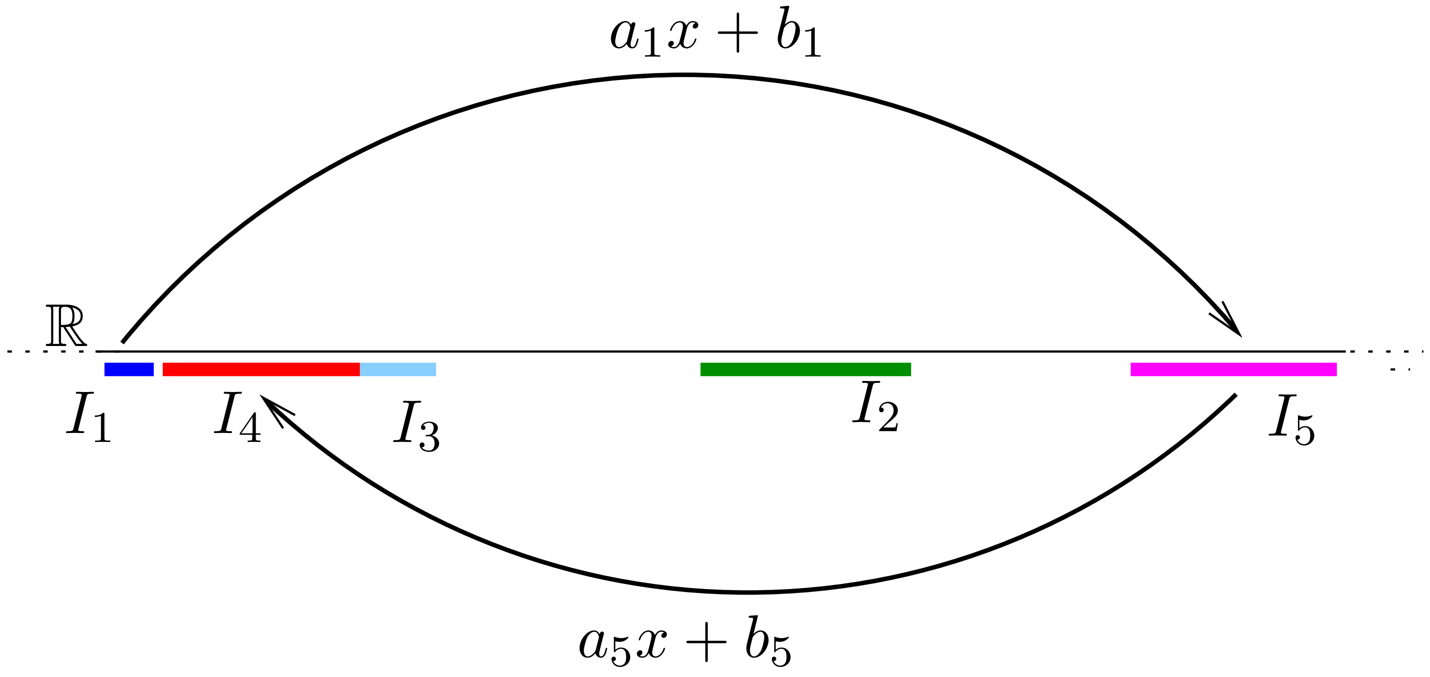
$f : \mathbb{R} \rightarrow \mathbb{R}, \;\; f(x) = a_i x + b_i$ for $x \in I_i$



Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

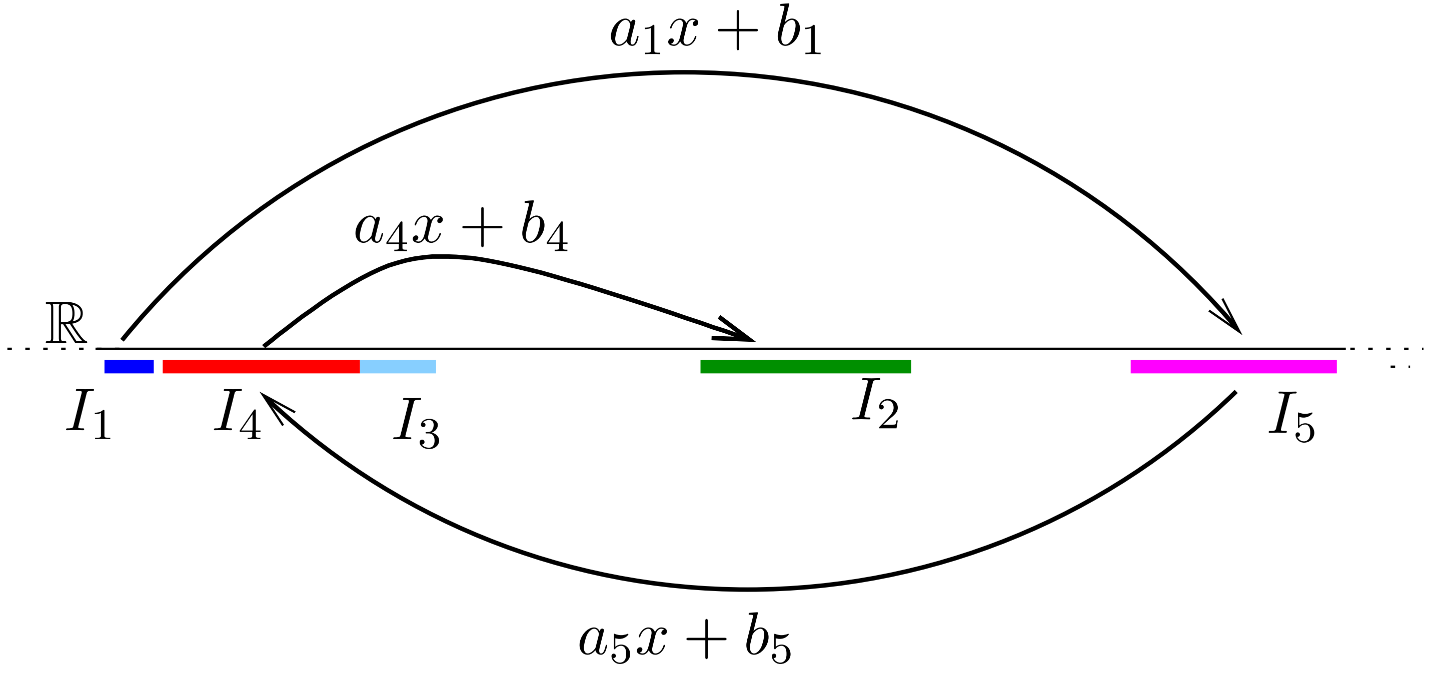
$f : \mathbb{R} \to \mathbb{R}, \;\; f(x) = a_i x + b_i$ for $x \in I_i$



Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

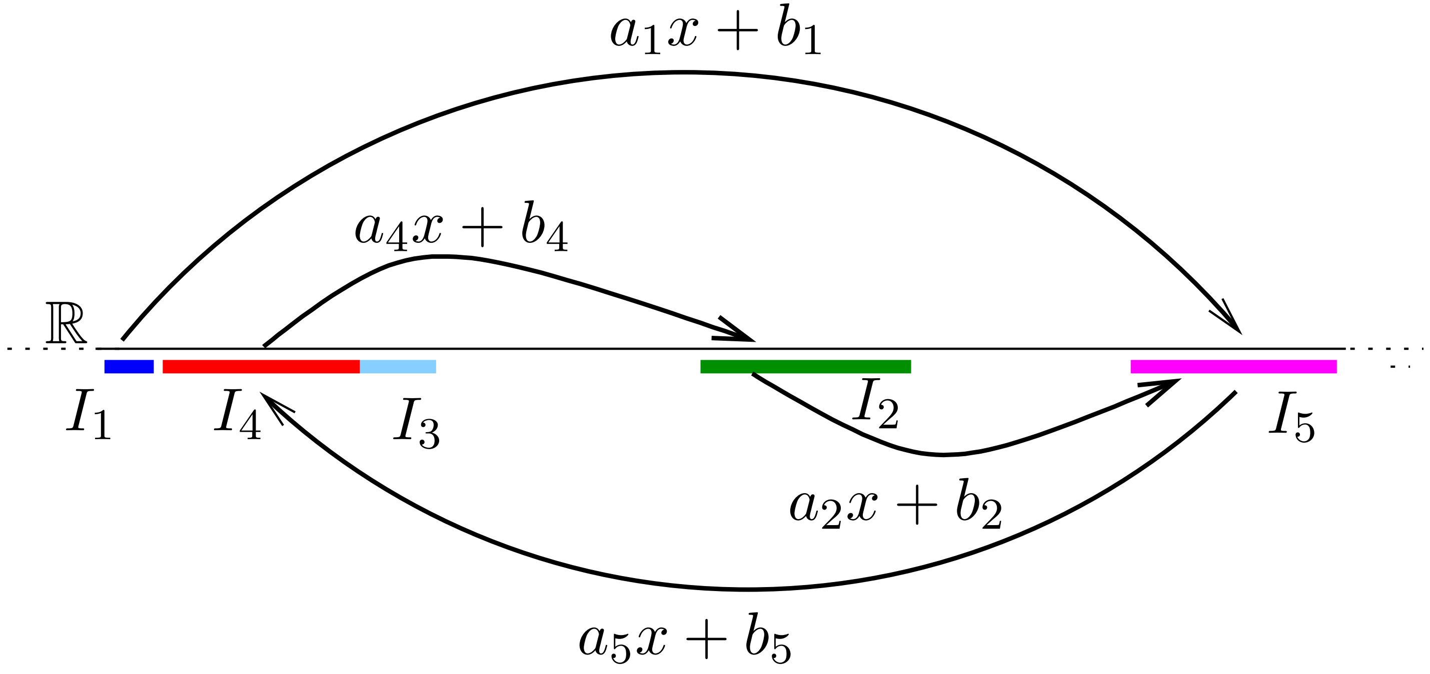
$f : \mathbb{R} \rightarrow \mathbb{R}, \;\; f(x) = a_i x + b_i$ for $x \in I_i$



Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

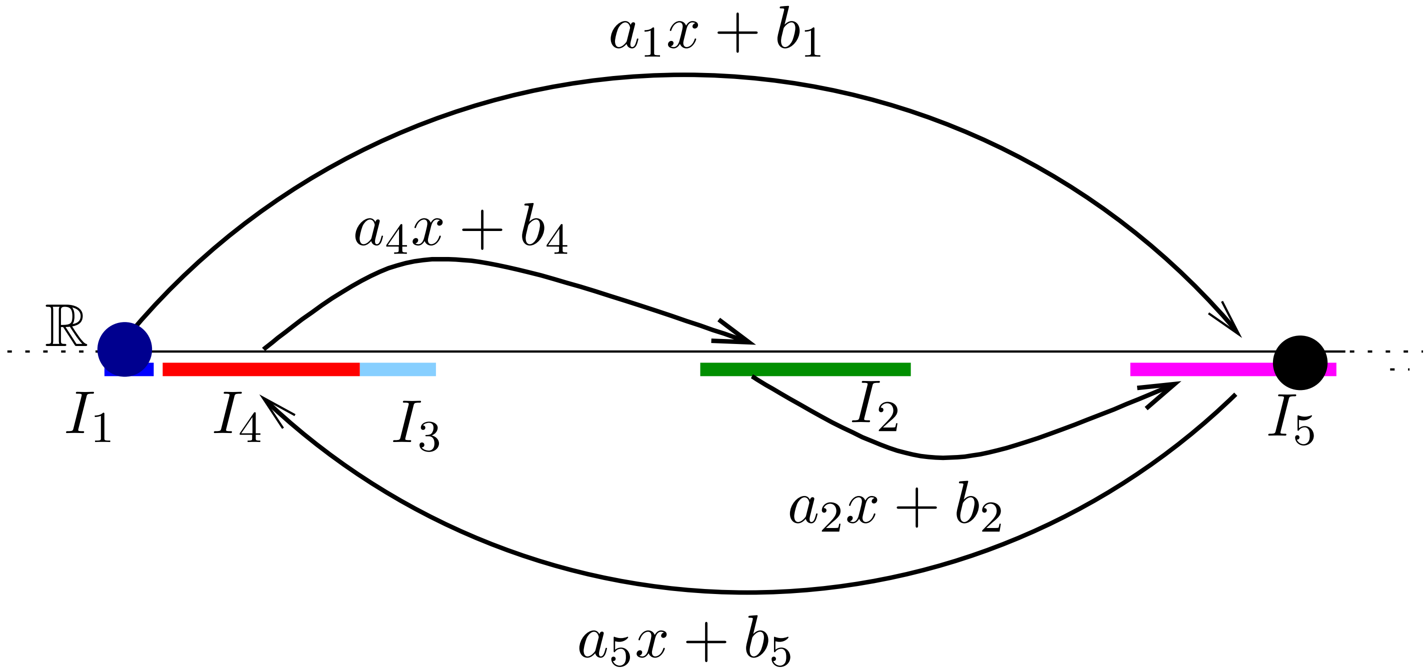
$f : \mathbb{R} \rightarrow \mathbb{R}, \; f(x) = a_i x + b_i$ for $x \in I_i$



Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

$f : \mathbb{R} \to \mathbb{R}, \; f(x) = a_i x + b_i$ for $x \in I_i$

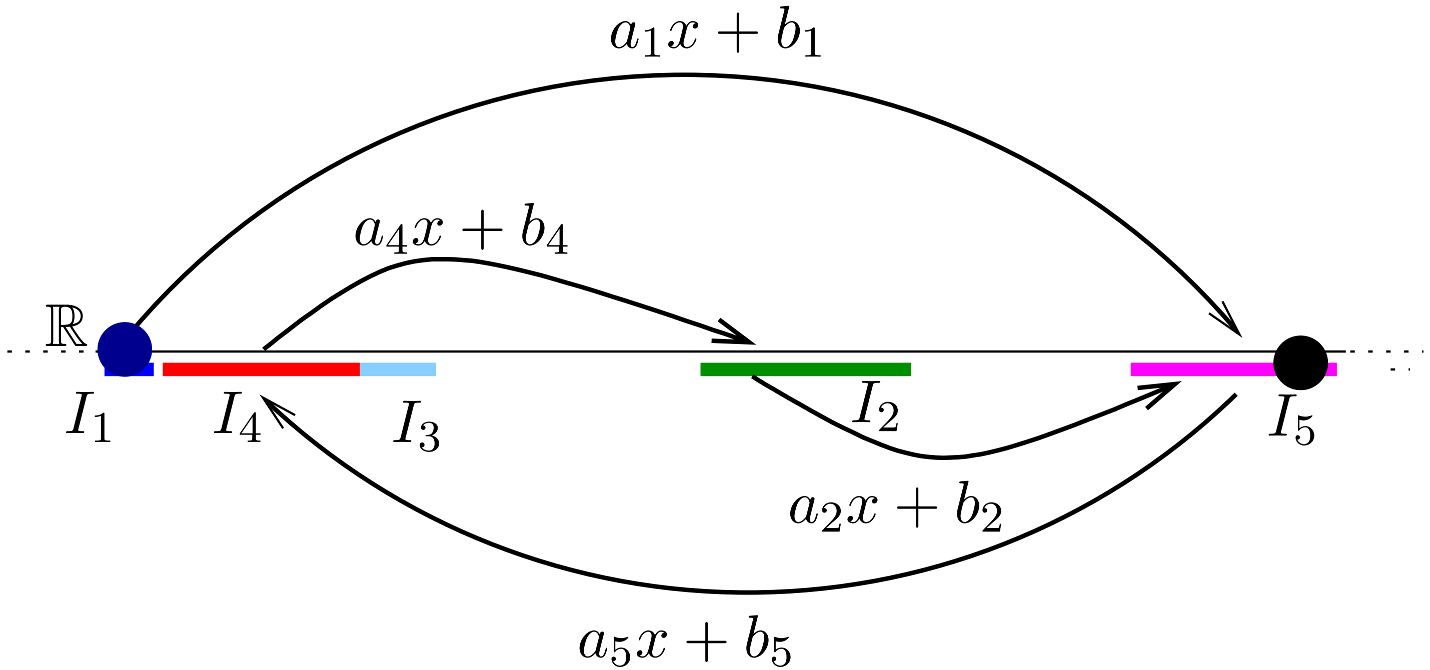


Reachability?

