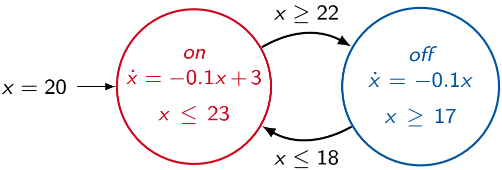
Old-established methods in a new look: How HyPro speeds up reachability computations for hybrid systems

Stefan Schupp   Erika Ábrahám

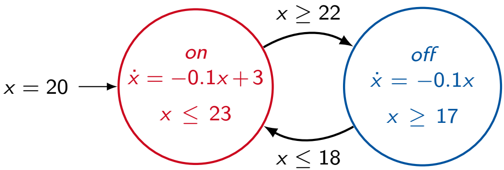
RWTH Aachen University, Germany

FLoC/ADHS 2018, Oxford, UK

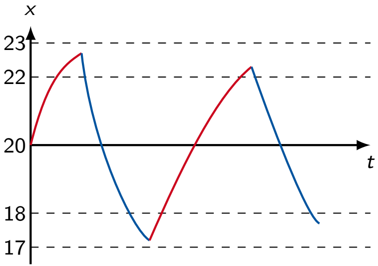
Hybrid systems – Example: Thermostat



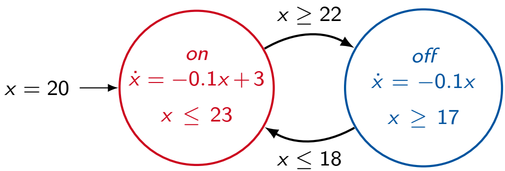
Hybrid systems – Example: Thermostat



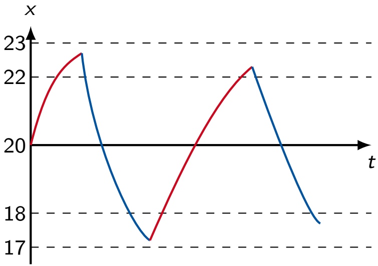
Example trajectory:



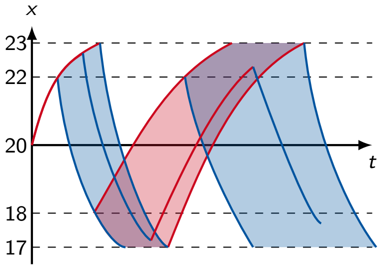
Hybrid systems – Example: Thermostat



Example trajectory:



Set of reachable states:



Reachability problem on hybrid automata

subclasses	derivatives	conditions	bounded reachability	unbounded reachability
timed automata	$\dot{x} = 1$	$x \sim c$	decidable	decidable
initialized rectangular automata	$\dot{x} \in [c_1, c_2]$	$x \in [c_1, c_2]$ necessary when $\dot{x}$ changes	decidable	decidable
rectangular automata	$\dot{x} \in [c_1, c_2]$	$x \in [c_1, c_2]$	decidable	undecidable
linear hybrid automata I	$\dot{x} = c$	$x \sim g_{linear}$	decidable	undecidable
linear hybrid automata II	$\dot{x} = f_{linear}$	$x \sim g_{linear}$	undecidable	undecidable
hybrid automata	$\dot{x} = f$	$x \sim g$	undecidable	undecidable

Reachability problem on hybrid automata

subclasses	derivatives	conditions	bounded reachability	unbounded reachability
timed automata	$\dot{x} = 1$	$x \sim c$	decidable	decidable
initialized rectangular automata	$\dot{x} \in [c_1, c_2]$	$x \in [c_1, c_2]$ necessary when $\dot{x}$ changes	decidable	decidable
rectangular automata	$\dot{x} \in [c_1, c_2]$	$x \in [c_1, c_2]$	decidable	undecidable
linear hybrid automata I	$\dot{x} = c$	$x \sim g_{linear}$	decidable	undecidable
linear hybrid automata II	$\dot{x} = f_{linear}$	$x \sim g_{linear}$	undecidable	undecidable
hybrid automata	$\dot{x} = f$	$x \sim g$	undecidable	undecidable

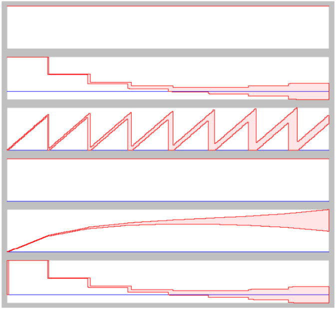
Impressive tool development for hybrid systems reachability analysis

