Efficient Bounded Reachability Computation for Rectangular Automata

Xin Chen[1] Erika Ábrahám[1] Goran Frehse[2]

[1]RWTH Aachen University, Germany

[2]Université Grenoble 1 Joseph Fourier - Verimag, France

RP 2011

Outline

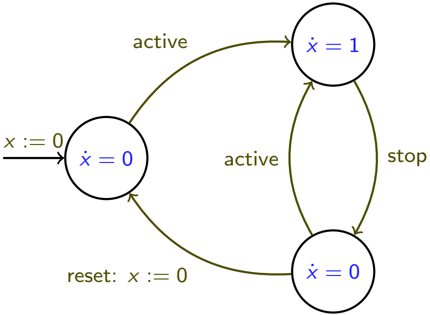
1. Reachability computation for rectangular automata
2. Compute reachable sets efficiently
3. Comparison with PHAVer
4. Future work

Outline

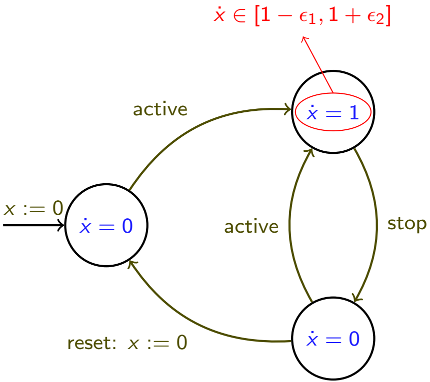
1. Reachability computation for rectangular automata
2. Compute reachable sets efficiently
3. Comparison with PHAVer
4. Future work

Example: a mechanical stopwatch

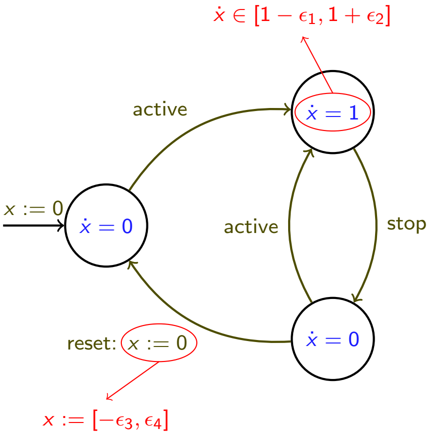
Example: a mechanical stopwatch



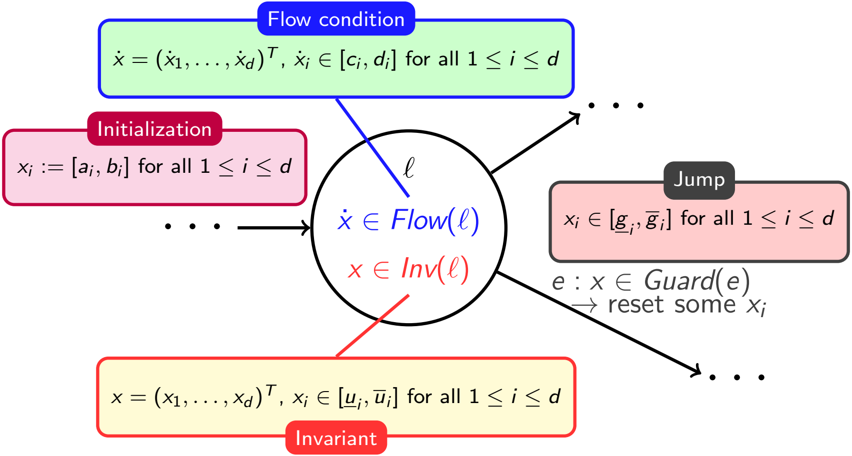
Example: a mechanical stopwatch



Example: a mechanical stopwatch

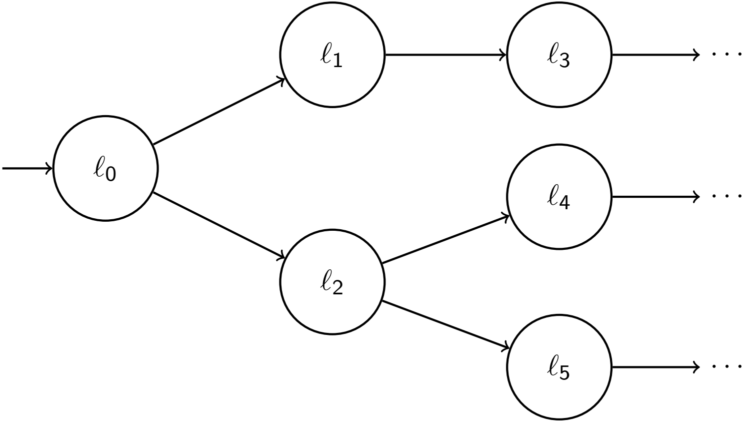


Rectangular automata

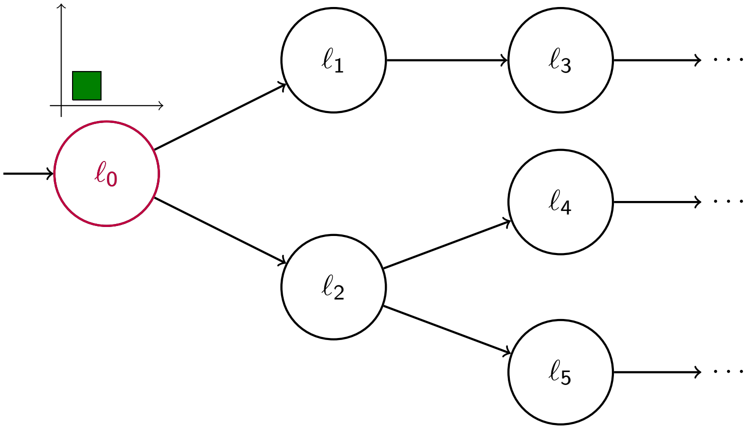


Henzinger et al. What's Decidable about Hybrid Automata? In JCSS (1998)