Our Reference Model

1-dim Piecewise Affine Maps (PAMs):

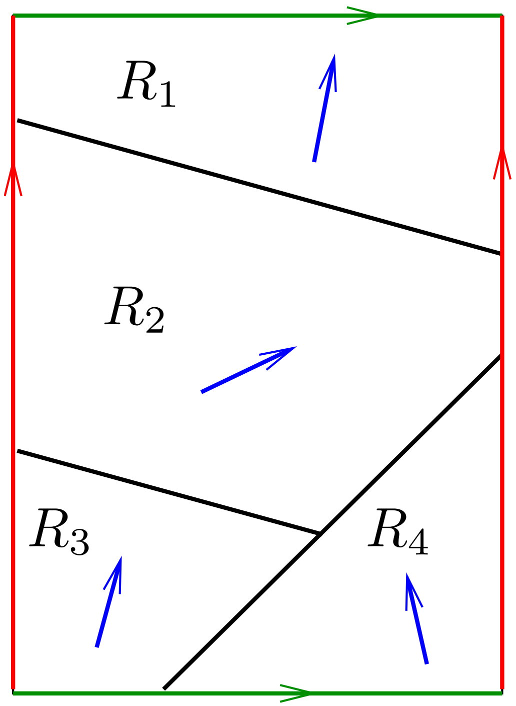
$f : \mathbb{R} \to \mathbb{R},\ \ f(x) = a_i x + b_i$ for $x \in I_i$



Reachability? Open problem!

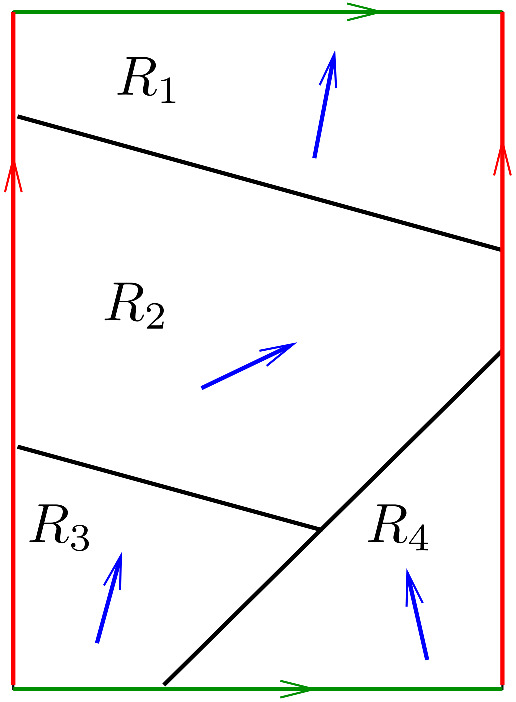
PCD on 2-dim manifolds (PCD_{2m})

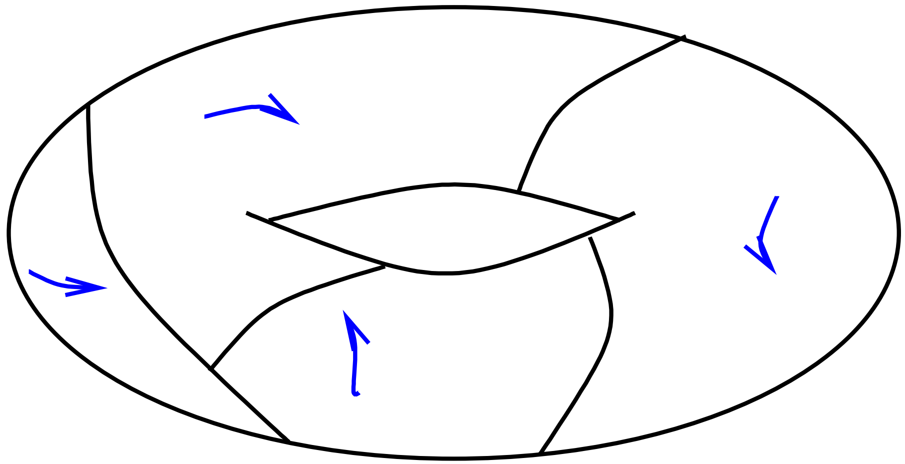
Example: Torus



PCD on 2-dim manifolds (PCD_{2m})

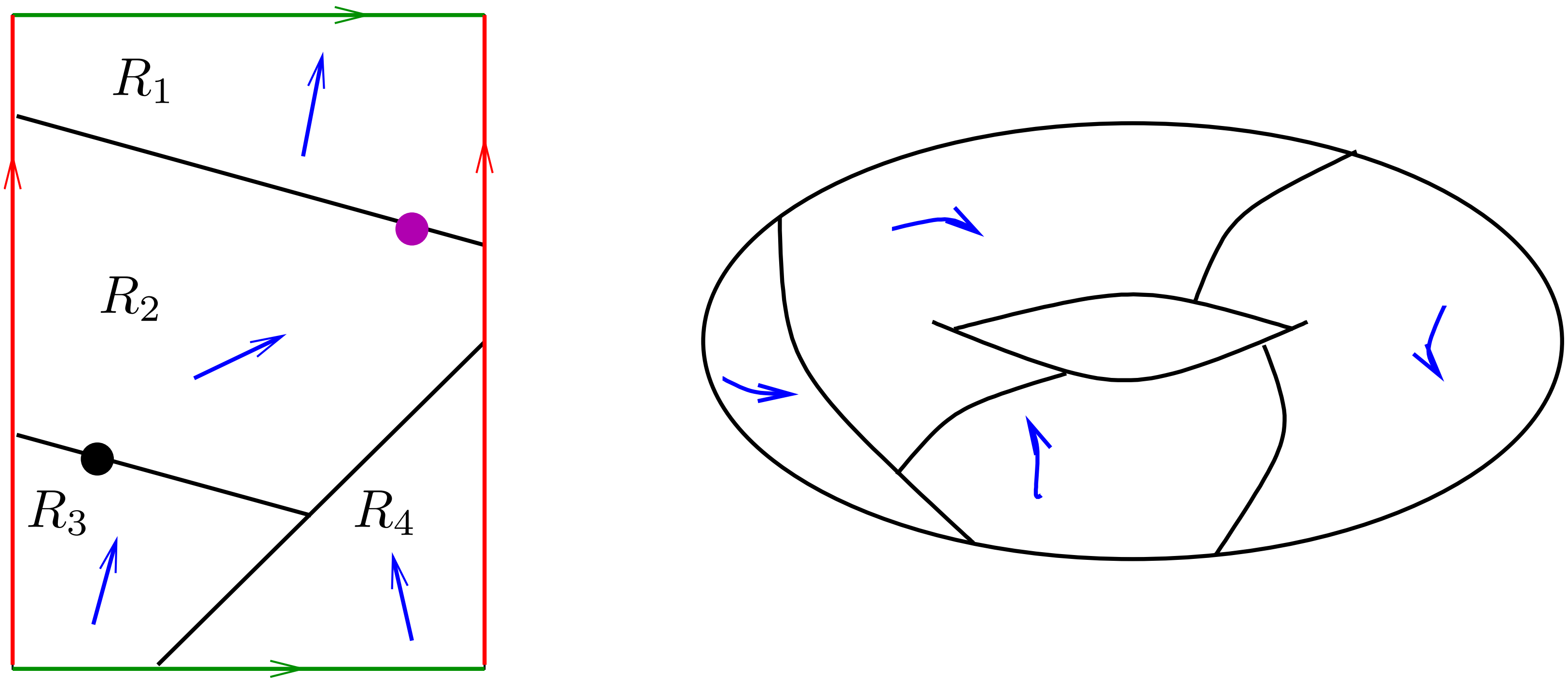
Example: Torus





PCD on 2-dim manifolds (PCD_{2m})

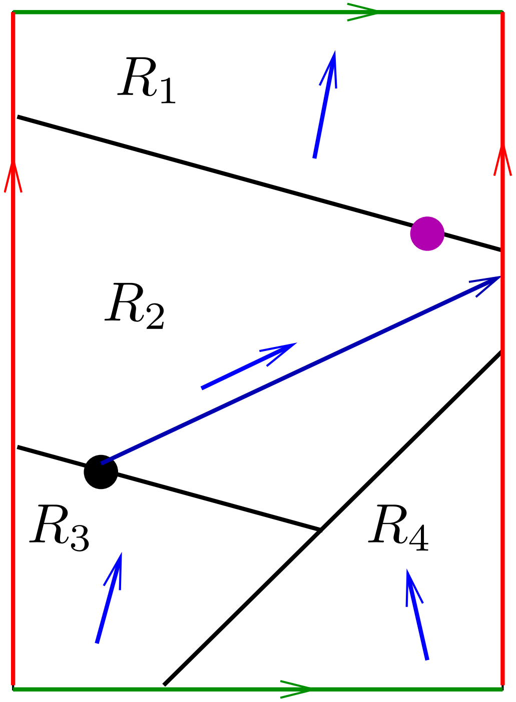
Example: Torus

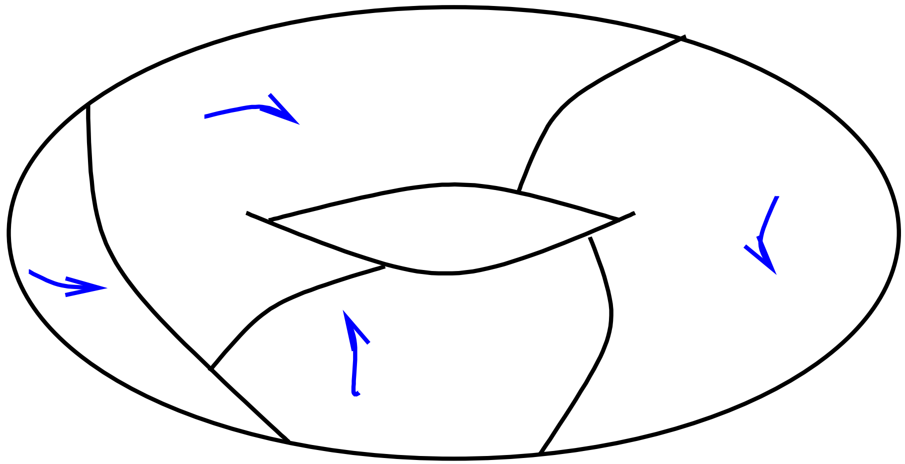


Reachability?

PCD on 2-dim manifolds (PCD_{2m})

Example: Torus

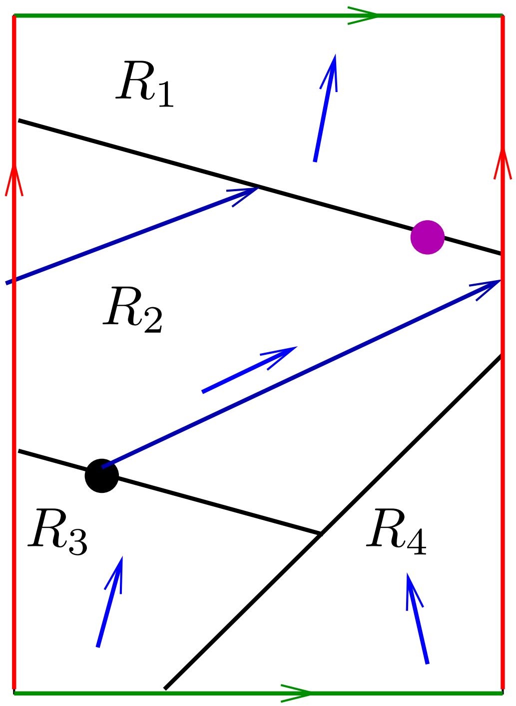


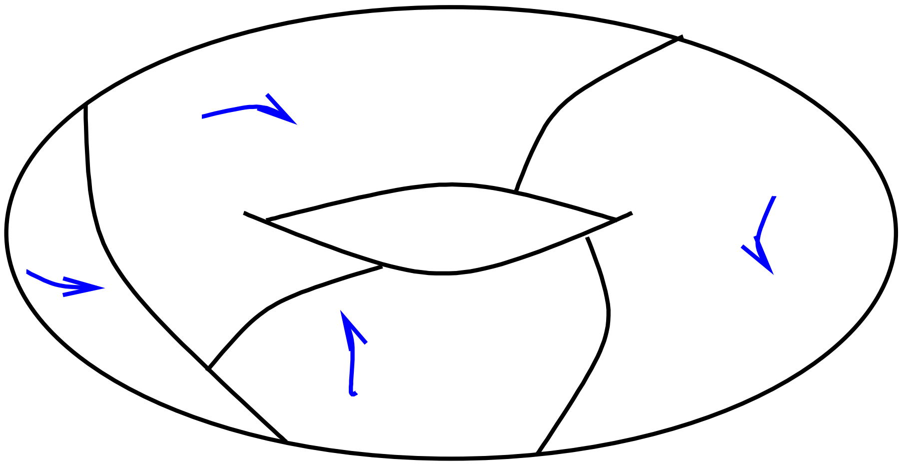


Reachability?

PCD on 2-dim manifolds (PCD_{2m})

Example: Torus

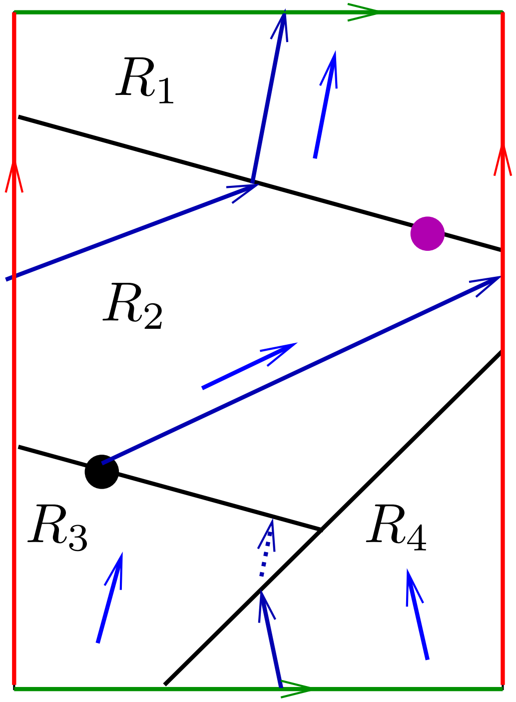


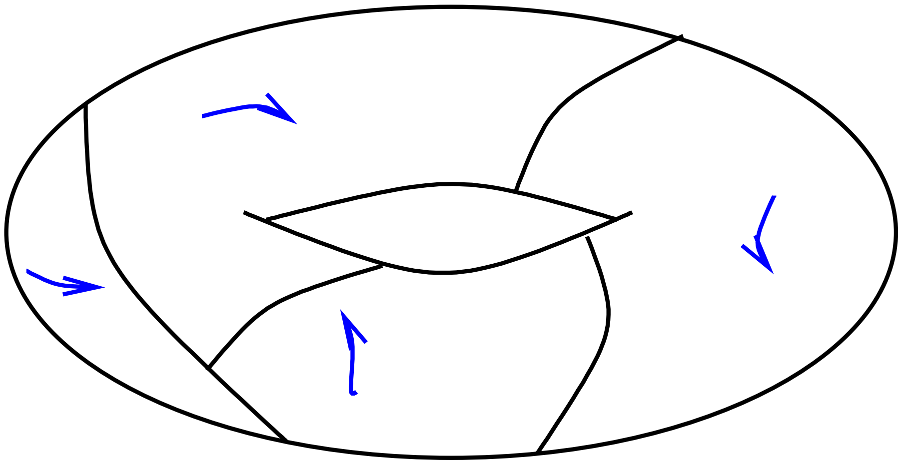


Reachability?