(incomplete list)

- HSolver [Ratschan et al., HSCC 2005]
- iSAT-ODE [Eggers et al., ATVA 2008]
- KeYmaera (X) [Platzer et al., IJCAR 2008]
- PowerDEVS [Bergero et al., Simulation 2011]
- SpaceEx [Frehse et al., CAV 2011]
- S-TaLiRo [Annapureddy et al., TACAS 2011]
- Ariadne [Collins et al., ADHS 2012]
- HySon [Bouissou et al., RSP 2012]
- Flow* [Chen et al., CAV 2013]
- HyCreate [Bak et al., HSCC 2013]
- HyEQ [Sanfelice et al., HSCC 2013]
- NLTOOLBOX [Testylier et al., ATVA 2013]
- SoapBox [Hagemann et al., ARCH 2014]
- Acumen [Taha et al., IoT 2015]
- C2E2 [Duggirala et al., TACAS 2015]
- Cora [Althoff et al., ARCH 2015]
- dReach [Kong et al, TACAS 2015]
- Isabelle/HOL [Immler, TACAS 2015]
- HyLAA [Bak et al., HSCC 2017]
- HyPro/HyDRA [Schupp et al., NFM 2017]

Verification techniques/tools for hybrid systems

(Rigorous/verified) simulation: Besides simulation for testing, rigorous/verified simulation can be used for (bounded) reachability analysis.
Some tools: Acumen, C2E2, HyEQ, HyLAA, HySon, S-TaLiRo, PowerDEVS

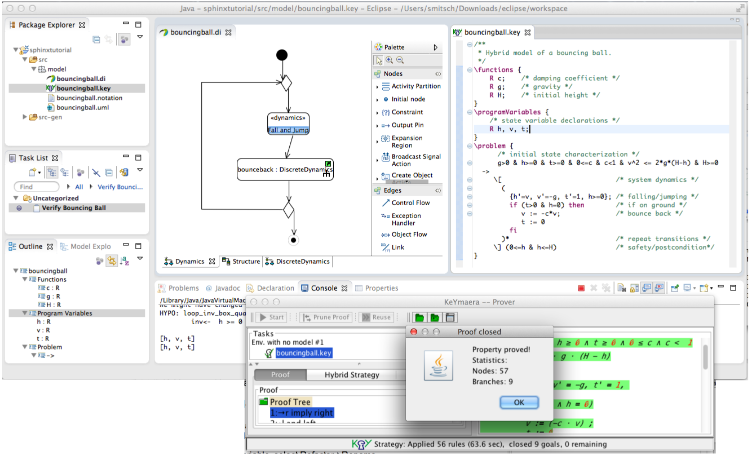


Source: http://www.acumen-language.org/

Verification techniques/tools for hybrid systems

Deduction: Finding and showing invariants using theorem proving.

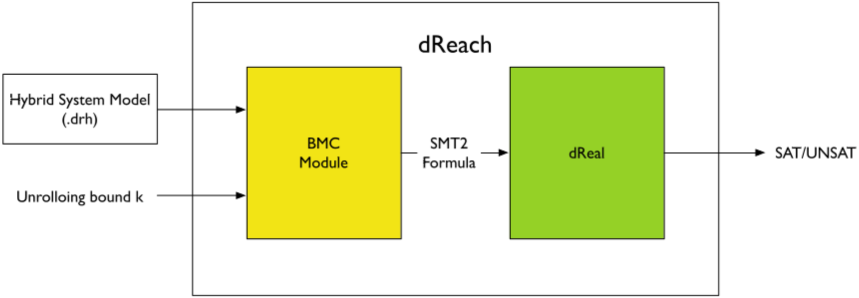
Some tools: Ariadne, Isabelle/HOL, KeYmaera



Source: `http://symbolaris.com/info/keymaera.html`

Verification techniques/tools for hybrid systems

Bounded model checking / interval arithmetic: System executions and requirements are encoded by logical formulas; satisfiability checking tools (SMT solvers) are used for (bounded) reachability analysis.
Some tools: dReach, HSolver, iSAT-ODE

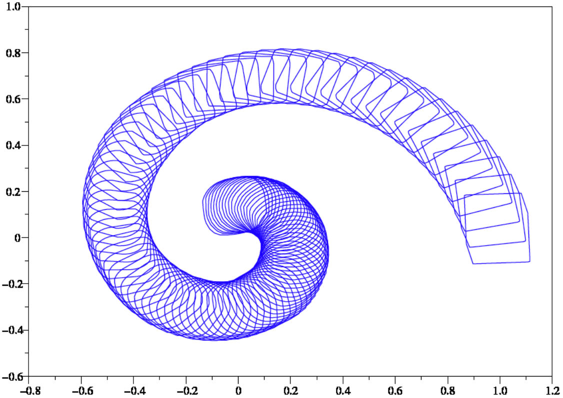


Source: `http://dreal.github.io/dReach/`

Verification techniques/tools for hybrid systems

Over-approximating flowpipe construction: Iterative (bounded) forward reachability analysis based on some over-approximative symbolic state set representations.