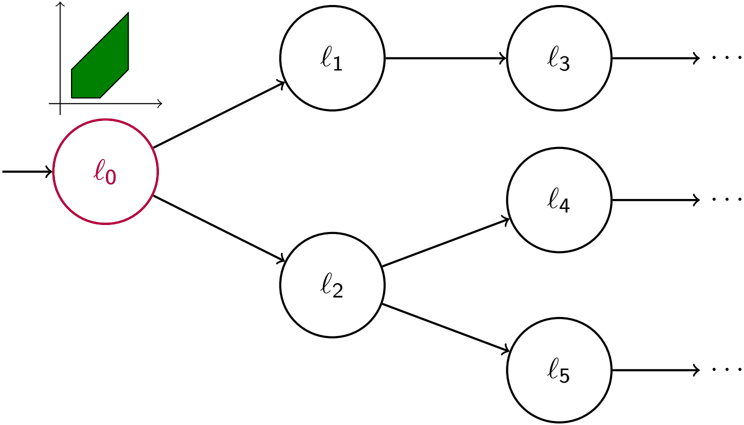
Bounded reachability computation



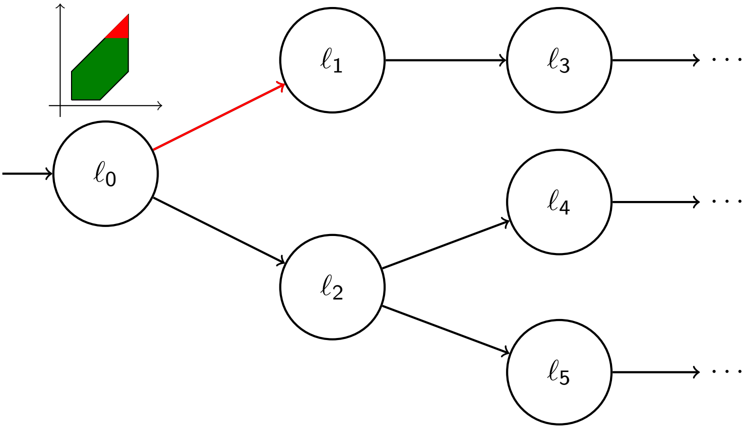
Bounded reachability computation



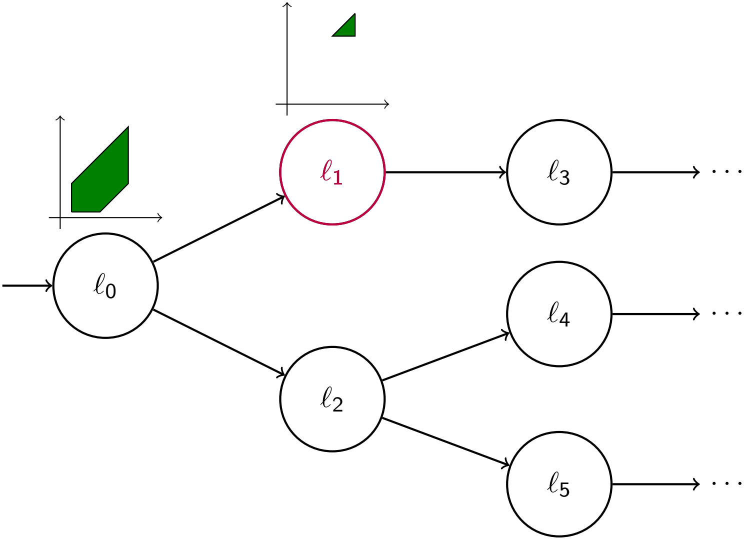
Bounded reachability computation



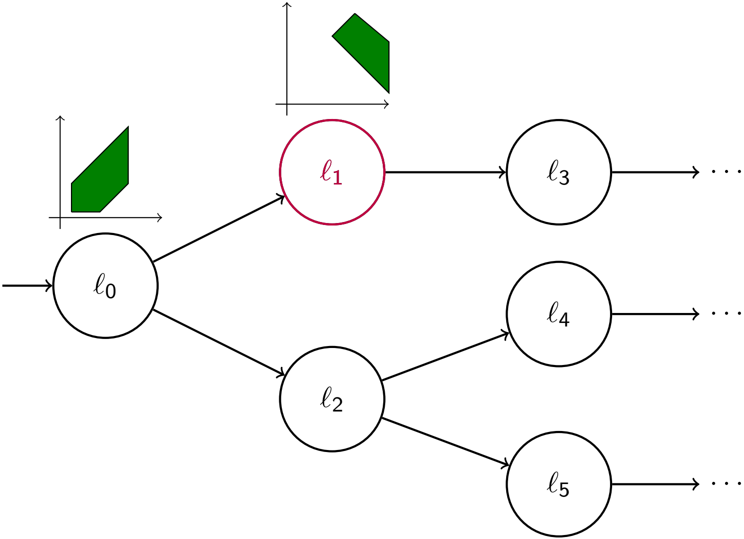
Bounded reachability computation



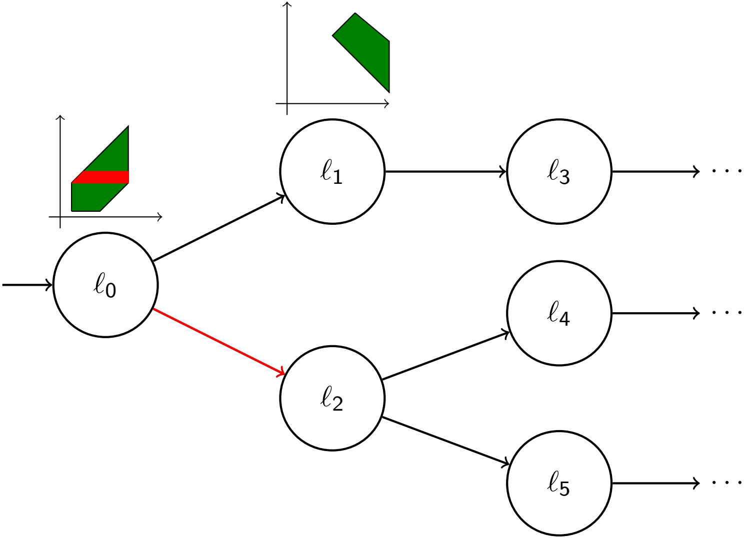
Bounded reachability computation



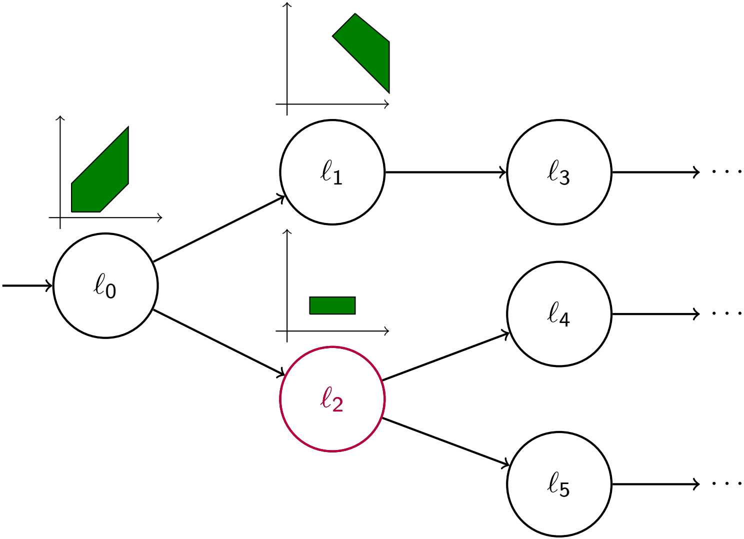
Bounded reachability computation



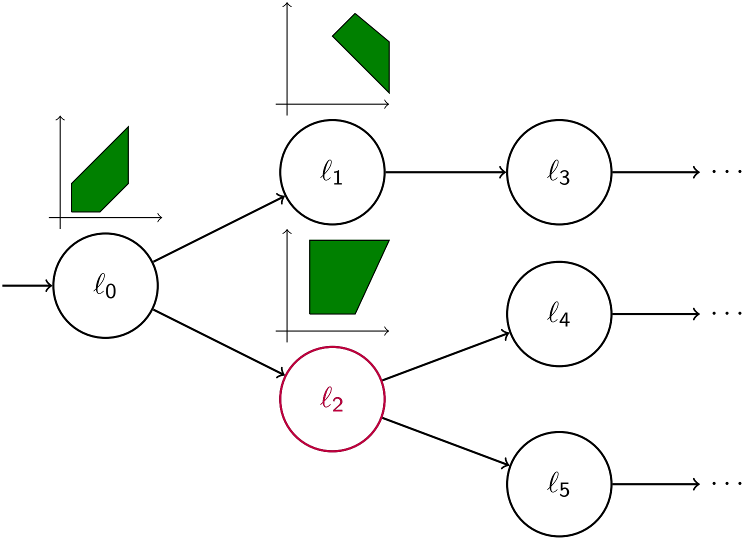
Bounded reachability computation



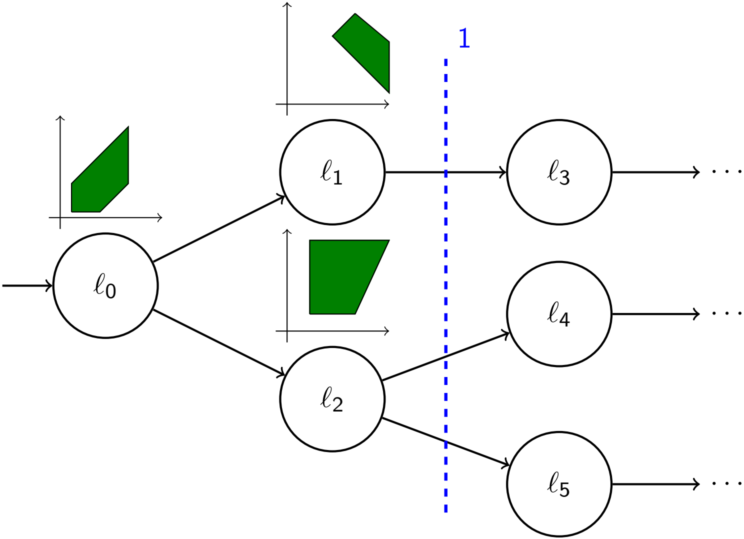
Bounded reachability computation



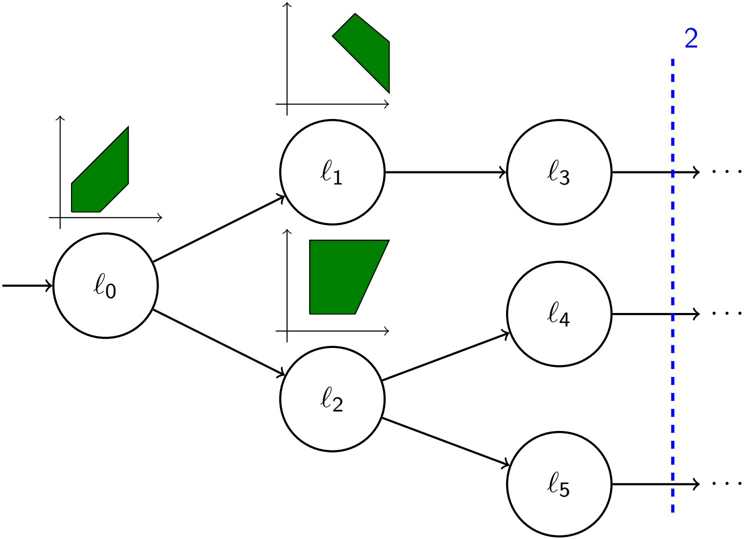
Bounded reachability computation



Bounded reachability computation



Bounded reachability computation



Representations for the state sets

Representations for the state sets

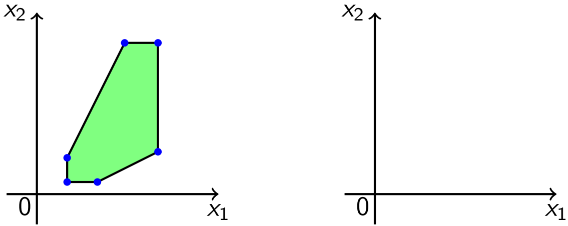
- Polyhedron for the reachable set under a flow condition.