PCD on 2-dim manifolds ($\mathrm{PCD_{2m}}$)

Example: Torus

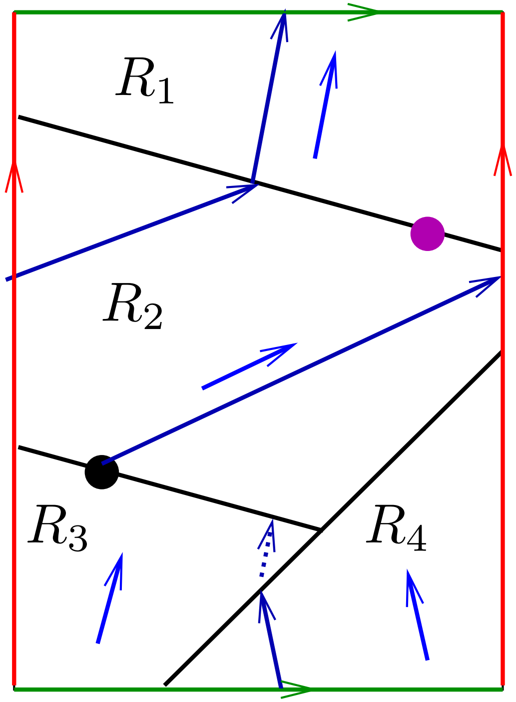


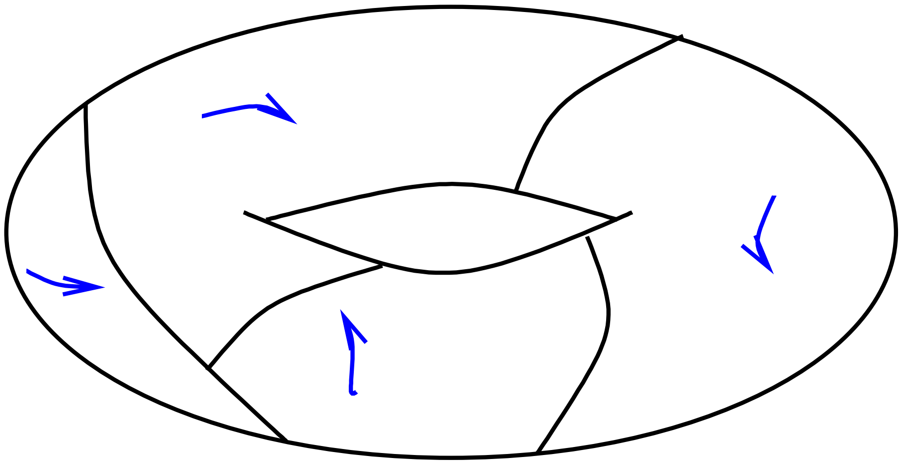


Reachability?

PCD on 2-dim manifolds (PCD_{2m})

Example: Torus

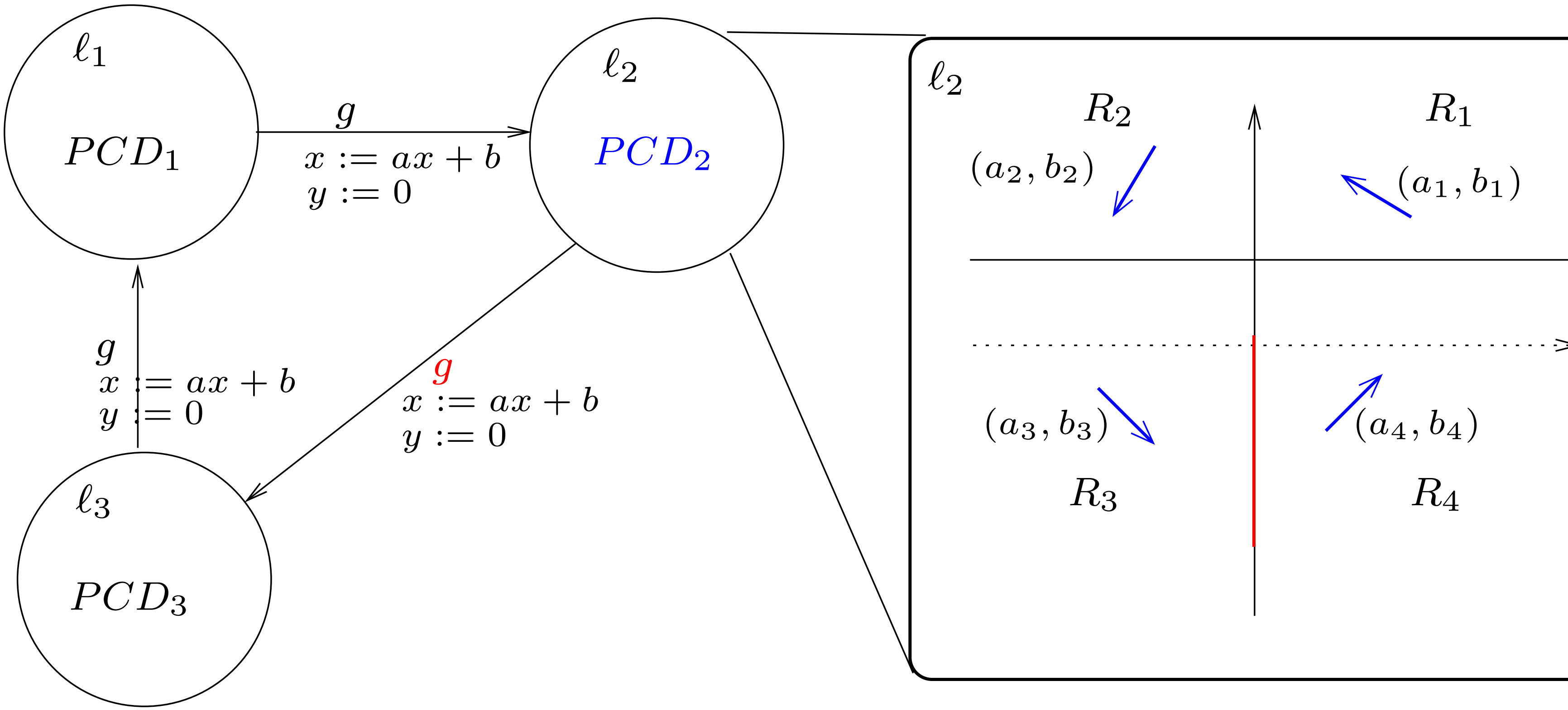




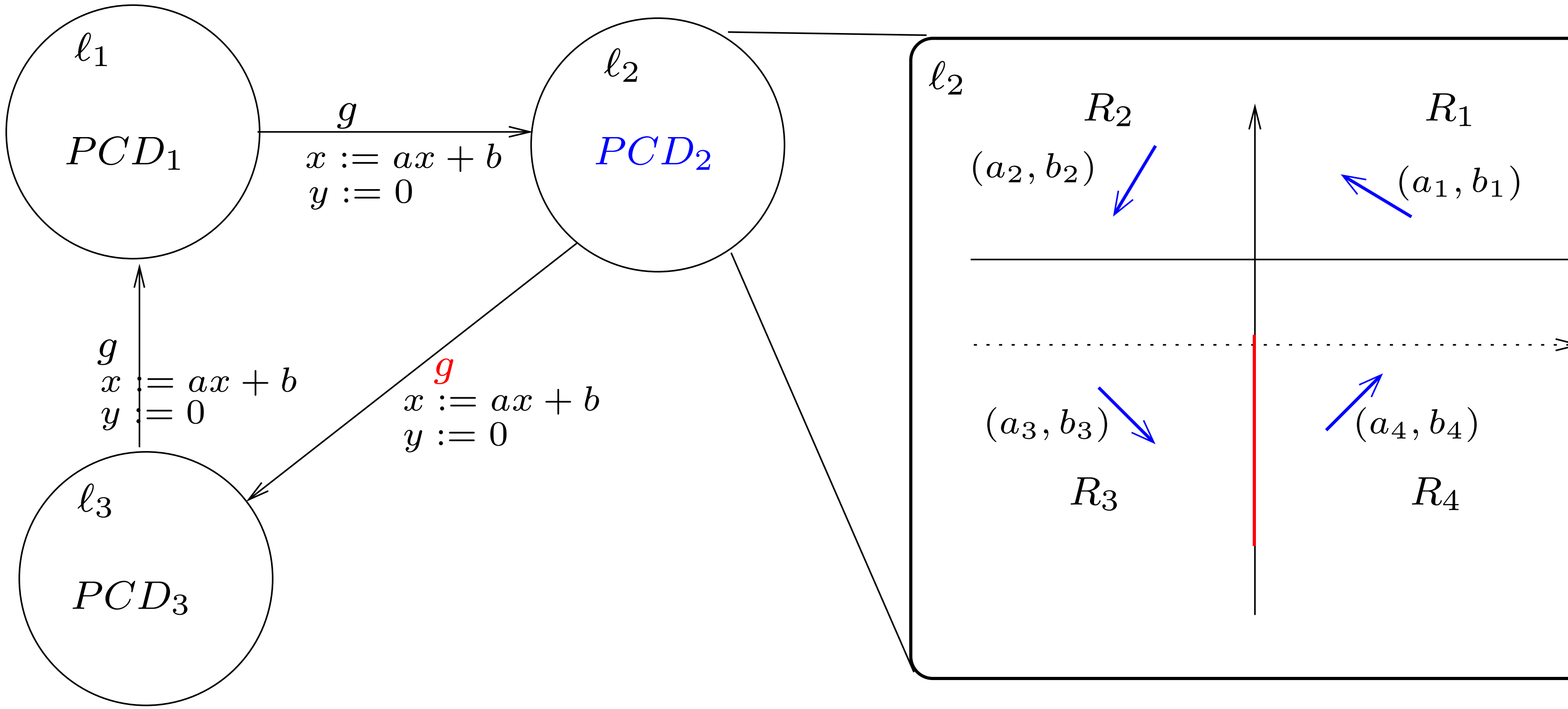
Reachability?

Theorem: $\mathrm{PCD}_{2m} \equiv \mathrm{PAM}$

Hierarchical PCDs (HPCD)



Hierarchical PCDs (HPCD)



Reachability?

Theorem: HPCD $\equiv$ PAM

Undecidable 2-dim Systems

Undecidability Results

- HPCDs with One Counter (HPCD_{1c})
- HPCDs with Infinite Partition (HPCD_{∞})
- Origin-dependent rate HPCDs (HPCD_{x})

Undecidability Results

- HPCDs with One Counter (HPCD_{1c})
- HPCDs with Infinite Partition (HPCD_{∞})
- Origin-dependent rate HPCDs (HPCD_{x})

Reachability?

Undecidability Results

- HPCDs with One Counter (HPCD_{1c})
- HPCDs with Infinite Partition (HPCD_{∞})
- Origin-dependent rate HPCDs (HPCD_{x})

Reachability? UNDECIDABLE!

Theorem:

HPCD_{1c}, HPCD_{∞} and HPCD_{x}

simulate

Turing machines

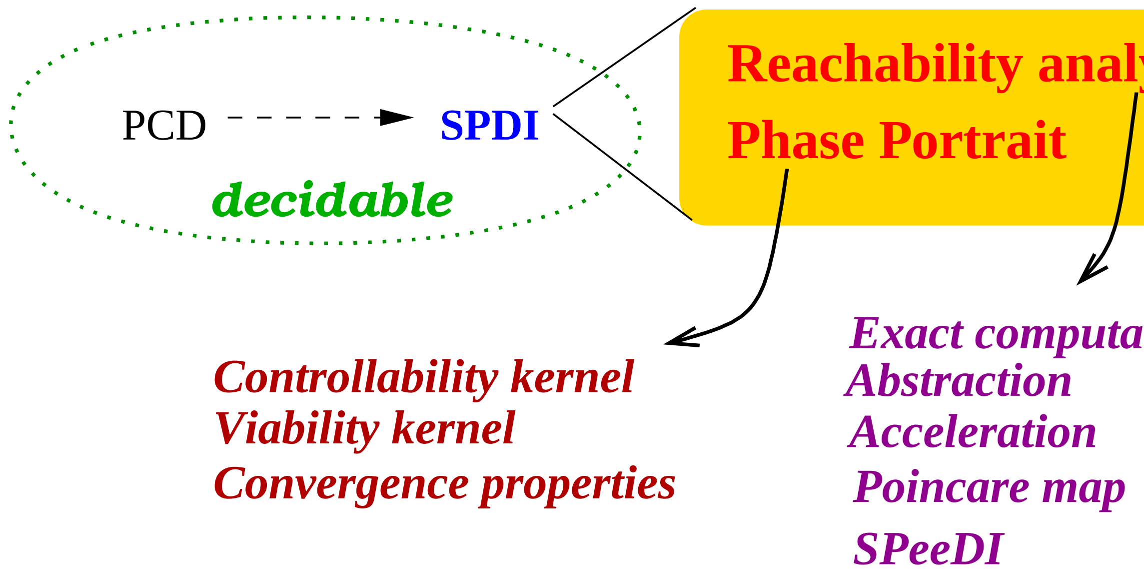
Summary of Results

Summary of Results

PCD - - - - - - -► **SPDI**

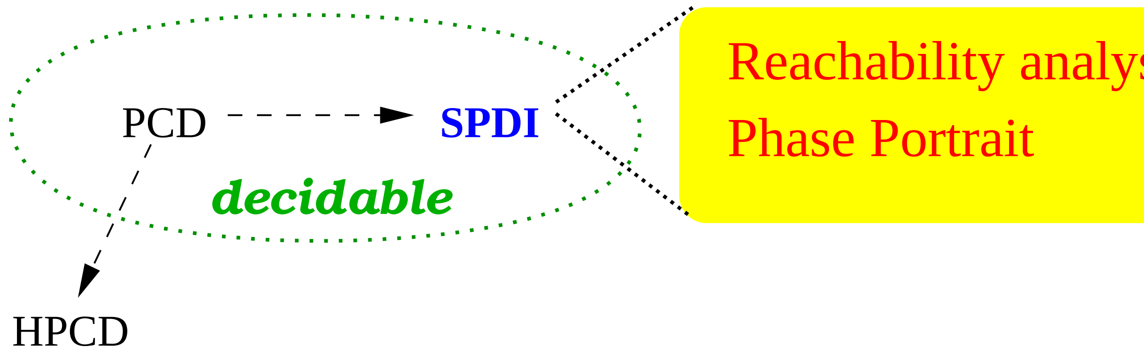
A - - - -► B "A is a particular case of B"

Summary of Results



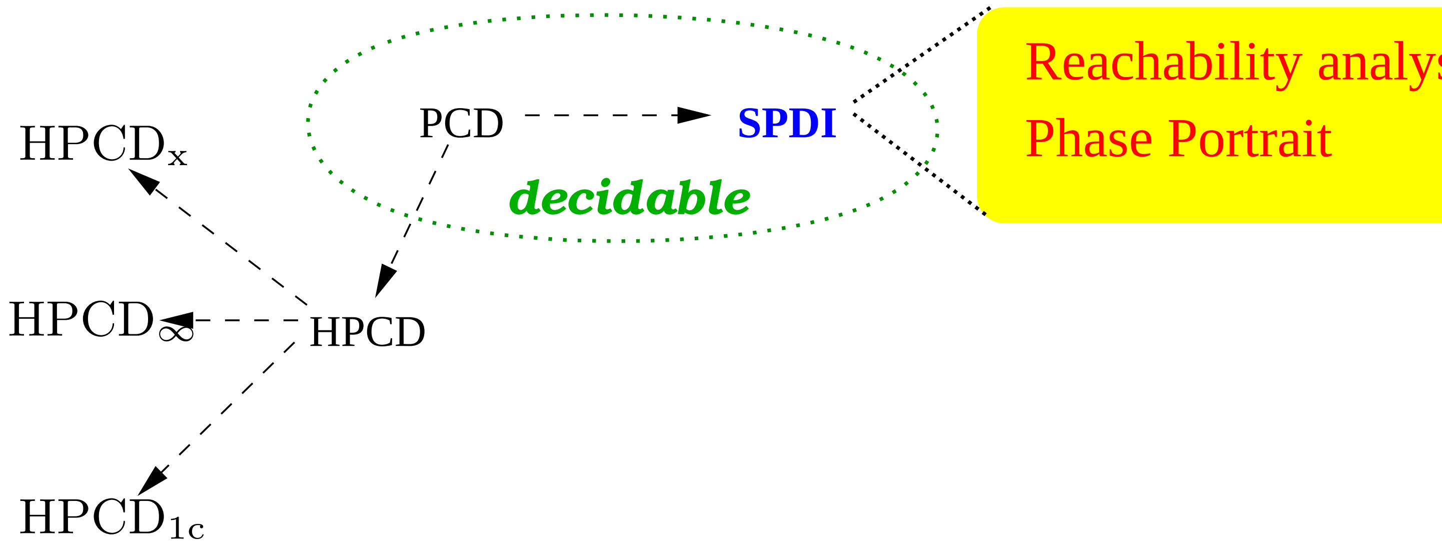
A - - - -► B "A is a particular case of B"