Some tools: Cora, Flow*, HyCreate, HyPro/HyDRA, NLTOOLBOX, SoapBox, SpaceEx



State set representations

Most well-known state set representations

boxes (hyper-rectangles) [Moore et al., 2009]
oriented rectangular hulls [Stursberg et al., 2003]
convex polyhedra [Ziegler, 1995] [Chen at el, 2011]
template polyhedra [Sankaranarayanan et al., 2008]
orthogonal polyhedra [Bournez et al., 1999]
zonotopes [Girard, 2005]
ellipsoids [Kurzhanski et al., 2000]
support functions [Le Guernic et al., 2009]
Taylor models [Berz and Makino, 1998, 2009] [Chen et al., 2012]

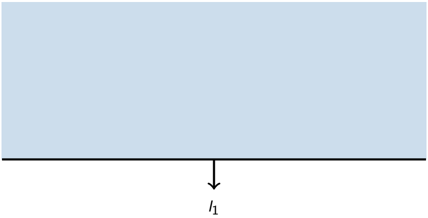
Some needed set operations:

intersection	union	projection
linear transformation	Minkowski sum	
test for membership	test for emptiness	

Example state set representation: Polytopes

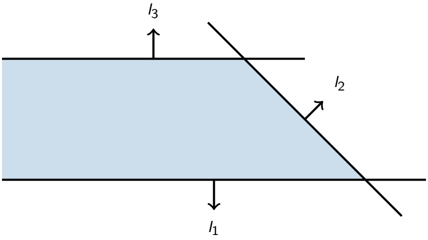
Example state set representation: Polytopes

- Halfspace: set of points x satisfying $l \cdot x \leq z$



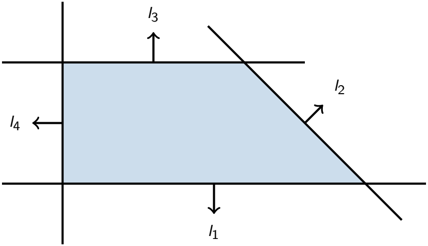
Example state set representation: Polytopes

- Halfspace: set of points x satisfying $l \cdot x \leq z$
- Polyhedron: an intersection of finitely many halfspaces



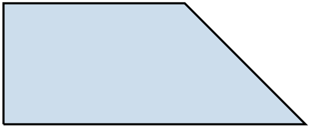
Example state set representation: Polytopes

- Halfspace: set of points x satisfying $l \cdot x \leq z$
- Polyhedron: an intersection of finitely many halfspaces
- Polytope: a bounded polyhedron



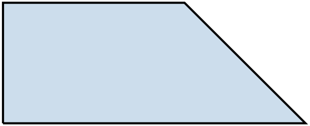
Example state set representation: Polytopes

- Halfspace: set of points x satisfying $l \cdot x \leq z$
- Polyhedron: an intersection of finitely many halfspaces
- Polytope: a bounded polyhedron



Example state set representation: Polytopes

- Halfspace: set of points x satisfying $l \cdot x \leq z$
- Polyhedron: an intersection of finitely many halfspaces
- Polytope: a bounded polyhedron

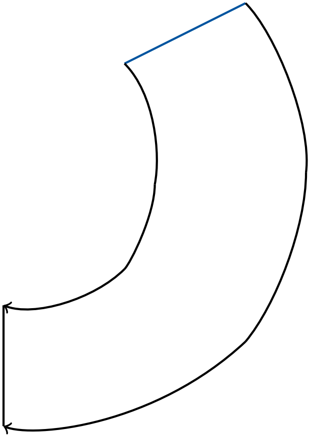


representation	union	intersection	Minkowski sum
$\mathcal{V}$-representation by vertices	easy	hard	easy
$\mathcal{H}$-representation by facets	hard	easy	hard