Representations for the state sets

- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.

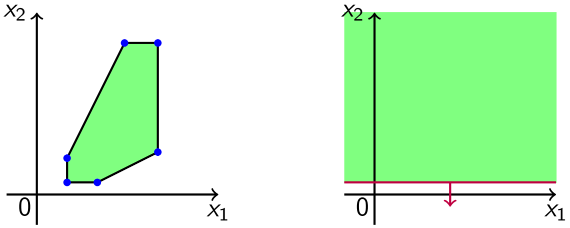
Representations for the state sets

- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



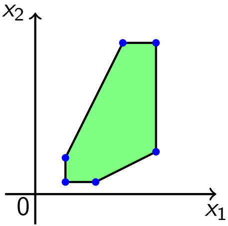
Representations for the state sets

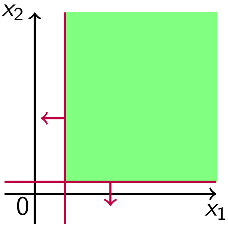
- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



Representations for the state sets

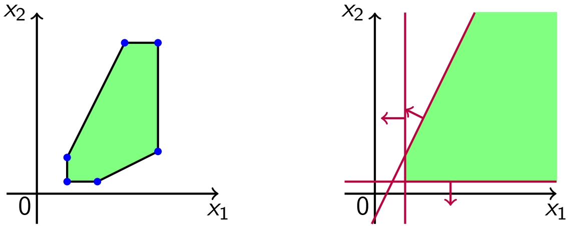
- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



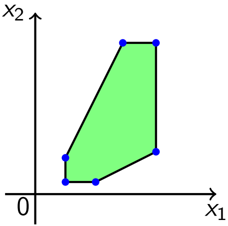


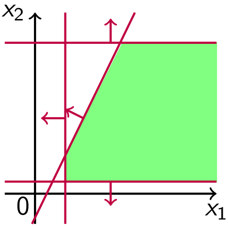
Representations for the state sets

- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:

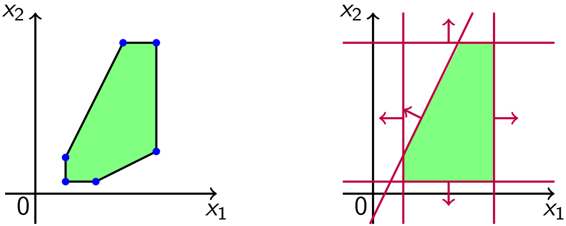


- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



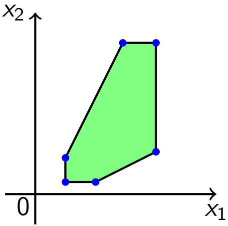


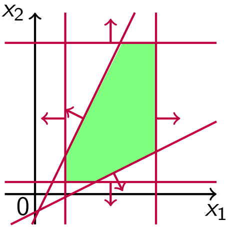
- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



Representations for the state sets

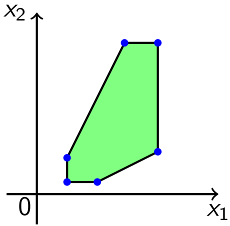
- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:

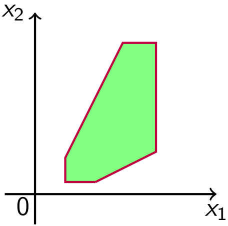




Representations for the state sets

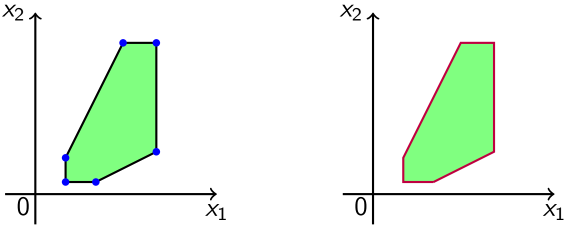
- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:





Representations for the state sets

- Polyhedron for the reachable set under a flow condition.
- Representations for polyhedra: vertex-based and constraint-based.
- Example:



- If $P : \mathcal{L}_P$ and $Q : \mathcal{L}_Q$, then $P \cap Q : \ \mathcal{L}_P \cup \mathcal{L}_Q$.

Facets and constraints

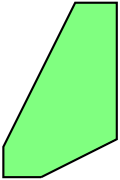
If $P \subseteq \mathbb{R}^d$ and $dim(P) = d'$, then

- facets: $(d' - 1)$-faces, vertices: 0-faces;
- there are $NF(P) + 2(d - d')$ constraints needed to define P where $NF(P)$ is the number of P's facets.

Facets and constraints

If $P \subseteq \mathbb{R}^d$ and $dim(P) = d'$, then

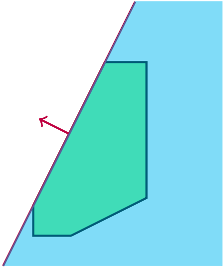
- facets: $(d' - 1)$-faces, vertices: 0-faces;
- there are $NF(P) + 2(d - d')$ constraints needed to define P where $NF(P)$ is the number of P's facets.



Facets and constraints

If $P \subseteq \mathbb{R}^d$ and $dim(P) = d'$, then

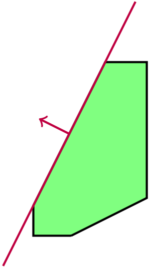
- facets: $(d' - 1)$-faces, vertices: 0-faces;
- there are $NF(P) + 2(d - d')$ constraints needed to define P where $NF(P)$ is the number of P's facets.



Facets and constraints

If $P \subseteq \mathbb{R}^d$ and $dim(P) = d'$, then

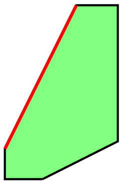
- facets: $(d' - 1)$-faces, vertices: 0-faces;
- there are $NF(P) + 2(d - d')$ constraints needed to define P where $NF(P)$ is the number of P's facets.

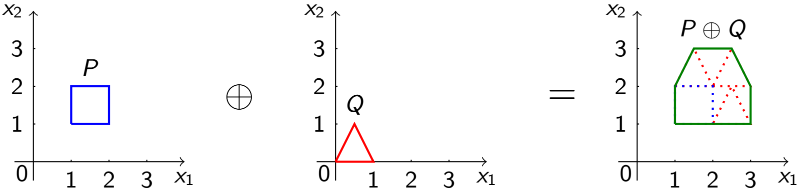


Facets and constraints

If $P \subseteq \mathbb{R}^d$ and $dim(P) = d'$, then

- facets: $(d' - 1)$-faces, vertices: 0-faces;
- there are $NF(P) + 2(d - d')$ constraints needed to define P where $NF(P)$ is the number of P's facets.