Summary of Results



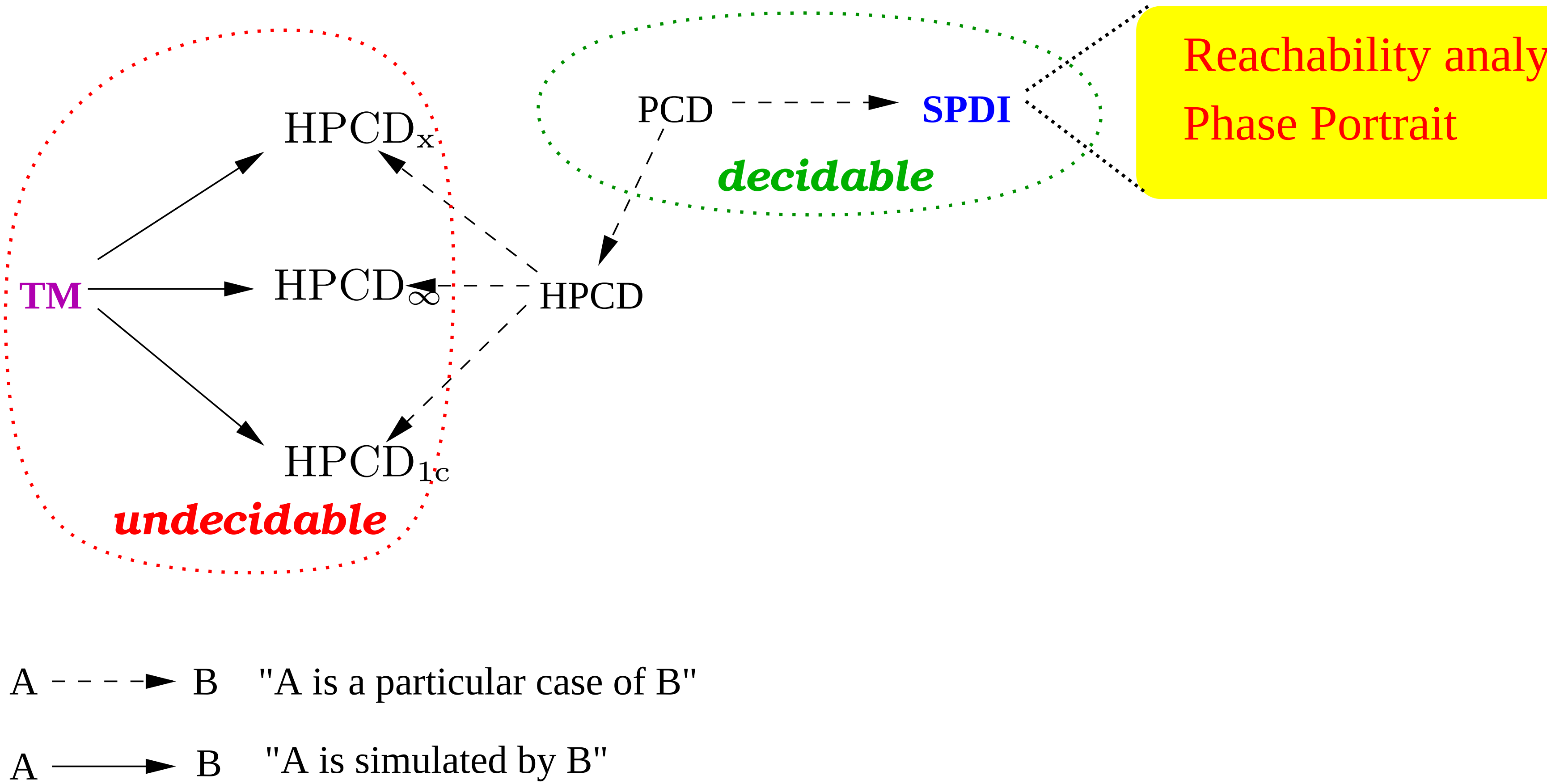
A - - - -► B "A is a particular case of B"

Summary of Results

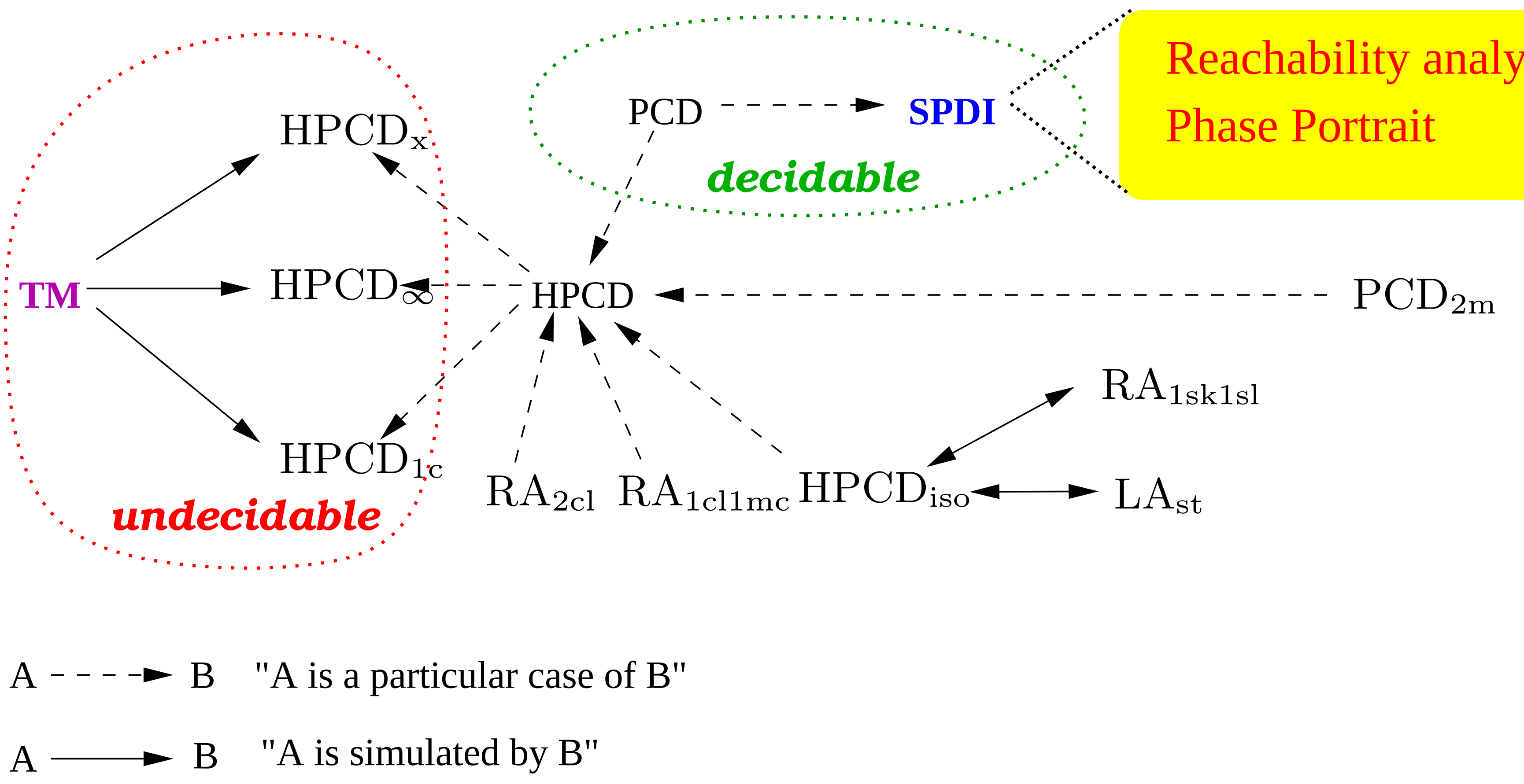


A - - - -► B "A is a particular case of B"

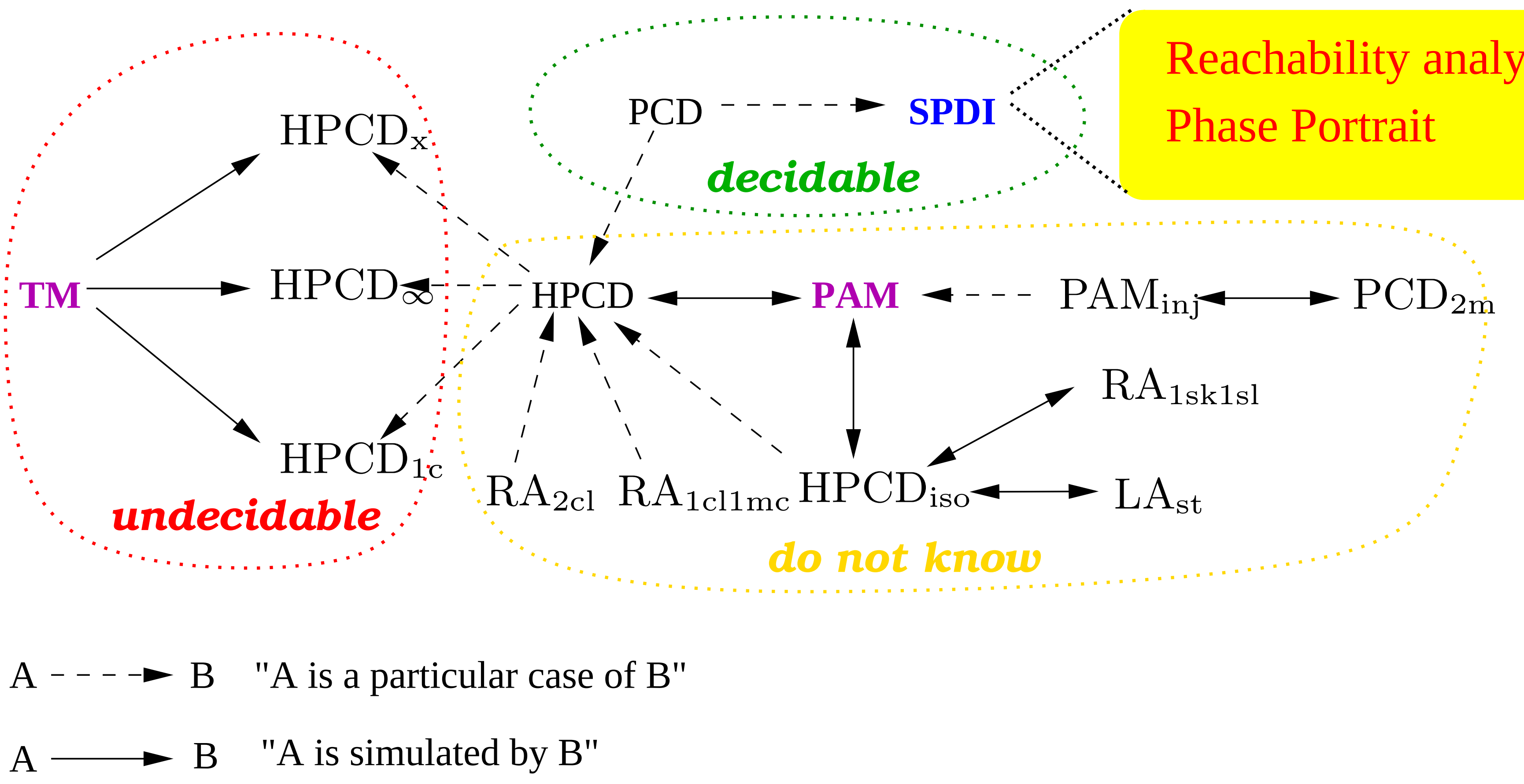
Summary of Results



Summary of Results



Summary of Results



Perspectives

- SPDI to approximate non-linear differential equations
- Conditions for decidability of PCDs on 2-dim manifolds
- Application of the *geometric* method to higher dimensions
- Extension of SPeeDI: algorithm for viability and controllability kernels
- SPeeDI: “Topological” optimizations

Merci!

Gracias!

Obrigado!

(Brasil Penta-Campeão!)

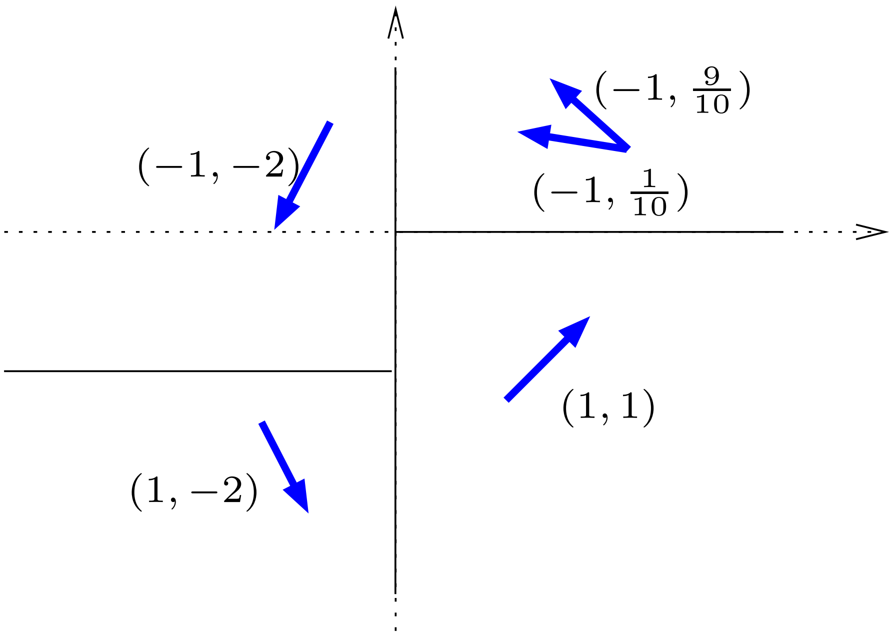
Thank you!

Theorem de Poincaré-Bendixson

A non-empty compact limit set of C^1 planar dynamical system that contains no equilibrium points is a close orbit.

Comparison with HyTech

Example:



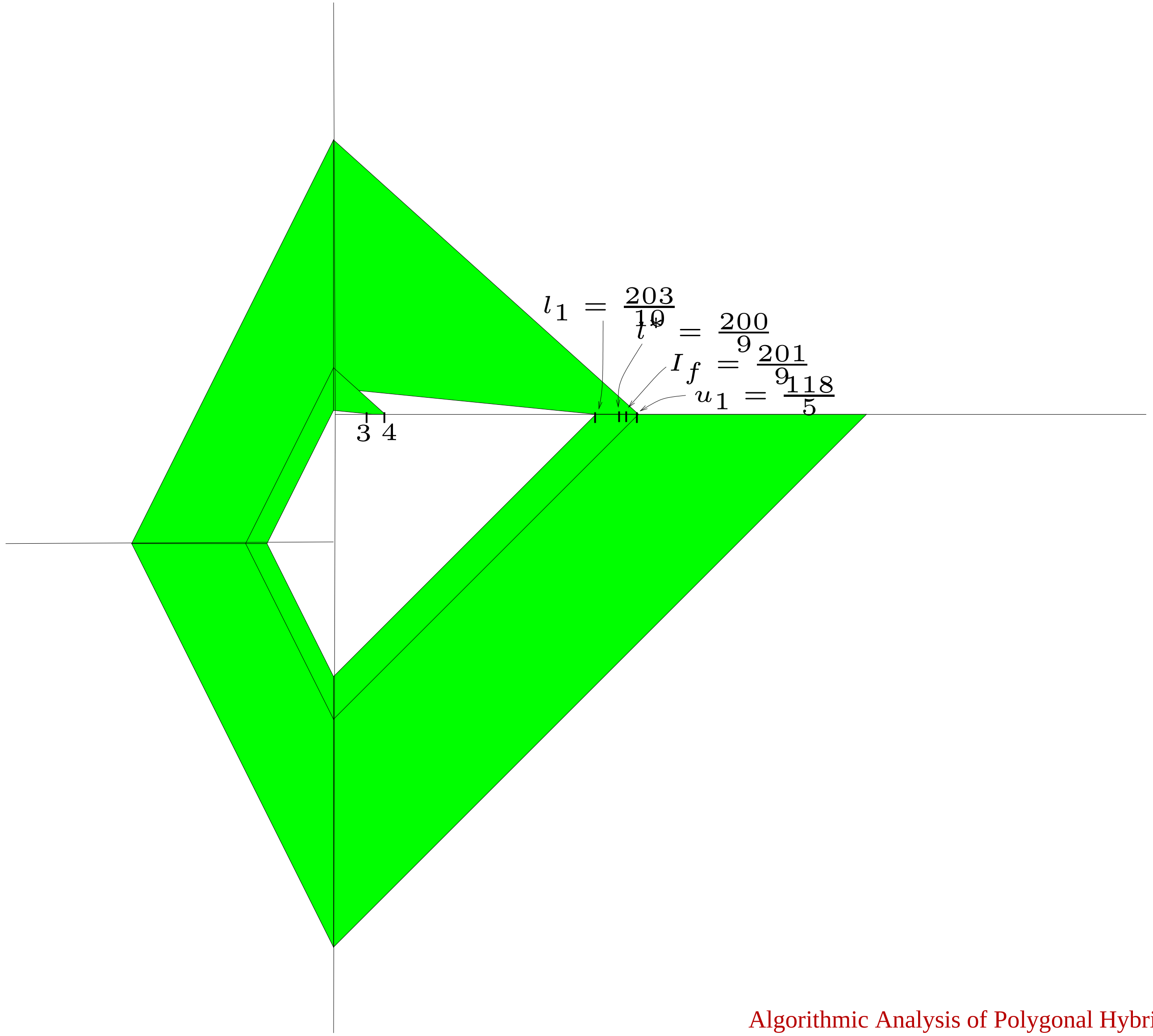
Fixpoint: $I^* = (\frac{200}{9}; 200)$

Comparison with HyTech

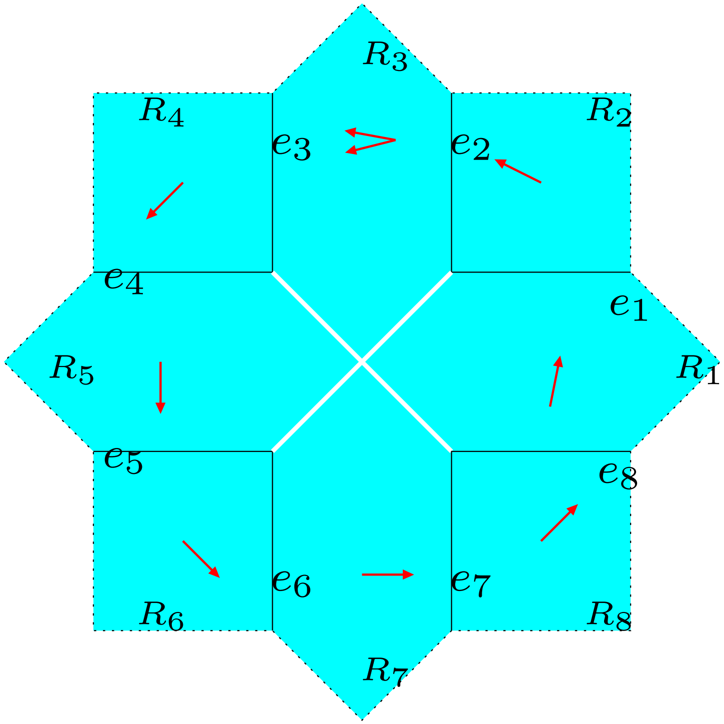
Final Point	*HyTech*	*SPeeDI*	*Reachable*
199	`overflow`	`0.05 sec`	Yes
200	`overflow`	`0.05 sec`	No
201	`overflow`	`0.01 sec`	No
210	`overflow`	`0.05 sec`	No
5	`0.04 sec`	`0.05 sec`	No
20	`0.07 sec`	`0.05 sec`	No
$\frac{200}{9}$	`0.10 sec`	`0.05 sec`	Yes
$\frac{201}{9}$	`overflow`	`0.03 sec`	Yes
$\frac{199}{9}$	`0.07 sec`	`0.04 sec`	Yes

Comparison with HyTech

Simulation of reachability for $x_f = \frac{201}{9}$



K_σ



Composition of TAMFs

TAMFs are closed under composition:
For

$$\mathcal{F}_1(x) = F_1(\{x\} \cap S_1) \cap J_1$$

and

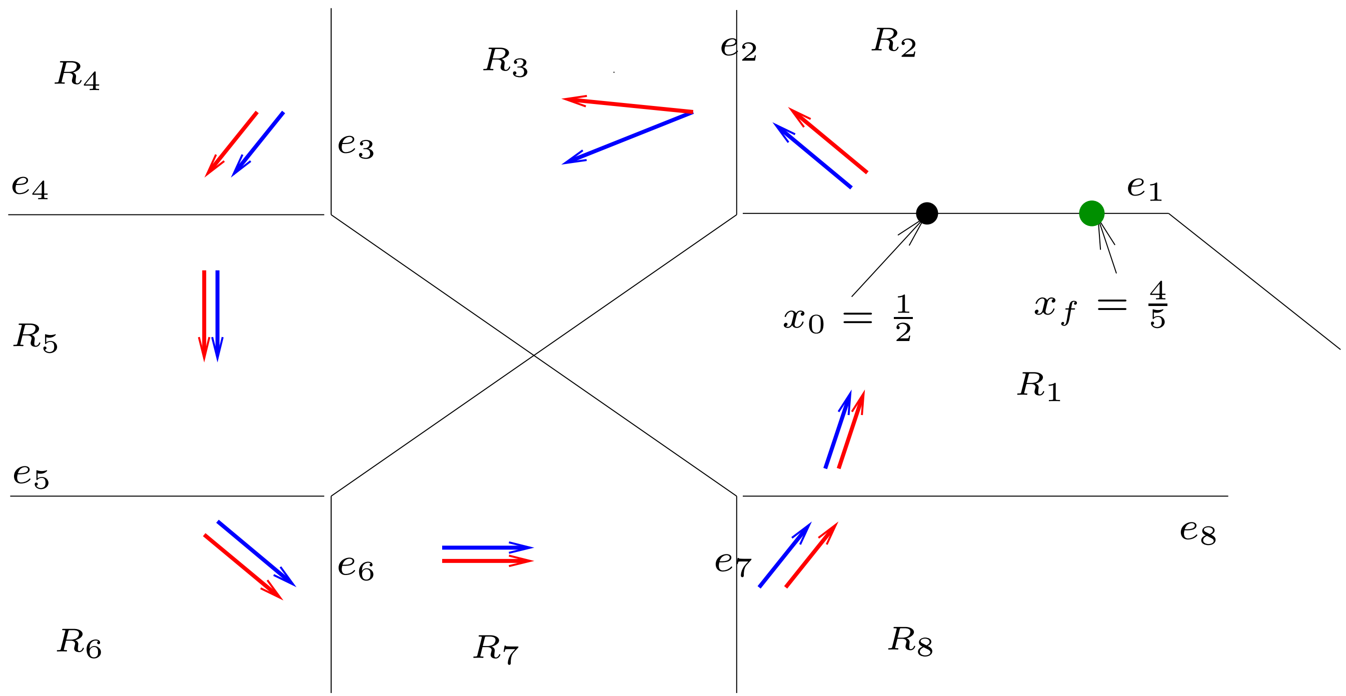
$$\mathcal{F}_2(x) = F_2(\{x\} \cap S_2) \cap J_2$$

we have that

$$\mathcal{F}_2 \circ \mathcal{F}_1(x) = \mathcal{F}_{F',S',J'}(x)$$

with
$F' = F_2 \circ F_1$,
$J' = J_2 \cap F_2(J_1 \cap S_2)$ and
$S' = S_1 \cap F_1^{-1}(J_1 \cap S_2)$

Reachability Algorithm (Example)

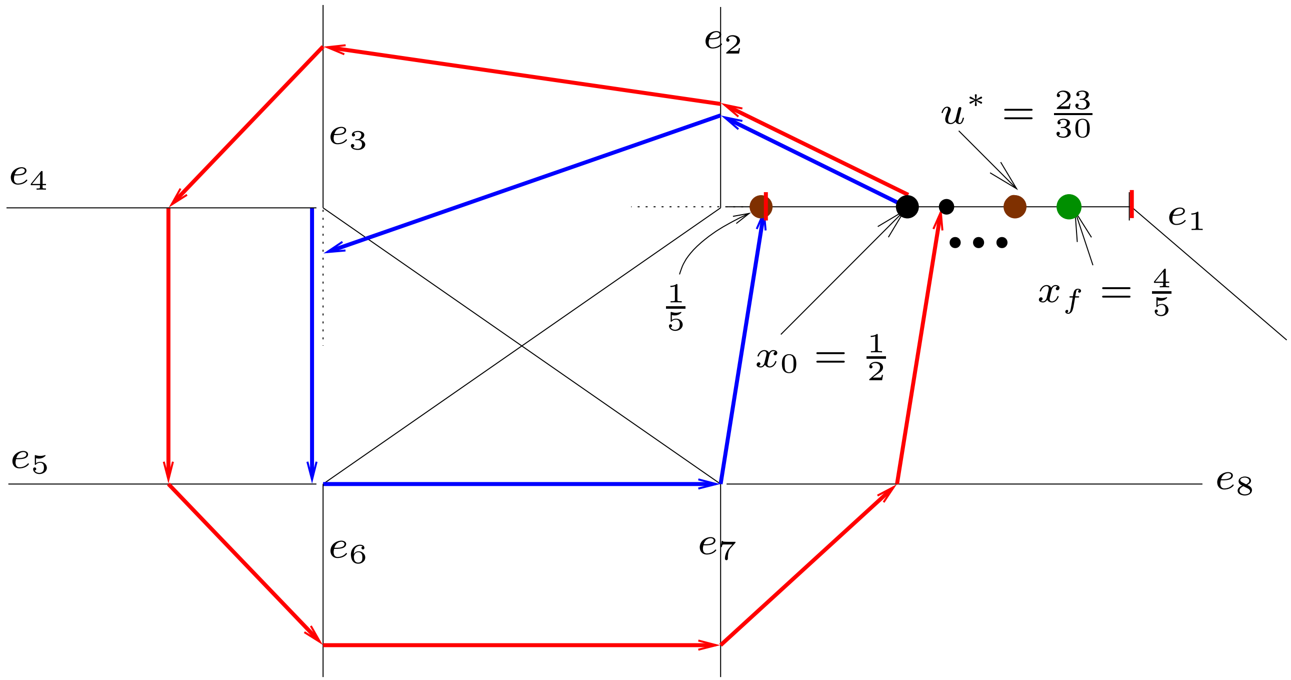


- Type of signature: $\sigma = (e_1 \cdots e_8)^*$
- Successor for the loop $s = e_1 \dots e_8$:

$$\mathsf{Succ}_s(l, u) = [\frac{l}{2} - \frac{1}{20}, \frac{u}{2} + \frac{23}{60}] \cap (\frac{1}{5}, 1)$$
$$\text{if } [l, u] \subseteq (0, 1)$$

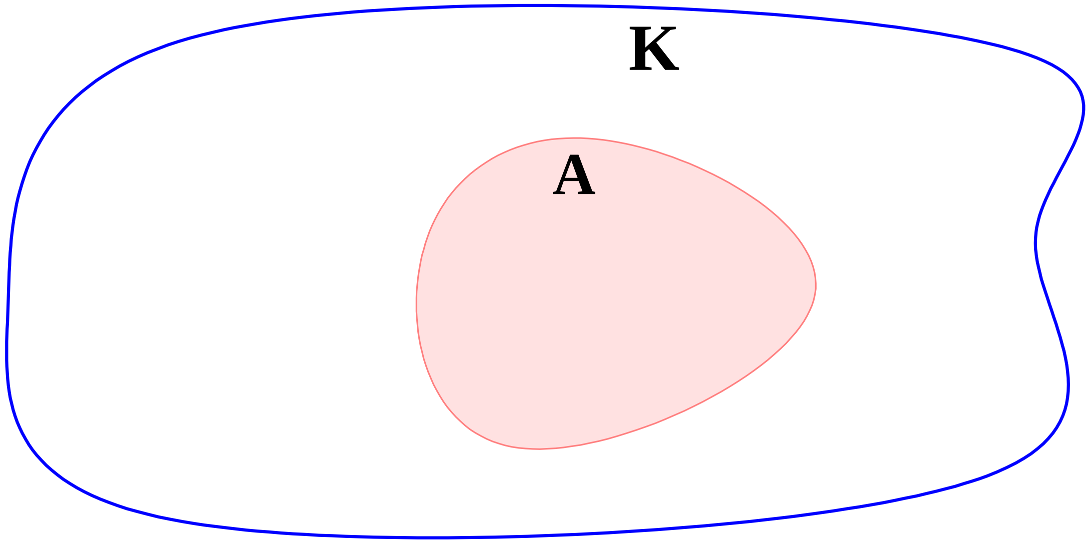
Reachability Algorithm (Example)

- Fixpoint equation: $\mathsf{Succ}_{e_1 \ldots e_8}(I^*) = I^*$
- Solution: $I^* = [l^*, u^*] = [\frac{1}{5}, \frac{23}{30}]$
- Hence: $\mathsf{Succ}_{e_1 \ldots e_8}(x_0) \subseteq [\frac{1}{5}, \frac{23}{30}]$

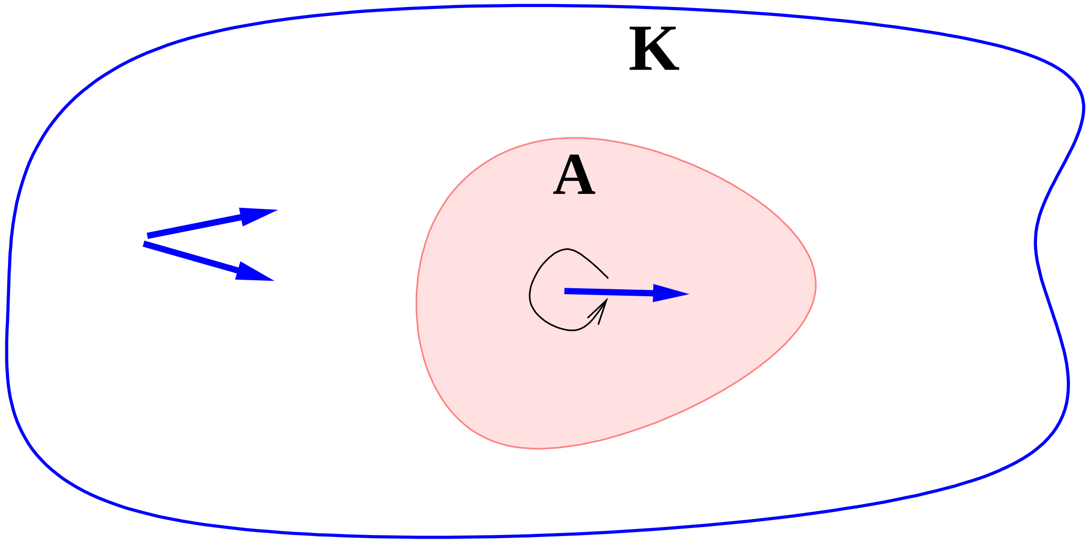


Conclusion: $x_f \notin [\frac{1}{5}, \frac{23}{30}]$).