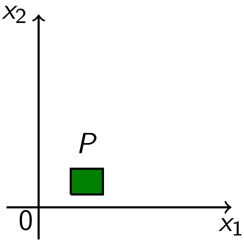




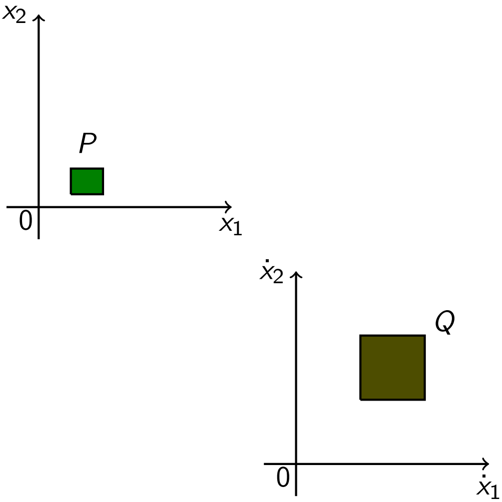
$$P \oplus Q = \{p + q \mid p \in P \text{ and } q \in Q\}$$

Reachable sets under flow transitions

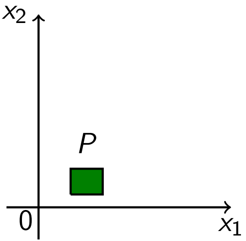
Reachable sets under flow transitions

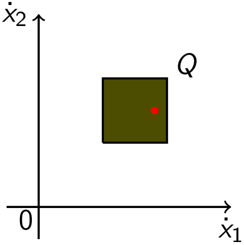


Reachable sets under flow transitions

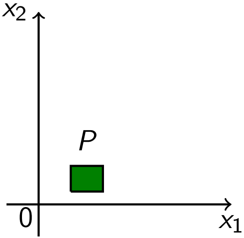


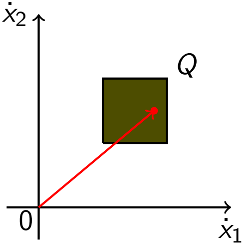
Reachable sets under flow transitions



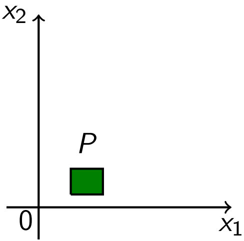


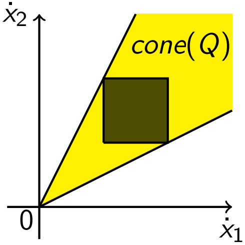
Reachable sets under flow transitions



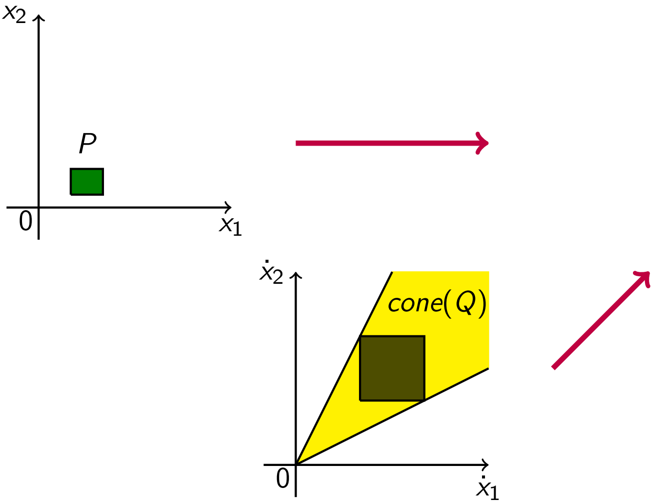


Reachable sets under flow transitions

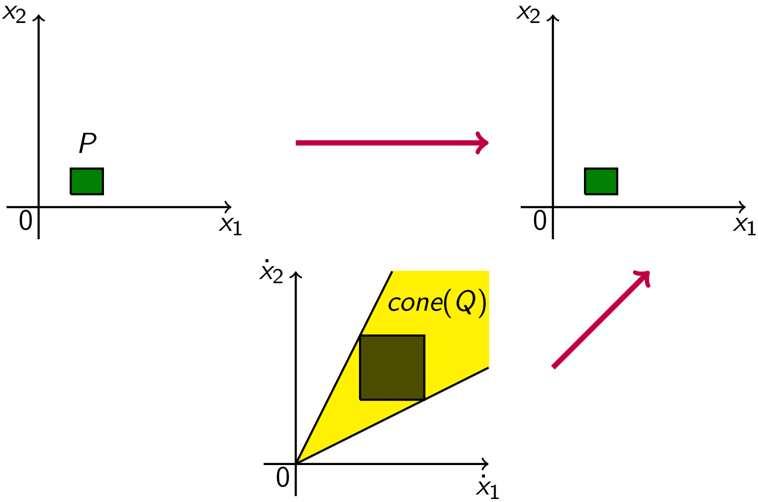




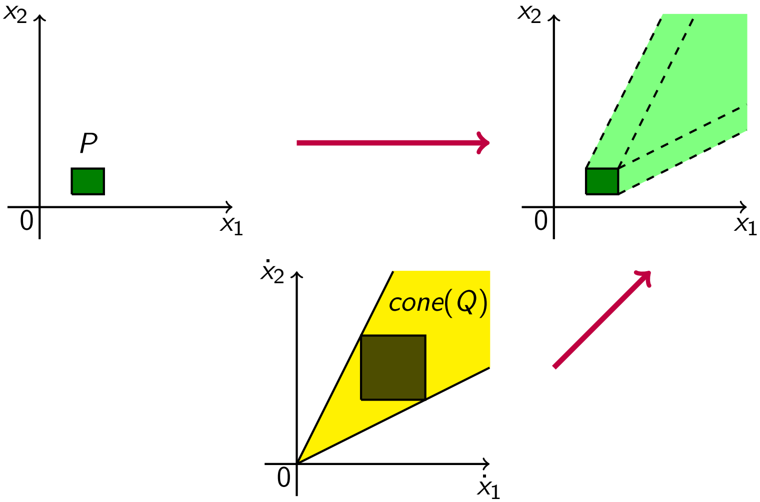
Reachable sets under flow transitions



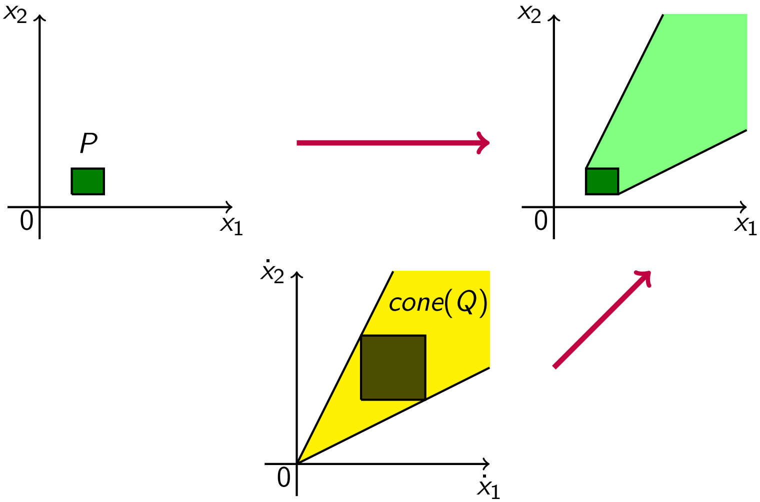
Reachable sets under flow transitions



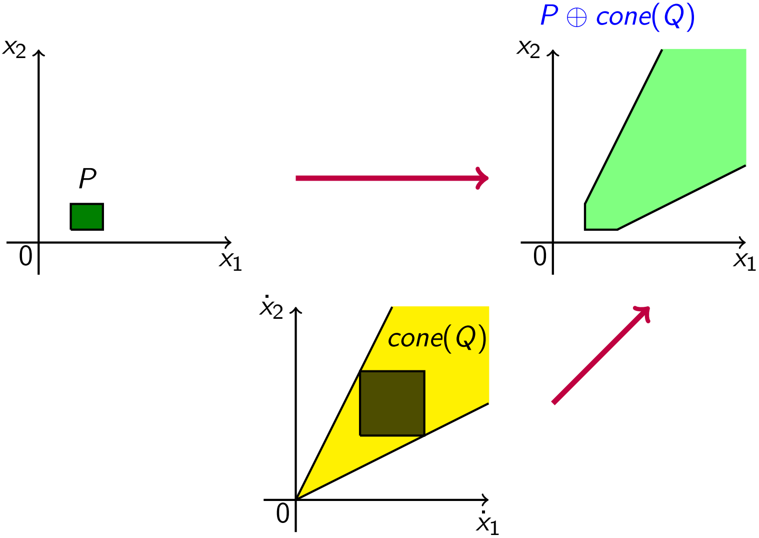
Reachable sets under flow transitions



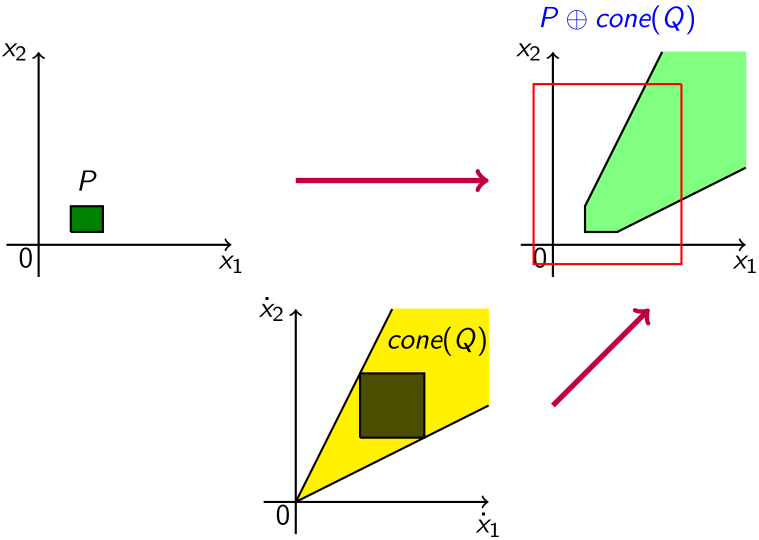
Reachable sets under flow transitions



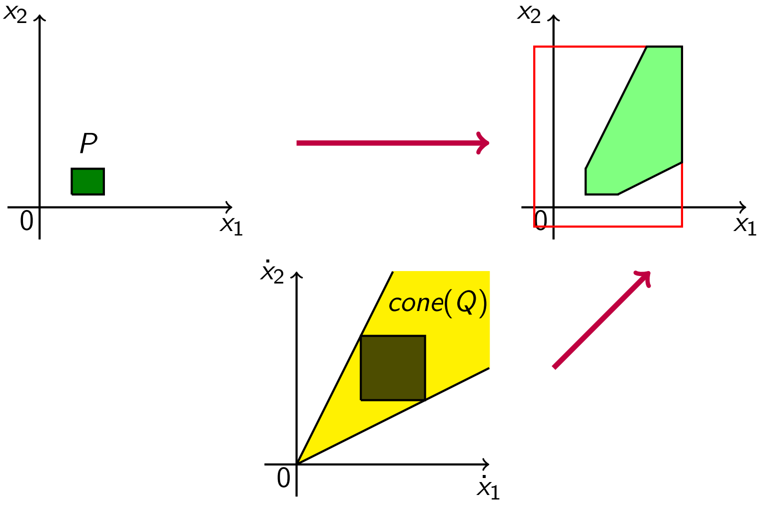
Reachable sets under flow transitions



Reachable sets under flow transitions

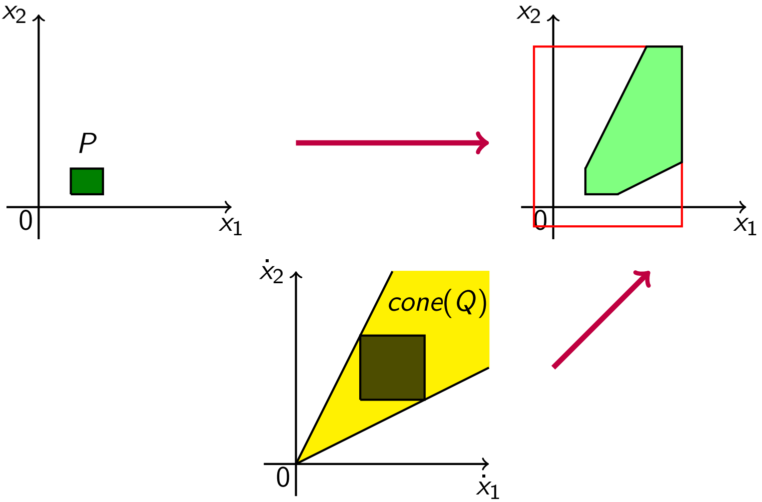


Reachable sets under flow transitions



Reachable sets under flow transitions

$$R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$$



Classical method for computing $R_\ell(P)$

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

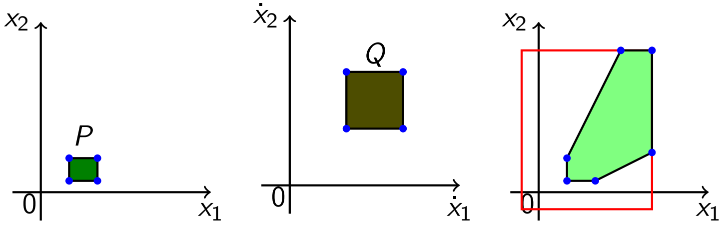