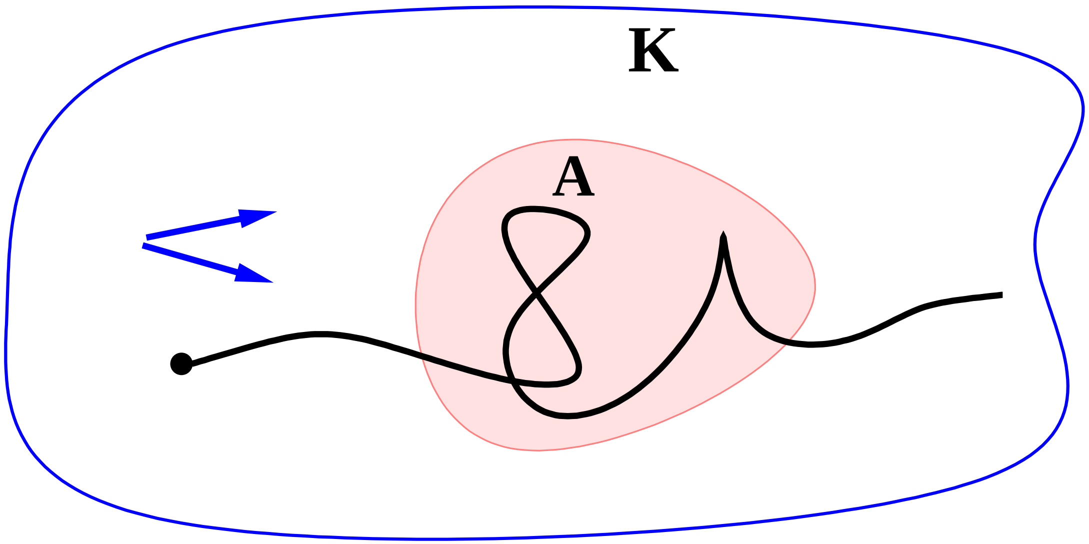
Viability Kernel



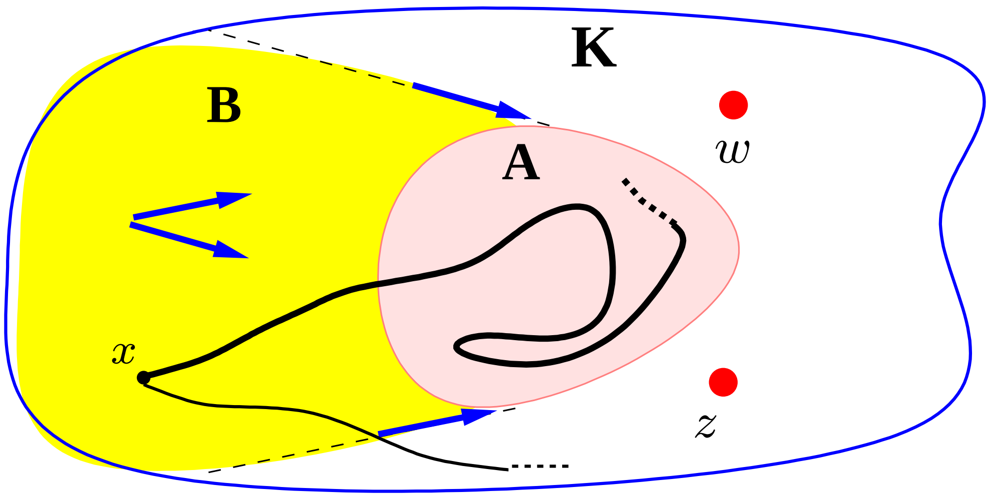
Viability Kernel



Viability Kernel



Viability Kernel



$$\mathsf{Viab}(K) = A \cup B$$

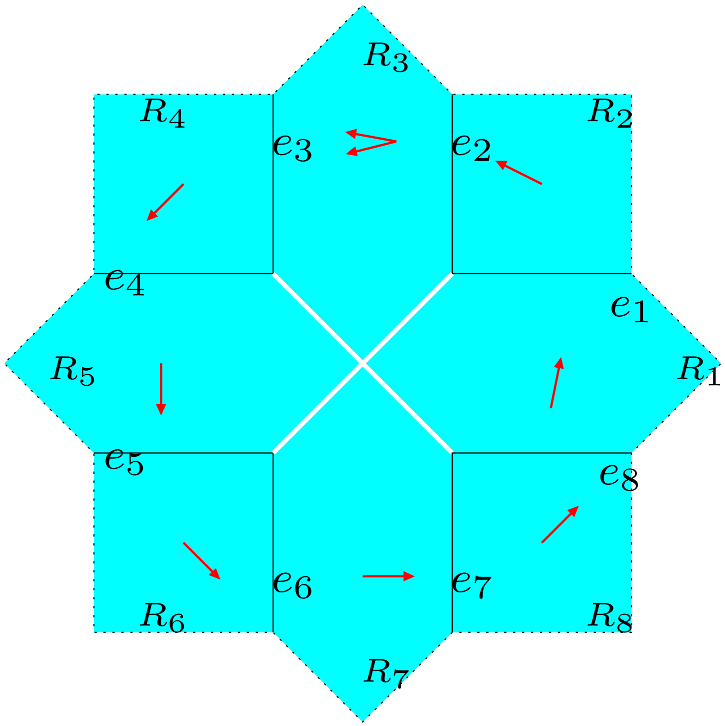
- M is a *viability domain* if $\forall \mathbf{x} \in M$, $\exists$ at least one trajectory ξ, starting in $\mathbf{x}$ and remaining in M
- $\mathsf{Viab}(K)$: *Viability kernel* of K is the largest viability domain M contained in K

Viability Kernel for SPDIs

- We can easily compute the Viability Kernel for one cycle, which is a polygon

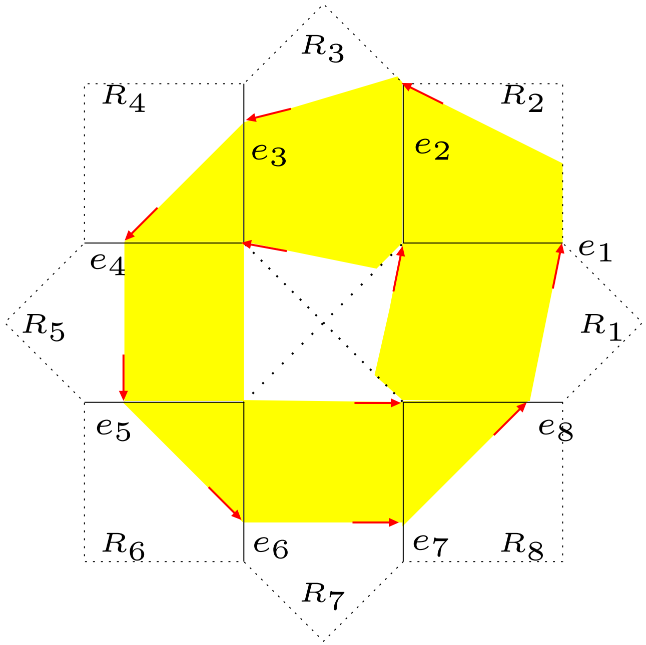
Viability Kernel for SPDIs

- We can easily compute the Viability Kernel for one cycle, which is a polygon



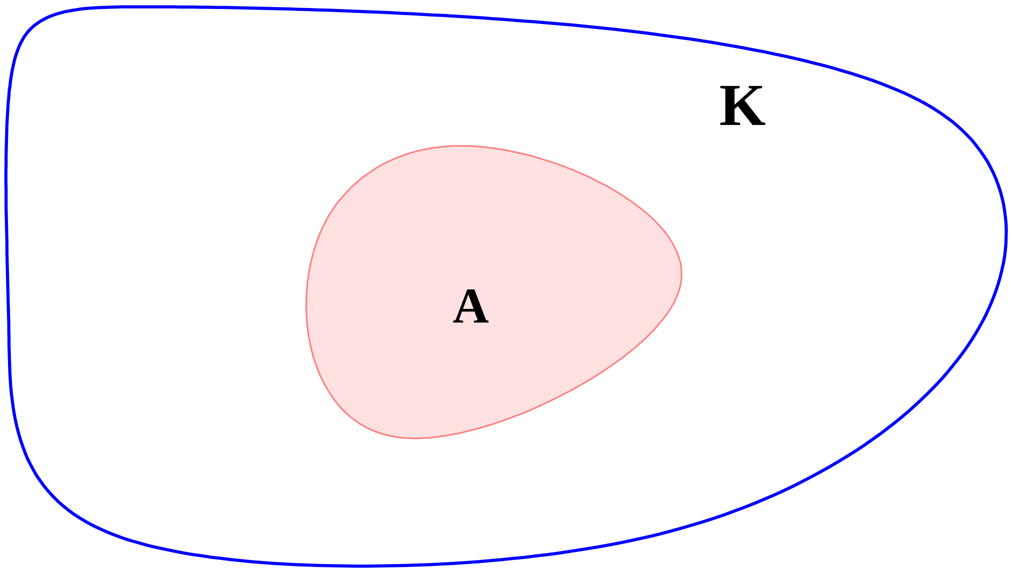
Viability Kernel for SPDIs

- We can easily compute the Viability Kernel for one cycle, which is a polygon

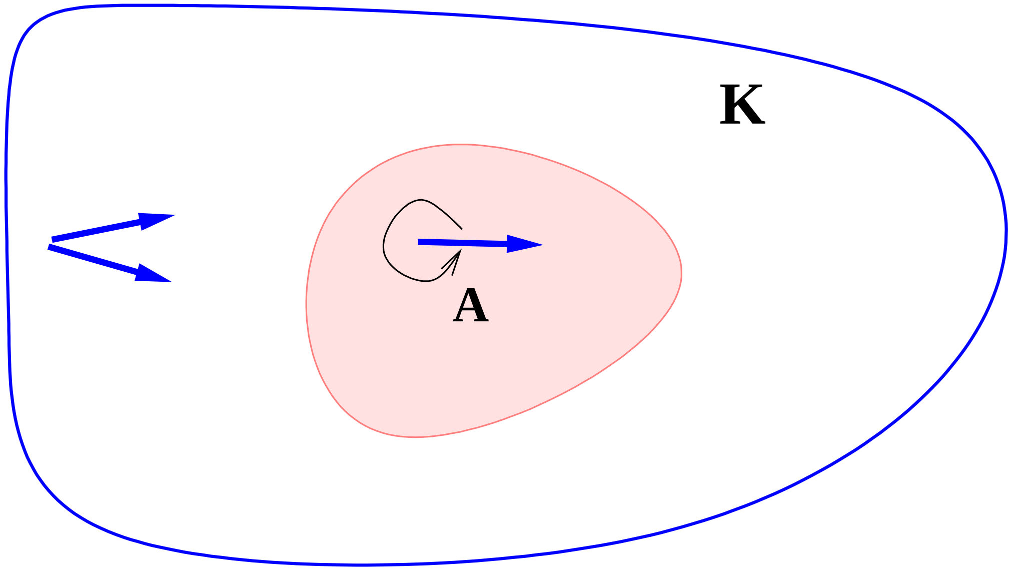


- **Theorem:** $\mathsf{Viab}(K_\sigma) = \overline{\mathsf{Pre}}_\sigma(\mathsf{Dom}(\mathsf{Succ}_\sigma))$

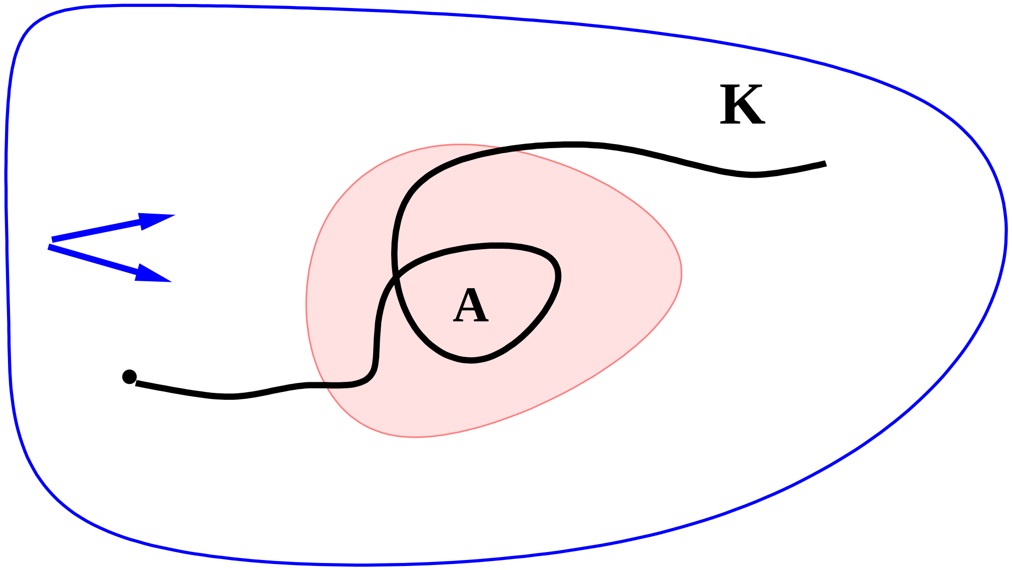
Controllability Kernel



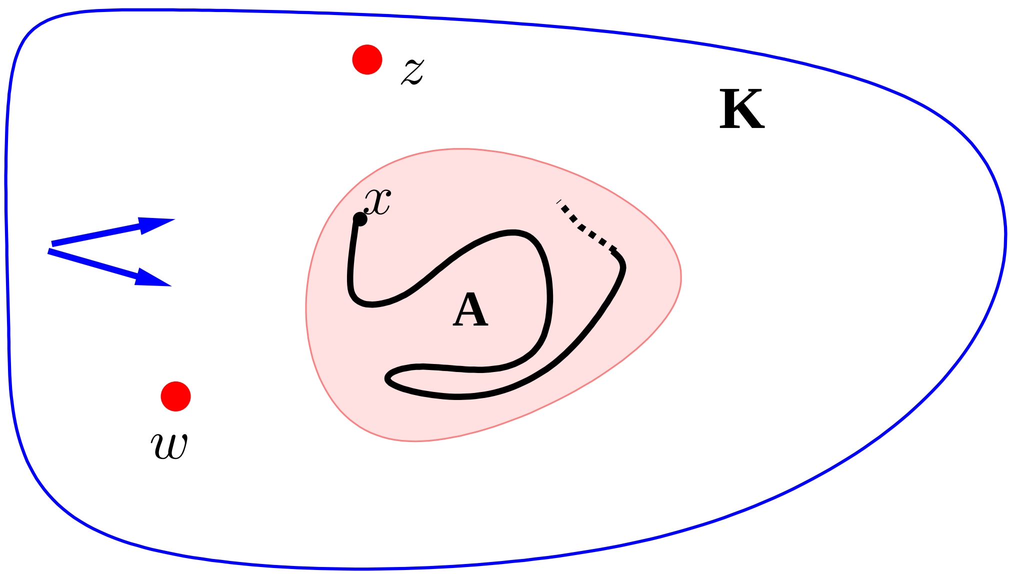
Controllability Kernel



Controllability Kernel



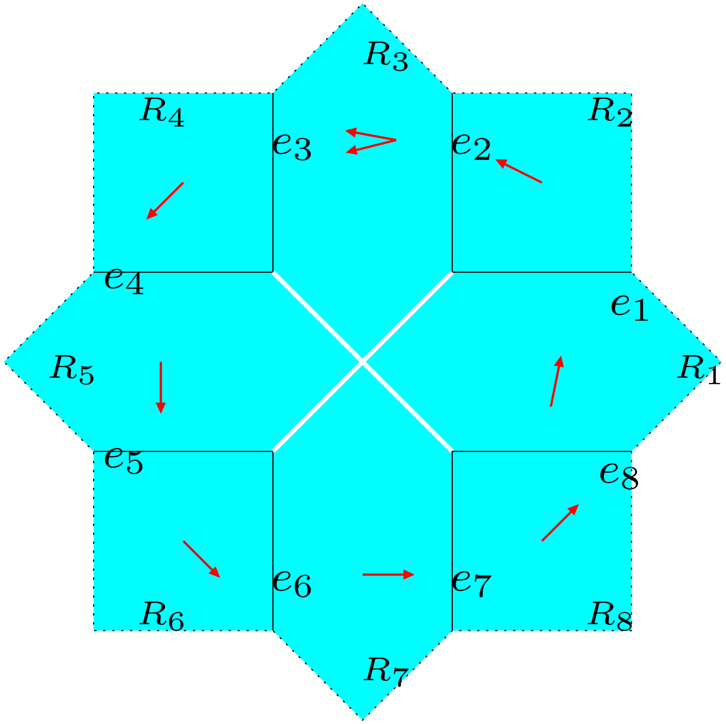
Controllability Kernel



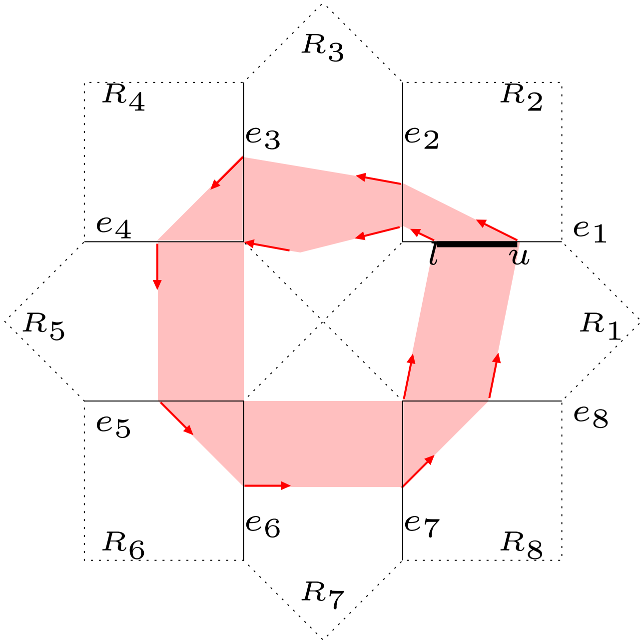
$$\mathsf{Cntr}(K) = A$$

- M is *controllable* if $\forall \mathbf{x}, \mathbf{y} \in M$, $\exists$ a trajectory segment ξ starting in $\mathbf{x}$ that reaches an arbitrarily small neighborhood of $\mathbf{y}$ without leaving M
- *Controllability kernel* of K, denoted $\mathsf{Cntr}(K)$, is the largest controllable subset of K