Classical method for computing $R_\ell(P)$

- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Classical method for computing $R_\ell(P)$

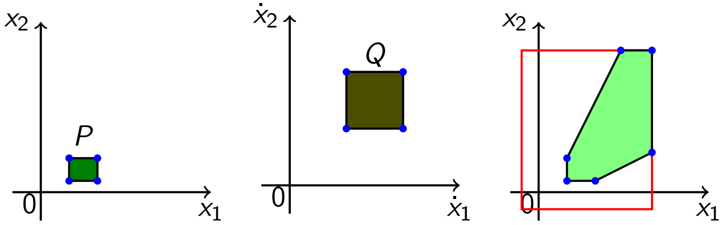
- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.
- Example:



Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Classical method for computing $R_\ell(P)$

- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.
- Example:

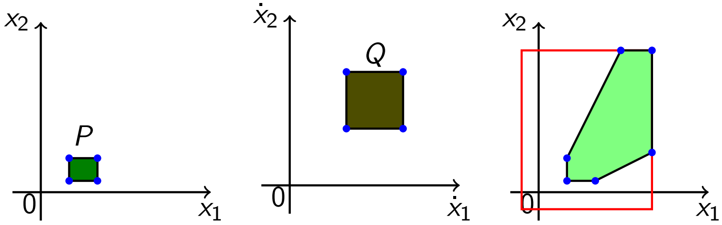


- Used by HyTech and PHAVer.

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Classical method for computing $R_\ell(P)$

- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.
- Example:

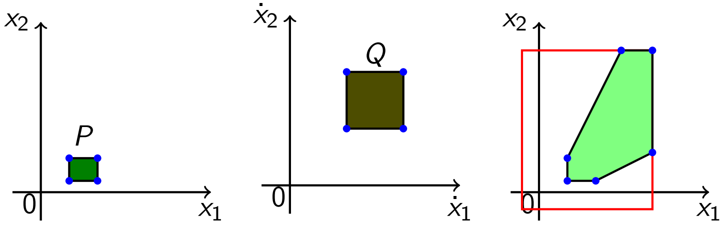


- Used by HyTech and PHAVer.
- Disadvantages:

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Classical method for computing $R_\ell(P)$

- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.
- Example:

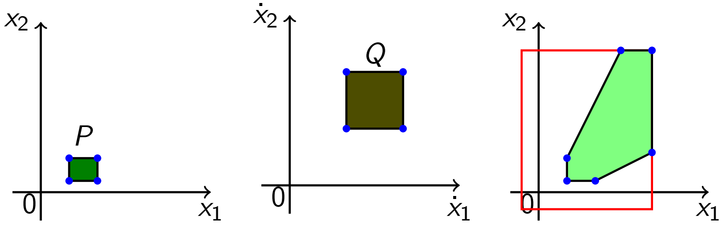


- Used by HyTech and PHAVer.
- Disadvantages:
  1. $O(2^d)$ many vertices for each flow condition;

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Classical method for computing $R_\ell(P)$

- Compute the vertices of $R_\ell(P) = (P \oplus cone(Q)) \cap Inv(\ell)$.
- Example:

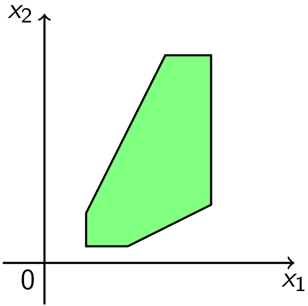


- Used by HyTech and PHAVer.
- Disadvantages:
  1. $O(2^d)$ many vertices for each flow condition;
  2. intersection with an invariant could generate a large number of vertices.