Controllability Kernel for SPDIs

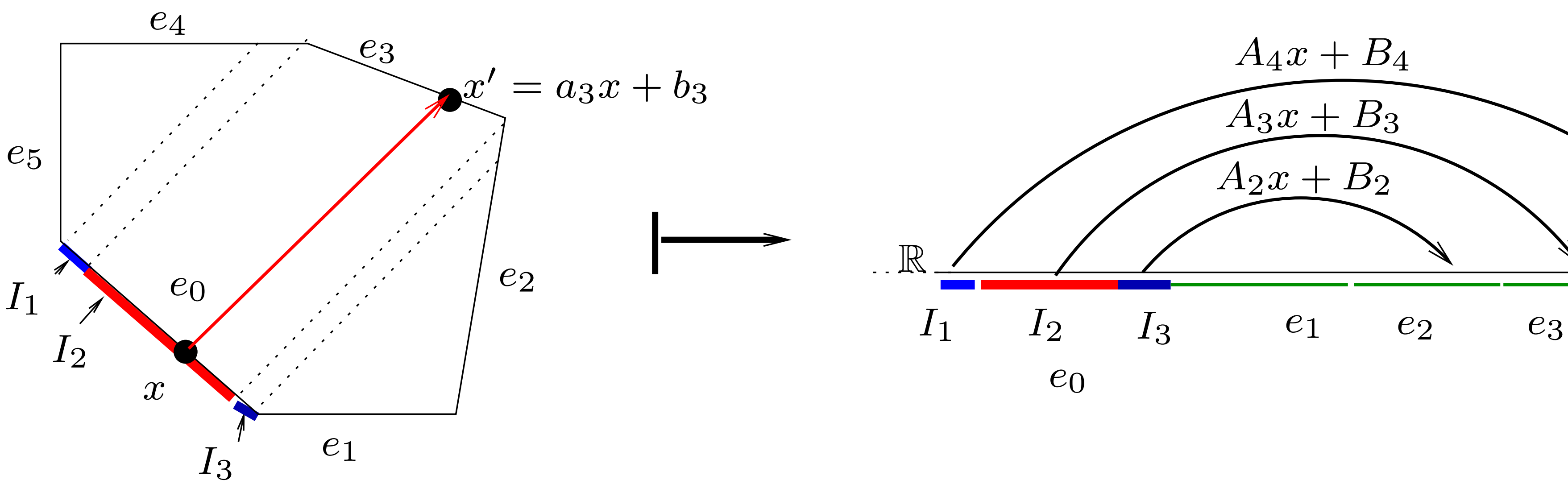


Controllability Kernel for SPDIs

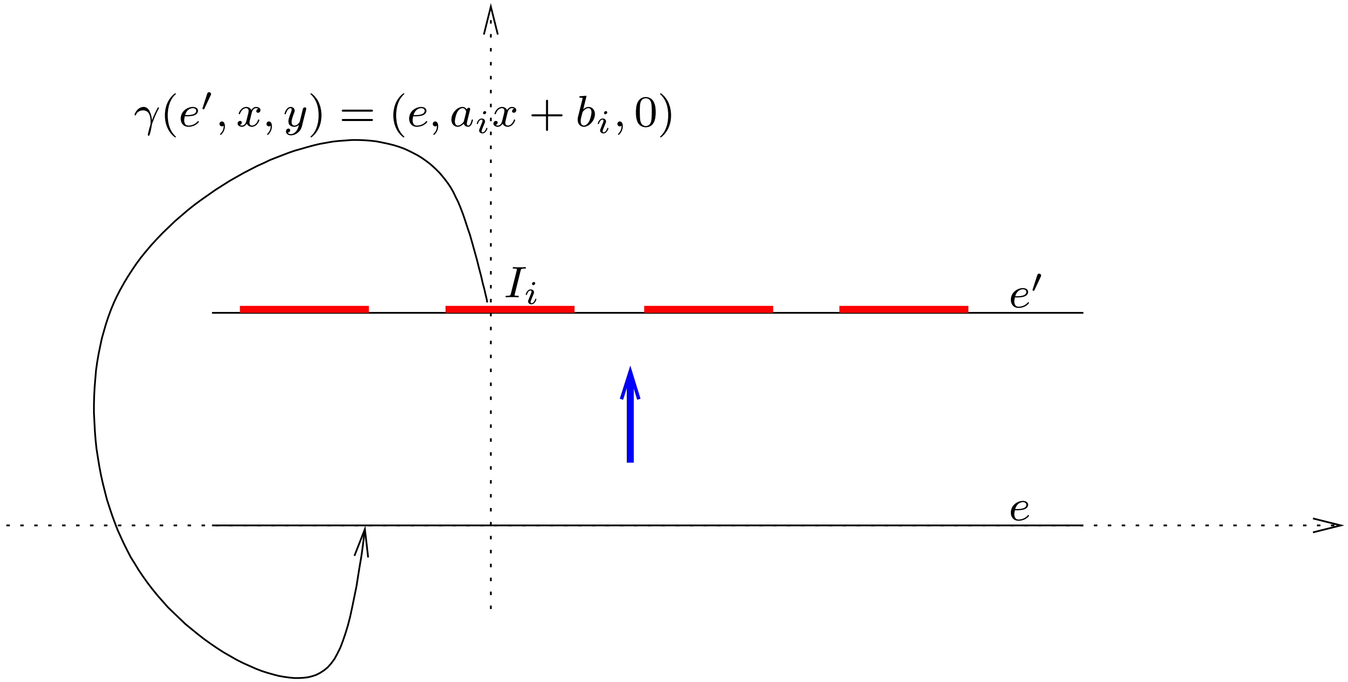


- **Theorem:** $\mathsf{Cntr}(K_\sigma) = (\overline{\mathsf{Succ}}_\sigma \cap \overline{\mathsf{Pre}}_\sigma)(\mathcal{C}_\mathcal{D}(\sigma))$
  (We know how to compute the special interval $\mathcal{C}_\mathcal{D}(\sigma) = [l, u]$)