Henzinger et al. HYTECH: A Model Checker for Hybrid Systems. CAV'97

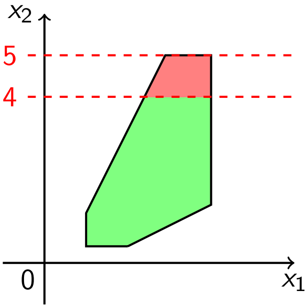
Frehse. PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech. HSCC'05

Reachable sets under jumps



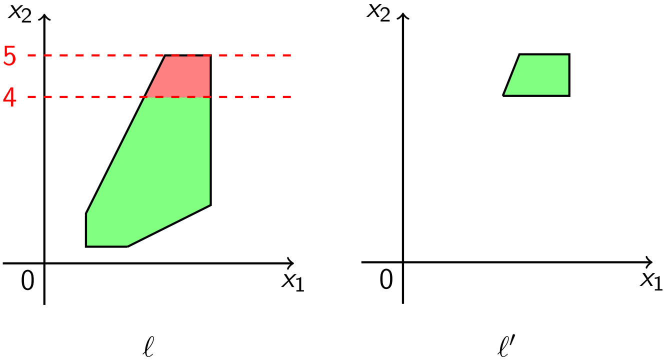
ℓ

Reachable sets under jumps

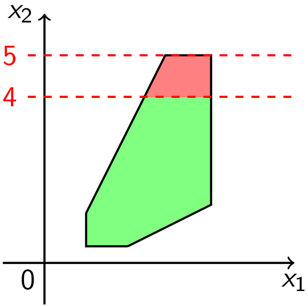


ℓ

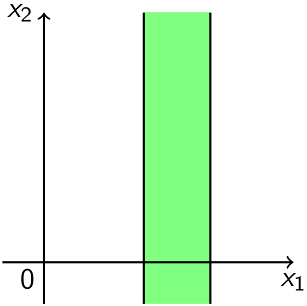
Reachable sets under jumps



Reachable sets under jumps

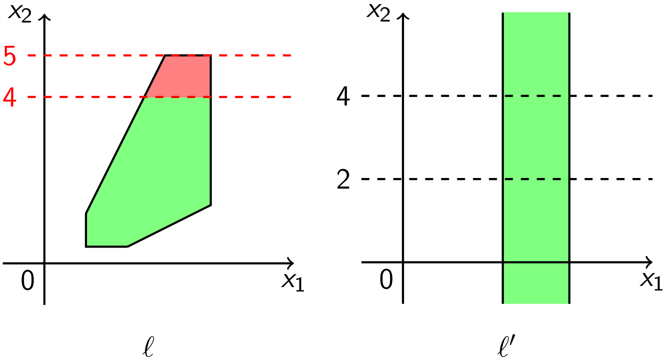


ℓ

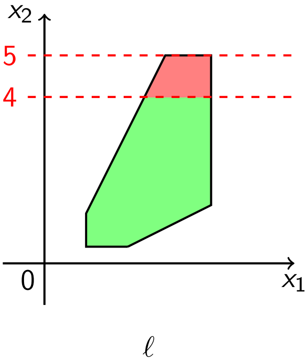


ℓ'

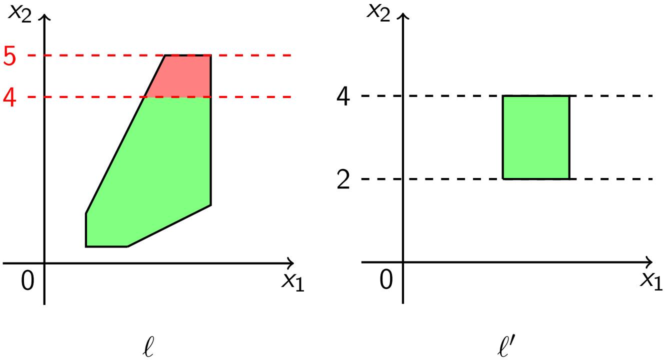
Reachable sets under jumps



Reachable sets under jumps

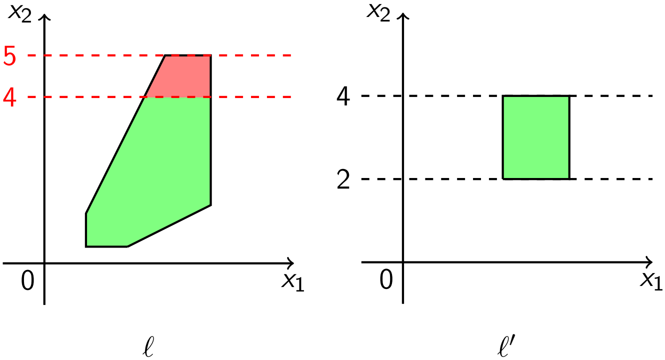


Reachable sets under jumps



- Computed via projection and Minkowski sum.

Reachable sets under jumps



- Computed via projection and Minkowski sum.
- At least $O(2^d)$ many vertices need to be handle.

Our goal

- The reachable set computation under a flow condition is polynomial in d.
- The bounded reachability computation is cheap.

Outline

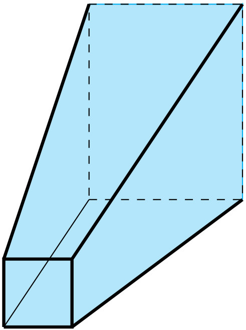
1. Reachability computation for rectangular automata
2. Compute reachable sets efficiently
3. Comparison with PHAVer
4. Future work

Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

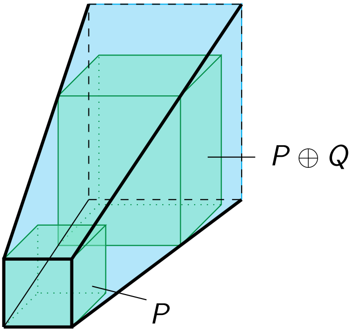
Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:



Properties of the facets of R

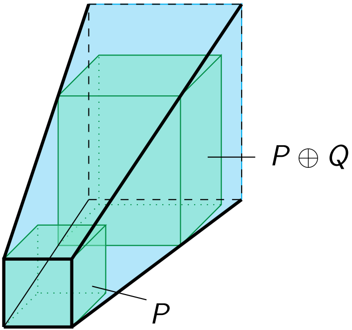
The properties of a facet F_R of $R = P \oplus cone(Q)$:



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

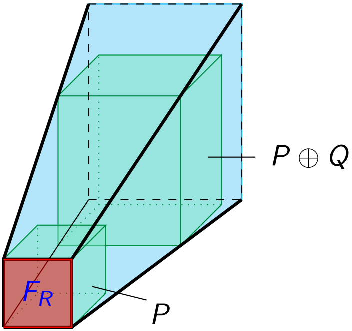
- **Case 1:** F_R is either a facet of P, or



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

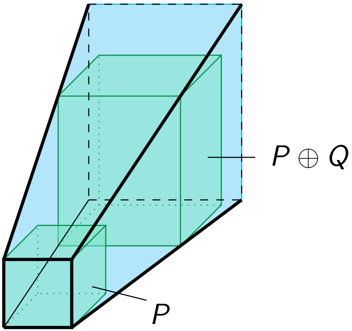
- **Case 1:** F_R is either a facet of P, or



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

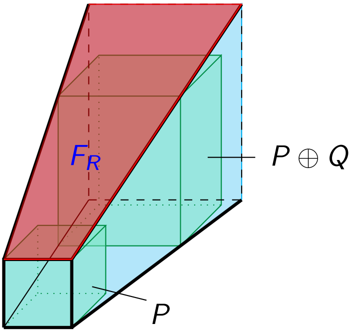
- **Case 1:** F_R is either a facet of P, or
- **Case 2:** $F_R = \bigcup_{\lambda \geq 0}(F_P \oplus \lambda F_Q)$ where F_P, F_Q are nonempty faces of P, Q respectively and $F_P \oplus F_Q$ is a face of $P \oplus Q$.



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

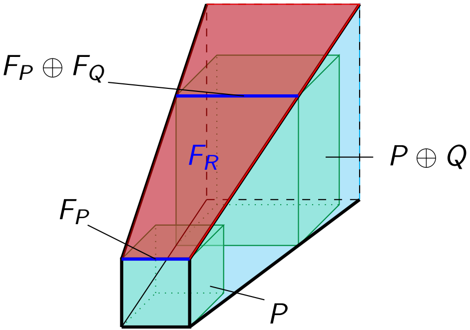
- **Case 1:** F_R is either a facet of P, or
- **Case 2:** $F_R = \bigcup_{\lambda \geq 0}(F_P \oplus \lambda F_Q)$ where F_P, F_Q are nonempty faces of P, Q respectively and $F_P \oplus F_Q$ is a face of $P \oplus Q$.



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

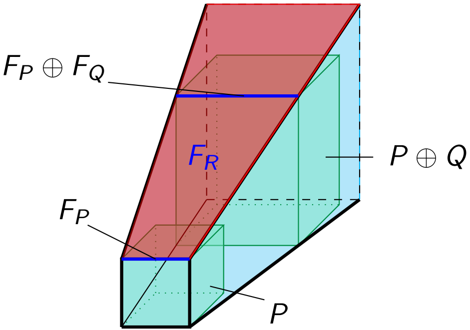
- **Case 1:** F_R is either a facet of P, or
- **Case 2:** $F_R = \bigcup_{\lambda \geq 0}(F_P \oplus \lambda F_Q)$ where F_P, F_Q are nonempty faces of P, Q respectively and $F_P \oplus F_Q$ is a face of $P \oplus Q$.



Properties of the facets of R

The properties of a facet F_R of $R = P \oplus cone(Q)$:

- **Case 1:** F_R is either a facet of P, or
- **Case 2:** $F_R = \bigcup_{\lambda \geq 0}(F_P \oplus \lambda F_Q)$ where F_P, F_Q are nonempty faces of P, Q respectively and $F_P \oplus F_Q$ is a face of $P \oplus Q$.
  $F_P \oplus F_Q$ is at least $(d-2)$-dimensional.



How to compute the constraints for R

Assume $P : \mathcal{L}_P$ and $P \oplus Q : \mathcal{L}_{P \oplus Q}$ are d-dimensional.

How to compute the constraints for R

Assume $P : \mathcal{L}_P$ and $P \oplus Q : \mathcal{L}_{P \oplus Q}$ are d-dimensional.
Main procedure:

1. Collect the valid constraints from $\mathcal{L}_P$ for R.
   The facets of Case 1.

How to compute the constraints for R

Assume $P : \mathcal{L}_P$ and $P \oplus Q : \mathcal{L}_{P \oplus Q}$ are d-dimensional.
Main procedure:

1. Collect the valid constraints from $\mathcal{L}_P$ for R.
   The facets of Case 1.
2. Collect the valid constraints from $\mathcal{L}_{P \oplus Q}$ for R.
   The facets of Case 2 where $F_P \oplus F_Q$ is $(d-1)$-dimensional.

How to compute the constraints for R

Assume $P : \mathcal{L}_P$ and $P \oplus Q : \mathcal{L}_{P \oplus Q}$ are d-dimensional.
Main procedure:

1. Collect the valid constraints from $\mathcal{L}_P$ for R.
   The facets of Case 1.
2. Collect the valid constraints from $\mathcal{L}_{P \oplus Q}$ for R.
   The facets of Case 2 where $F_P \oplus F_Q$ is $(d-1)$-dimensional.
3. For every two constraints $L_i, L_j \in \mathcal{L}_{P \oplus Q}$, compute $L_{i,j}$.
   The facets of Case 2 where $F_P \oplus F_Q$ is $(d-2)$-dimensional.

How to compute the constraints for R

Assume $P : \mathcal{L}_P$ and $P \oplus Q : \mathcal{L}_{P\oplus Q}$ are d-dimensional.
Main procedure:

1. Collect the valid constraints from $\mathcal{L}_P$ for R.
   The facets of Case 1.
2. Collect the valid constraints from $\mathcal{L}_{P\oplus Q}$ for R.
   The facets of Case 2 where $F_P \oplus F_Q$ is $(d-1)$-dimensional.
3. For every two constraints $L_i, L_j \in \mathcal{L}_{P\oplus Q}$, compute $L_{i,j}$.
   The facets of Case 2 where $F_P \oplus F_Q$ is $(d-2)$-dimensional.

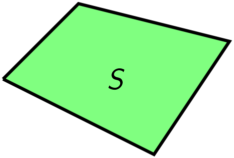
Complexity: $O(|\mathcal{L}_P| + |\mathcal{L}_{P\oplus Q}| + |\mathcal{L}_{P\oplus Q}|^2)$ linear programs need to be solved.

Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.

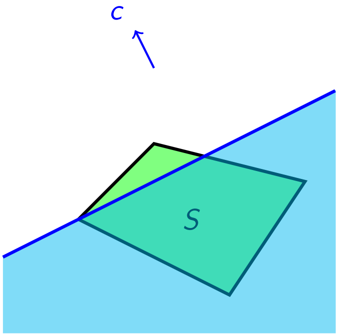
Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.



Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.

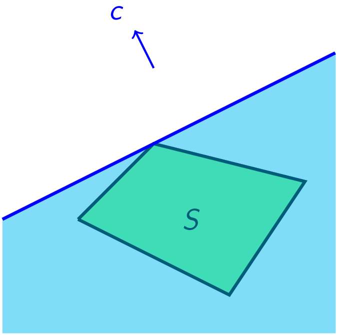


$L : c^T x \leq z'$

$z' < z$

Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.

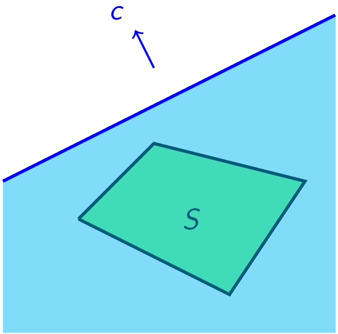


$L : c^T x \leq z'$

$z' = z$

Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.

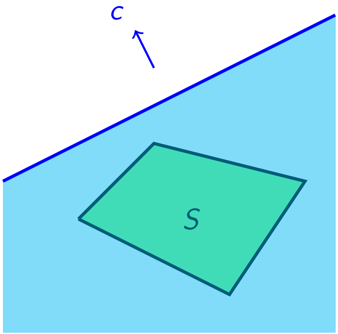


$L : c^T x \leq z'$

$z' > z$

Check the validity of a constraint

Polyhedron $S : \mathcal{L}_S$.



$L : c^T x \leq z'$
$z' > z$

$$L \text{ is valid for } S \text{ iff } \rho_S(c) \leq z',$$

where $\rho_S(c) = \sup c^T x \quad s.t. \quad x$ satisfies all $L_S \in \mathcal{L}_S$.

The validity of a constraint for R

The validity of a constraint for R

For any vector $c \in \mathbb{R}^d$ we have that

$$\rho_R(c) = \rho_{P \oplus cone(Q)}(c) = \sup_{\lambda \geq 0}(\rho_P(c) + \lambda \cdot \rho_Q(c))$$

The validity of a constraint for R

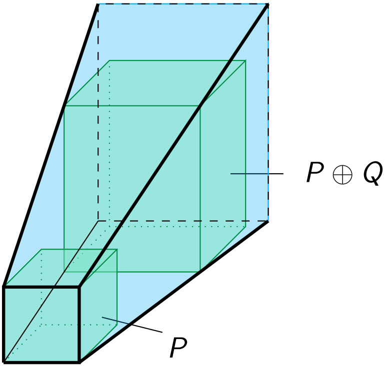
For any vector $c \in \mathbb{R}^d$ we have that

$$\rho_R(c) = \rho_{P \oplus cone(Q)}(c) = \sup_{\lambda \geq 0}(\rho_P(c) + \lambda \cdot \rho_Q(c))$$

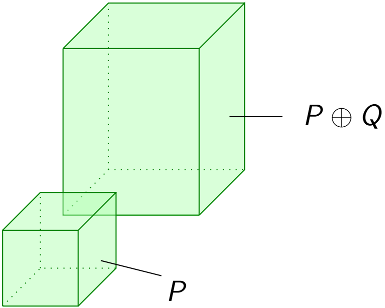
A constraint $L : c^T x \leq z$ is valid for R iff