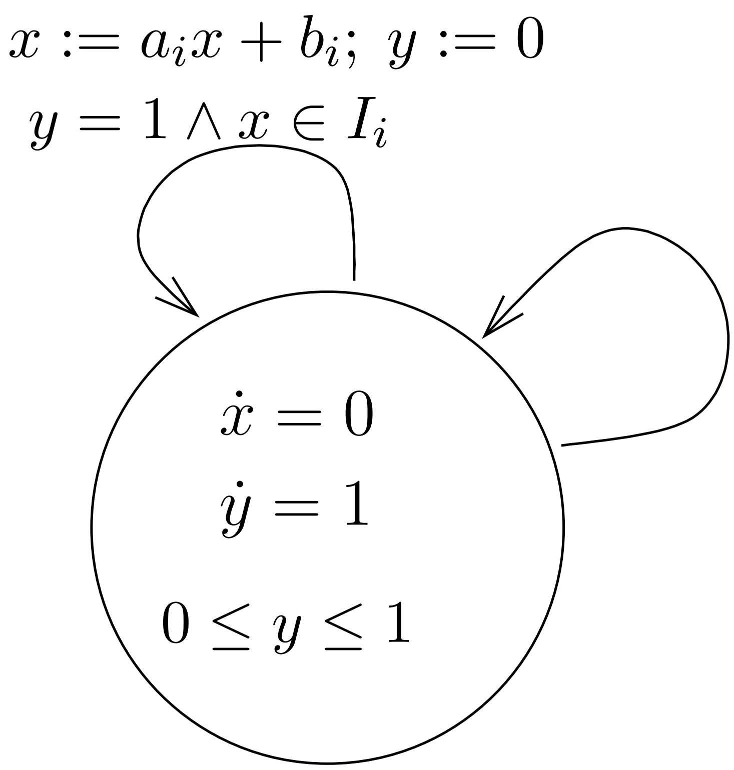
PAM simulate HPCD



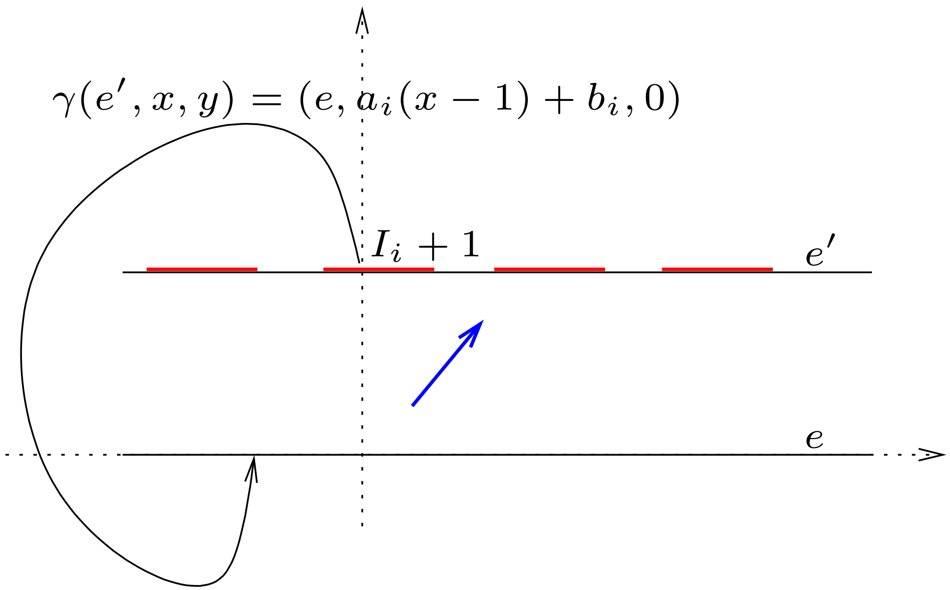
HPCD simulate PAM

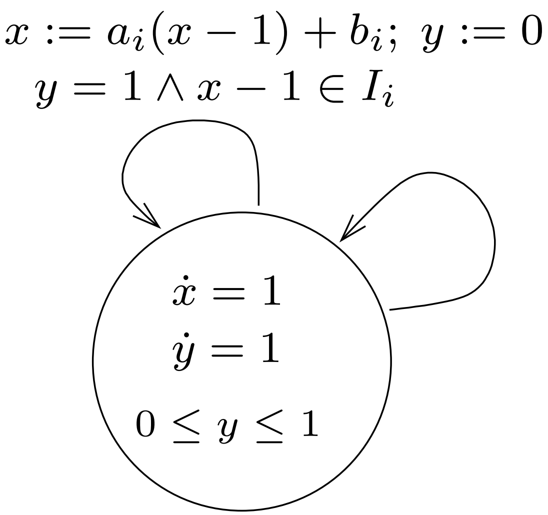


$\mathrm{RA}_{\mathrm{1cl1mc}}$ **equivalent to PAM**

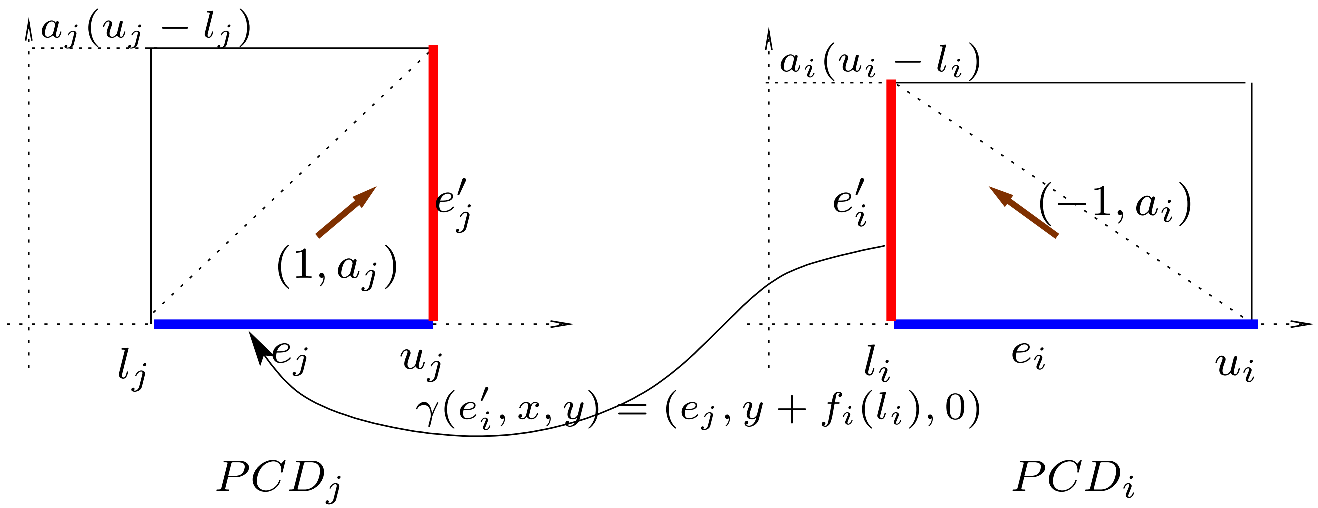


$\mathrm{RA}_{2\mathrm{cl}}$ **equivalent to PAM**

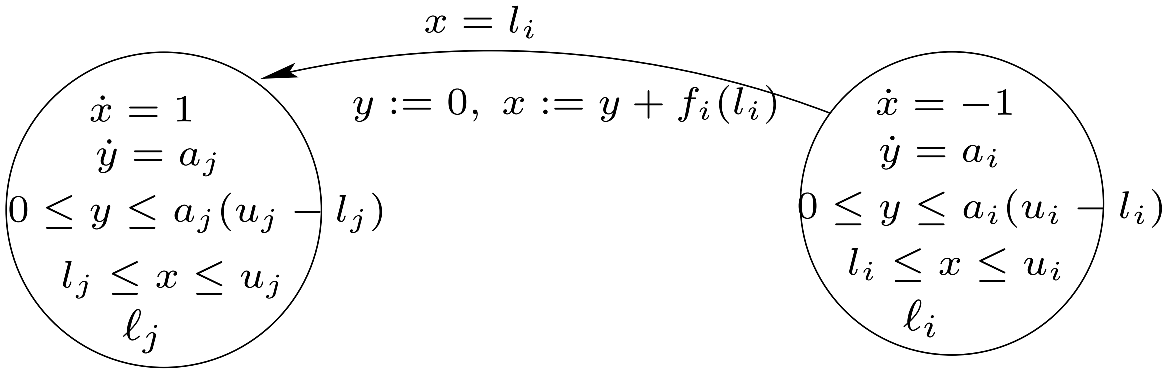




RA_{1sk1sl} **equivalent to PAM**

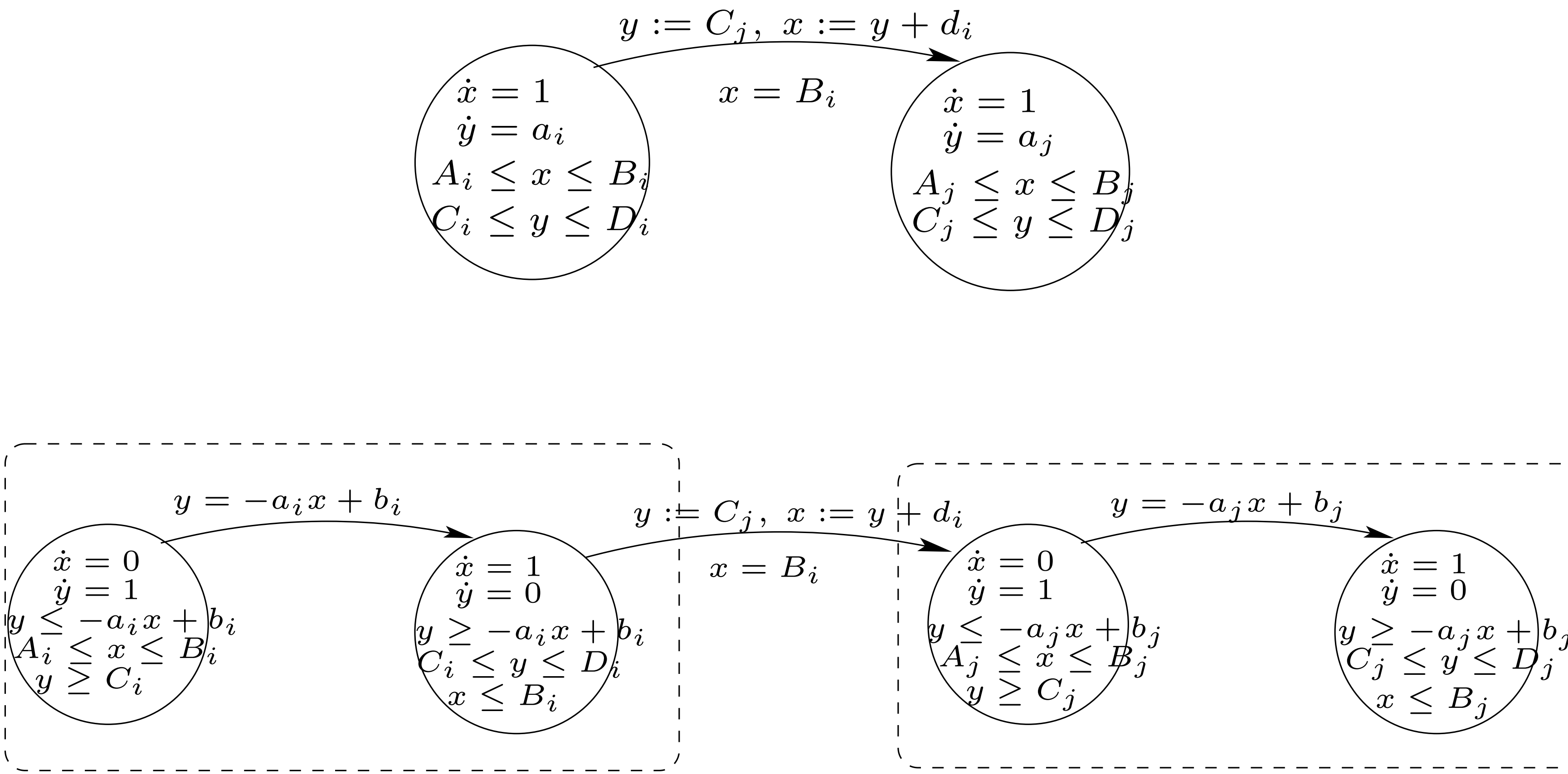


(a)



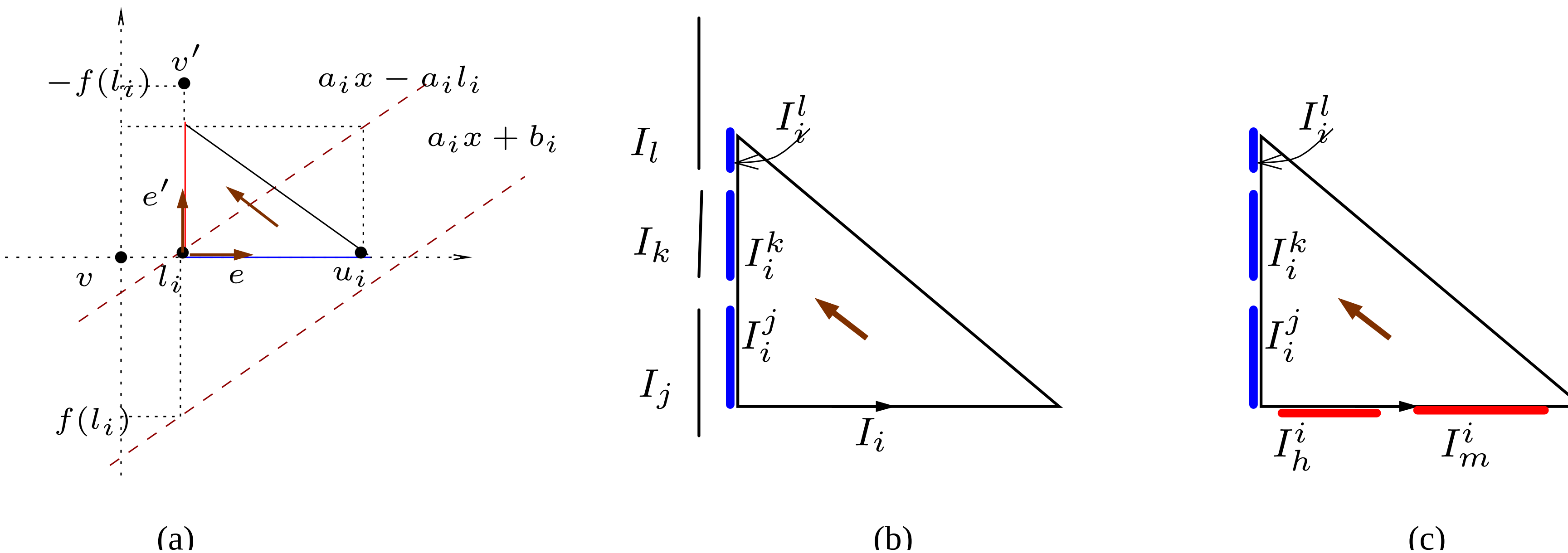
(b)

From $\mathrm{RA}_{1\mathrm{sk}1\mathrm{sl}}$ to $\mathrm{LA}_{\mathrm{st}}$

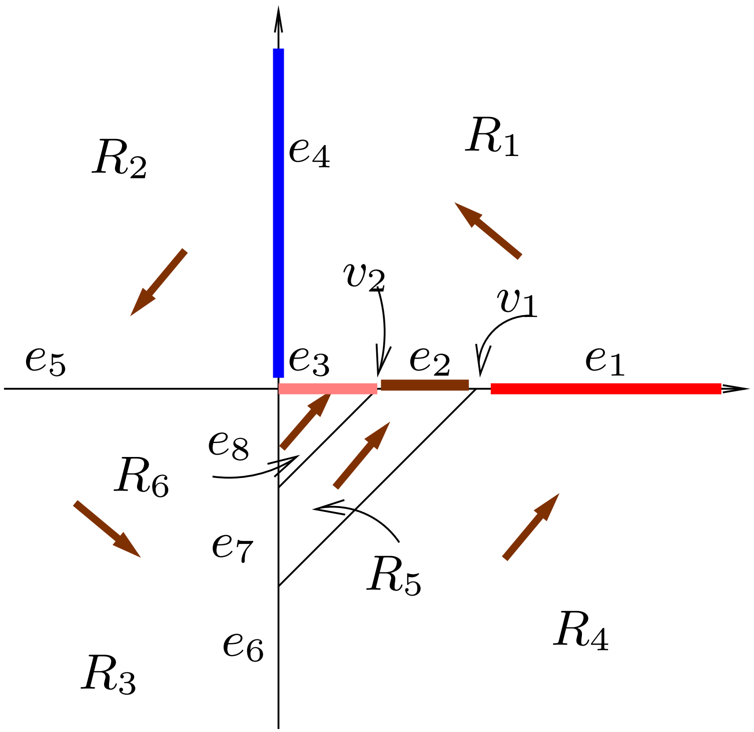


where $b_i = C_i + B_i a_i$ and $b_j = C_j + B_j a_j$

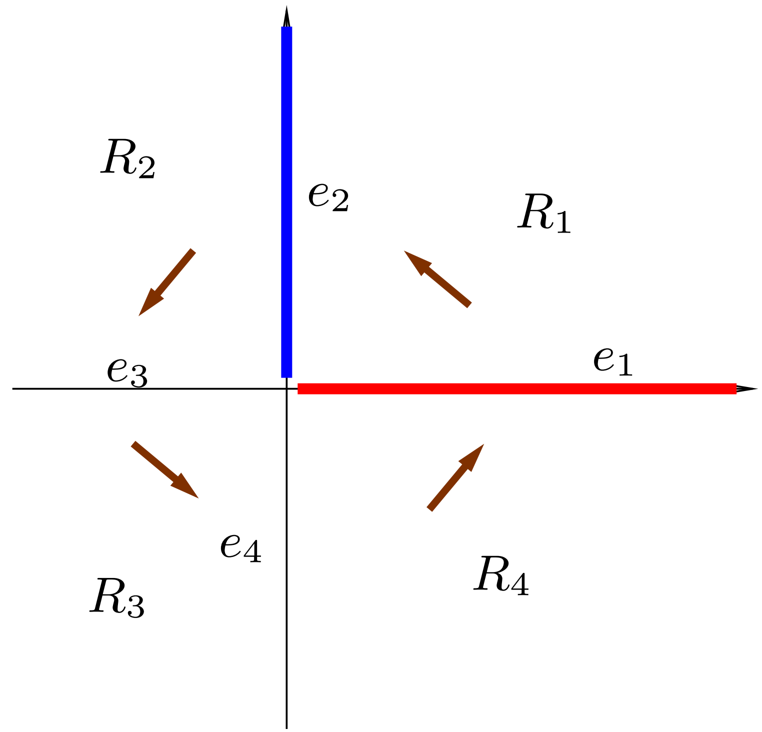
PCD_{2m} **simulate** PAM_{inj}



HPCD_{1c} **simulate TM**



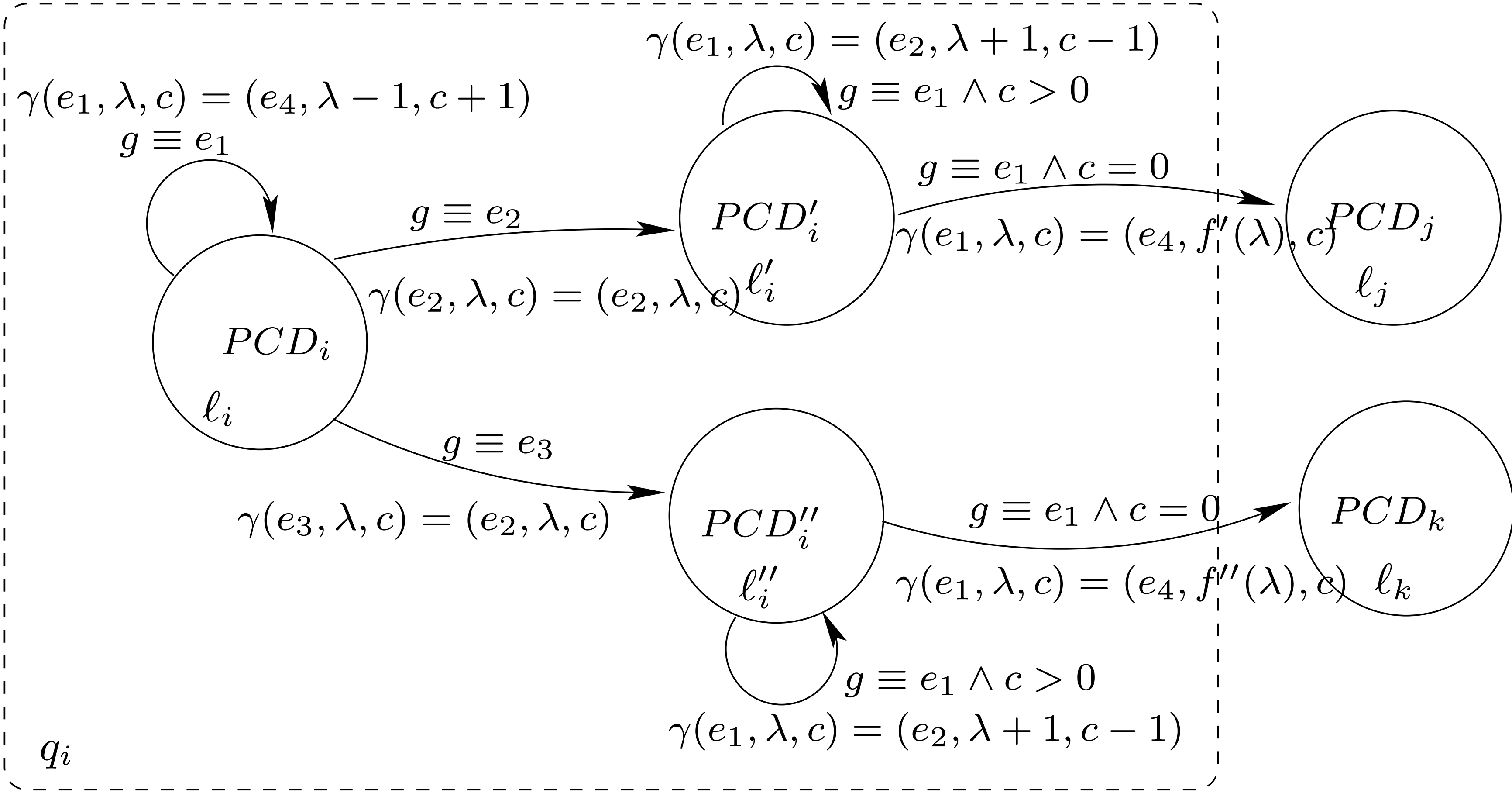
PCD_i



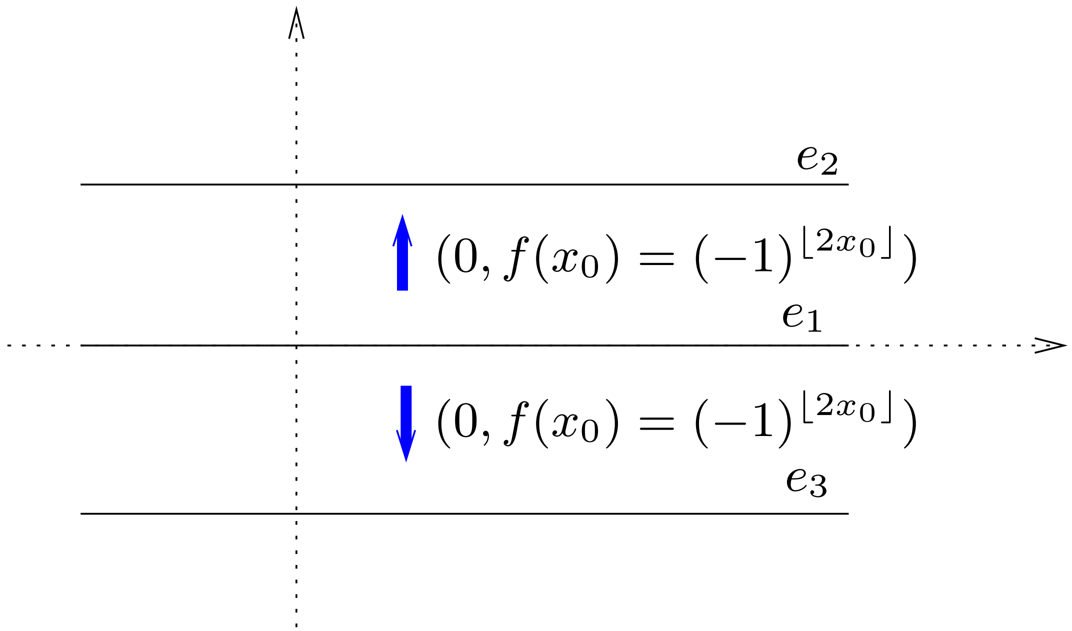
$PCD_i'; PCD_i''$

HPCD_{1c} **simulate TM**

TM-state q_i:



HPCD$_x$ simulate TM



$$f(x) = \begin{cases} 1 & \text{if } \mathrm{frac}_x < \frac{1}{2} \\ -1 & \text{otherwise} \end{cases}$$