$$\rho_P(c) \leq z \text{ and } \rho_Q(c) \leq 0.$$

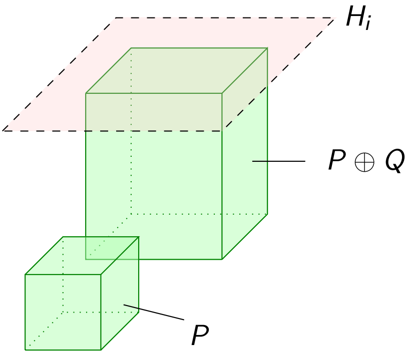
An example of $L_{i,j}$



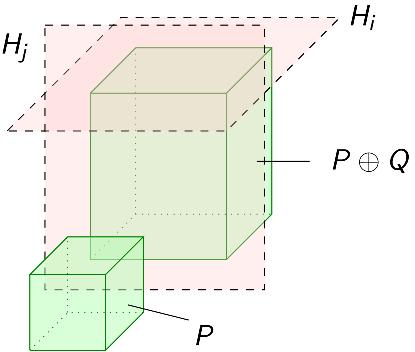
An example of $L_{i,j}$



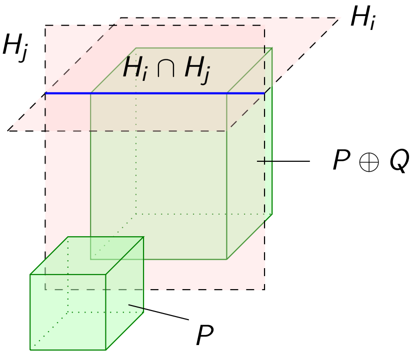
An example of $L_{i,j}$



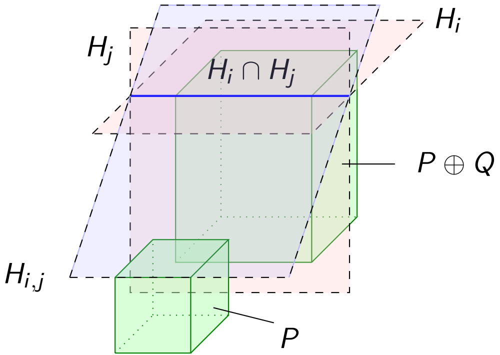
An example of $L_{i,j}$



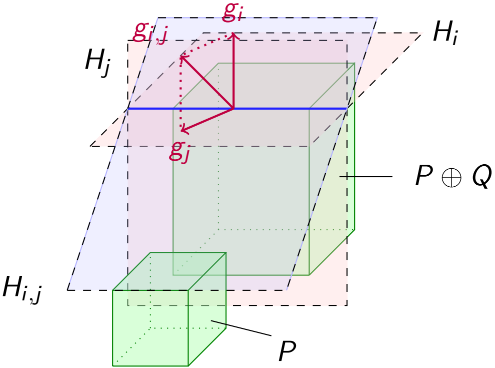
An example of $L_{i,j}$



An example of $L_{i,j}$

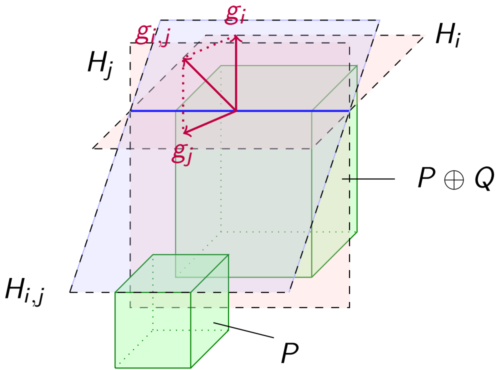


An example of $L_{i,j}$



$g_{i,j} = \alpha g_i + \beta g_j$ where $\alpha, \beta \geq 0$ and $\alpha + \beta > 0$.

An example of $L_{i,j}$



$g_{i,j} = \alpha g_i + \beta g_j$ where $\alpha, \beta \geq 0$ and $\alpha + \beta > 0$.

$H_{i,j} : c^T x = z$ can be found by linear programming and $L_{i,j} : c^T x \leq z$.

Compute the reachable set after a jump

Assume $\mathcal{L}$ defines the reachable set under a flow condition.

1. Eliminate all reset variables from the constraints in $\mathcal{L}$ by Fourier-Motzkin elimination.
2. Add the constraints $x_i \leq b, -x_i \leq -a$ into the new constraint set if there is a reset $x_i := [a, b]$.

Complexity of the computation

- The set of bounded executions along the location sequence:

$$\ell_0 \xrightarrow{e_1} \ell_1 \xrightarrow{e_2} \cdots \xrightarrow{e_k} \ell_k$$

- The corresponding computation sequence:

$$R_{\ell_0}(X_0) \xrightarrow{e_1} R_{\ell_1}(X_1) \xrightarrow{e_2} \cdots \xrightarrow{e_k} R_{\ell_k}(X_k)$$

where $X_j = R_{e_j}(R_{\ell_{j-1}}(X_{j-1}) \cap Guard(e_j) \cap Inv(\ell_{j-1}))$ for $1 \leq j \leq k$.

Complexity of the computation

Complexity of the computation

- The set X_j can be expressed by

$$\bigcup_{a_{j-1} \leq \lambda_{j-1} \leq b_{j-1}} \cdots \bigcup_{a_0 \leq \lambda_0 \leq b_0} R_{e_j}((\cdots R_{e_1}((X_0 \oplus \lambda_0 B_0) \cap G_0) \cdots \oplus \lambda_{j-1} B_{j-1}) \cap G_{j-1})$$

Complexity of the computation

- The set X_j can be expressed by

$$\bigcup_{a_{j-1}\le\lambda_{j-1}\le b_{j-1}} \cdots \bigcup_{a_0\le\lambda_0\le b_0} R_{e_j}((\cdots R_{e_1}((X_0\oplus\lambda_0 B_0)\cap G_0)\cdots\oplus\lambda_{j-1}B_{j-1})\cap G_{j-1})$$

- The number of the facets of X_j is bounded by

$$\mathcal{F}_j = \sum_{\max(d-j-1,0)\le d'\le d-1} \binom{j}{d-d'-1} 2^{d-d'} \binom{d}{d'}$$

Complexity of the computation

- The set X_j can be expressed by

$$\bigcup_{a_{j-1}\le\lambda_{j-1}\le b_{j-1}} \cdots \bigcup_{a_0\le\lambda_0\le b_0} R_{e_j}((\cdots R_{e_1}((X_0\oplus\lambda_0 B_0)\cap G_0)\cdots\oplus\lambda_{j-1}B_{j-1})\cap G_{j-1})$$

- The number of the facets of X_j is bounded by

$$\mathcal{F}_j = \sum_{\max(d-j-1,0)\le d'\le d-1} \binom{j}{d-d'-1} 2^{d-d'} \binom{d}{d'}$$

- $\mathcal{F}_j$ is polynomial in $\mathcal{F}_{j-1}$.

Complexity of the computation

- The set X_j can be expressed by

$$\bigcup_{a_{j-1}\leq\lambda_{j-1}\leq b_{j-1}} \cdots \bigcup_{a_0\leq\lambda_0\leq b_0} R_{e_j}((\cdots R_{e_1}((X_0\oplus\lambda_0 B_0)\cap G_0)\cdots\oplus\lambda_{j-1}B_{j-1})\cap G_{j-1})$$

- The number of the facets of X_j is bounded by

$$\mathcal{F}_j = \sum_{\max(d-j-1,0)\leq d'\leq d-1} \binom{j}{d-d'-1} 2^{d-d'} \binom{d}{d'}$$

- $\mathcal{F}_j$ is polynomial in $\mathcal{F}_{j-1}$.
- If j is fixed, then $\mathcal{F}_j$ is polynomial in d when d is large enough.

Main result

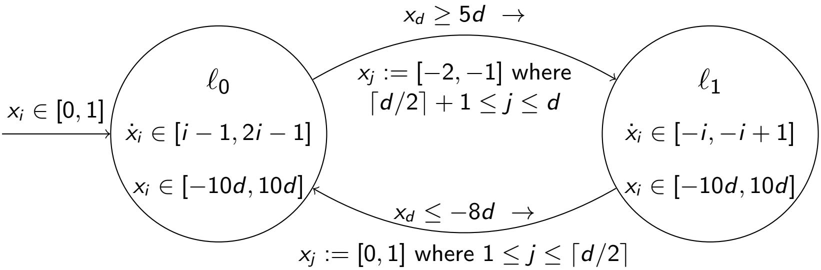
Theorem

The computational complexity of the reachable set with a bounded number of jumps is polynomial in d if the bound is viewed as a constant and d is large enough.

Outline

1. Reachability computation for rectangular automata
2. Compute reachable sets efficiently
3. Comparison with PHAVer
4. Future work

The scalable model



The experimental results

Dim	Jmp	PHAVer		Our method (on MATLAB)				
		Mem	Time	Mem	Time	ToLP	LPs	Cons
5	2	9.9	0.81	< 10	2.36	2.20	1837	81
6	2	48.1	21.69	< 10	4.96	4.68	3127	112
7	2	235.7	529.01	< 10	15.95	15.28	7214	163
8	2	-	-	< 10	27.42	26.48	10517	209
9	2	-	-	< 10	107.99	105.59	23639	287
10	2	-	-	< 10	218.66	215.45	32252	354
5	4	10.2	1.51	< 10	4.82	4.50	3734	167
6	4	51.1	35.52	< 10	11.25	10.64	7307	240
7	4	248.1	1191.64	< 10	32.93	31.60	16101	352
8	4	-	-	< 10	72.04	69.81	27375	466
9	4	-	-	< 10	240.51	235.61	64863	641
10	4	-	-	< 10	543.05	535.77	86633	816

Platform: Intel I7 2.8 GHz CPU, 4GB memory, Linux

Outline

1. Reachability computation for rectangular automata
2. Compute reachable sets efficiently
3. Comparison with PHAVer
4. Future work

Future work

- Bounded reachability computation for linear hybrid automata.
- Synthesis of switching controllers for linear hybrid automata.
- Approximative reachability computation for nonlinear systems.