Reachability computation for LHA: Time evolution

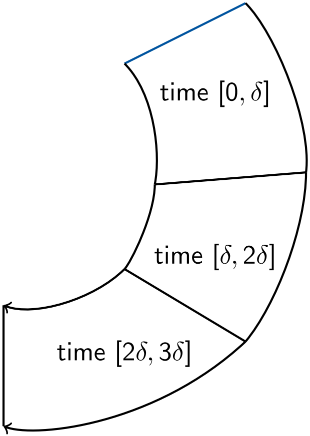
Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$



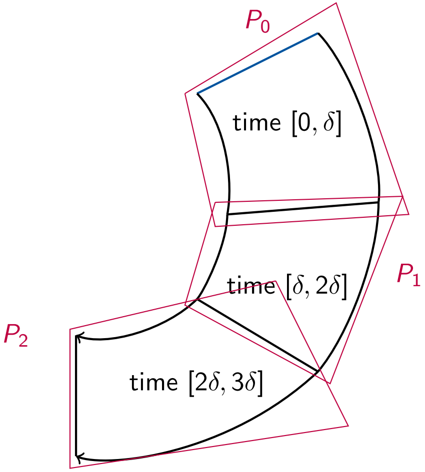
Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$



Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i

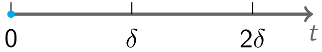


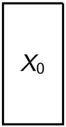
Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:

Reachability computation for LHA: Time evolution

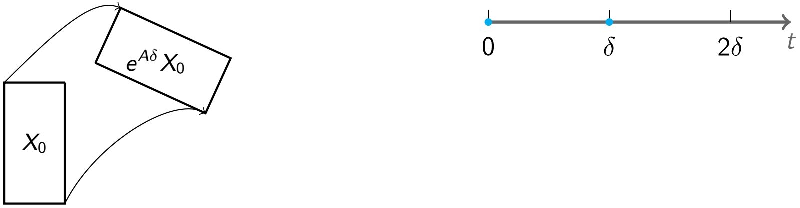
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:





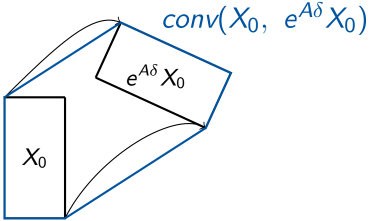
Reachability computation for LHA: Time evolution

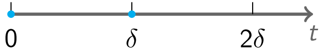
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$



Reachability computation for LHA: Time evolution

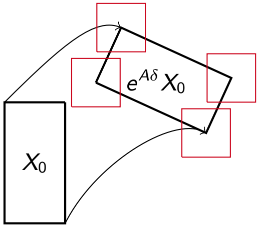
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

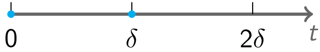




Reachability computation for LHA: Time evolution

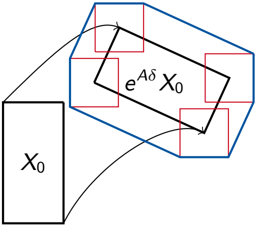
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

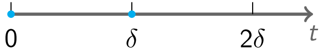




Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

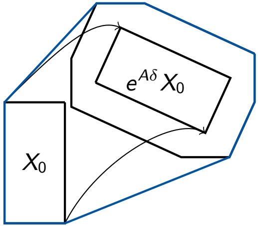




Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

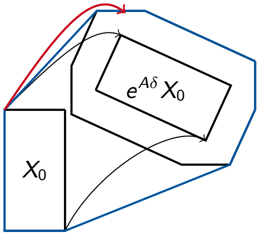
$conv(X_0,\ e^{A\delta} X_0 \oplus B_1)$

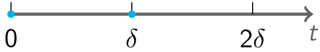


over-approximates flowpipe
for time $[0, \delta]$
under dynamics $\dot{x} = Ax$

Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

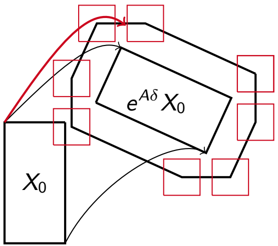


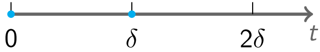


disturbance!

Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

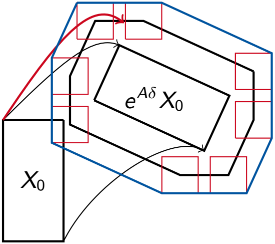




disturbance!

Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

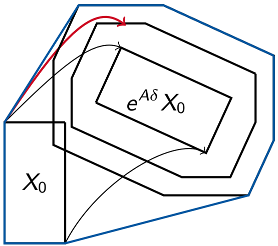


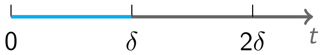
disturbance!

Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax+Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

$P_0 = conv(X_0,\ e^{A\delta}X_0 \oplus B_1 \oplus B_2)$

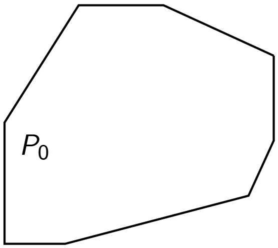


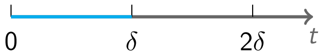


over-approximates flowpipe
for time $[0, \delta]$
under dynamics $\dot{x} = Ax+Bu$

Reachability computation for LHA: Time evolution

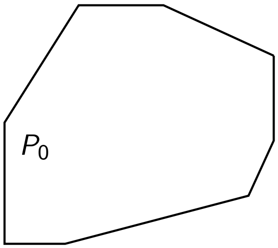
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$

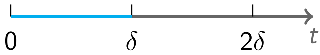




Reachability computation for LHA: Time evolution

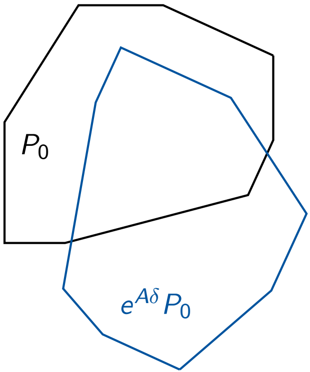
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:

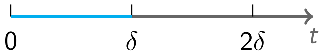




Reachability computation for LHA: Time evolution

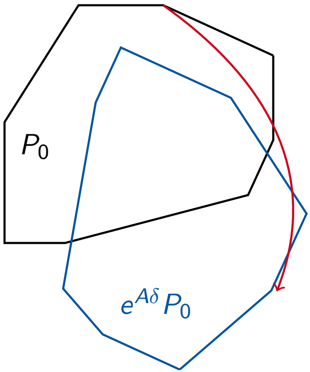
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:

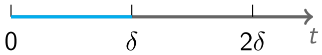




over-approximates flowpipe
for time $[\delta, 2\delta]$
under dynamics $\dot{x} = Ax$

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:

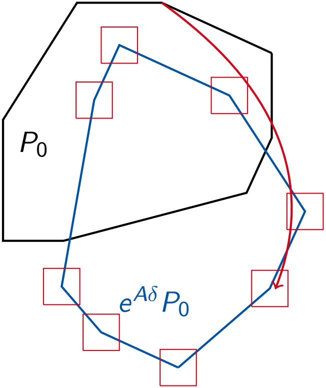




disturbance!

Reachability computation for LHA: Time evolution

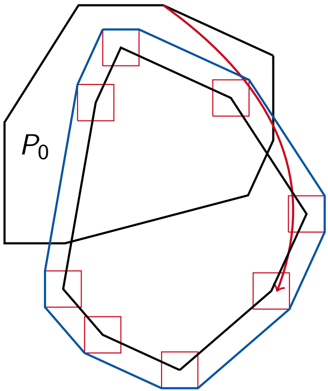
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:



disturbance!

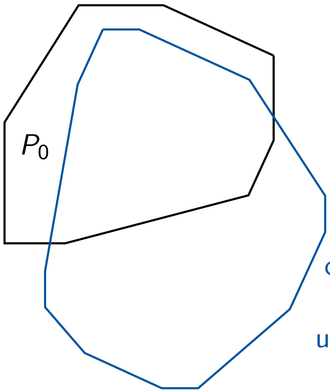
Reachability computation for LHA: Time evolution

- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:



Reachability computation for LHA: Time evolution

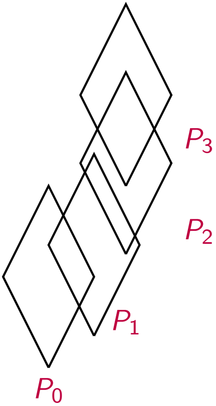
- Assume: initial set X_0, flow $\dot{x} = Ax + Bu$
- Over-approximate flowpipe segment for time $[i\delta, (i+1)\delta]$ by P_i
- The first flowpipe segment:
- Reminder matrix exponential: $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$
- The remaining ones:



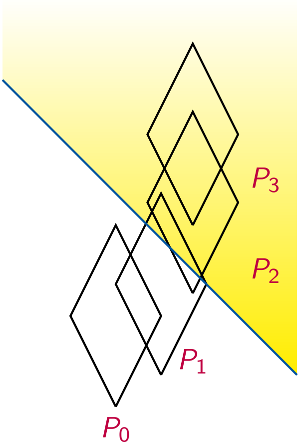
$P_1 = e^{A\delta} P_0 \oplus B_2$

over-approximates flowpipe
for time $[\delta, 2\delta]$
under dynamics $\dot{x} = Ax + Bu$

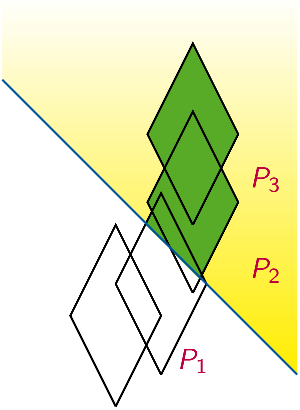
Reachability computation for LHA: Jump successors



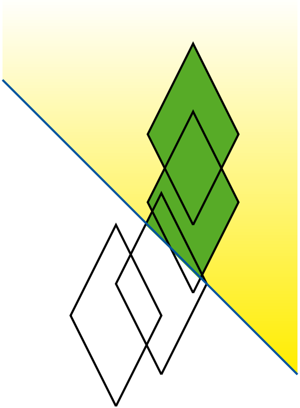
Reachability computation for LHA: Jump successors



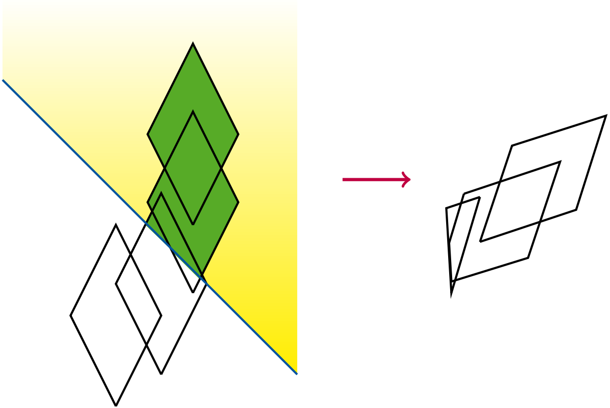
Reachability computation for LHA: Jump successors



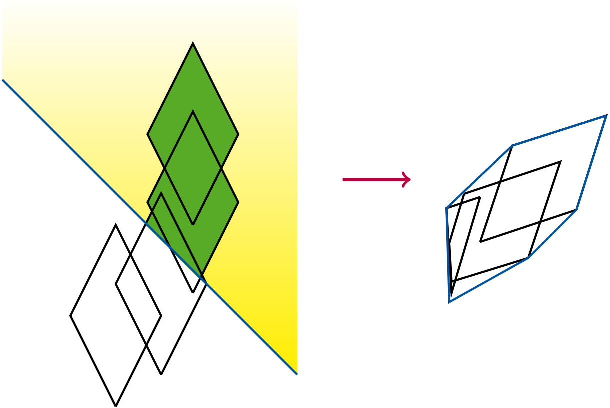
Reachability computation for LHA: Jump successors



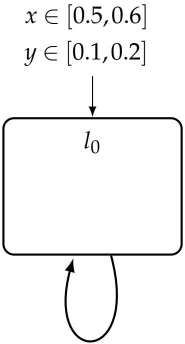
Reachability computation for LHA: Jump successors

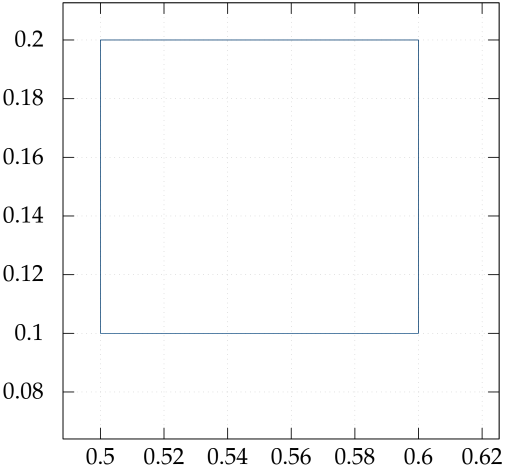


Reachability computation for LHA: Jump successors

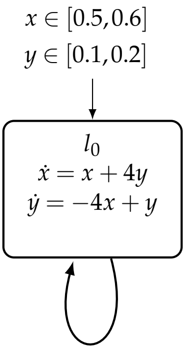


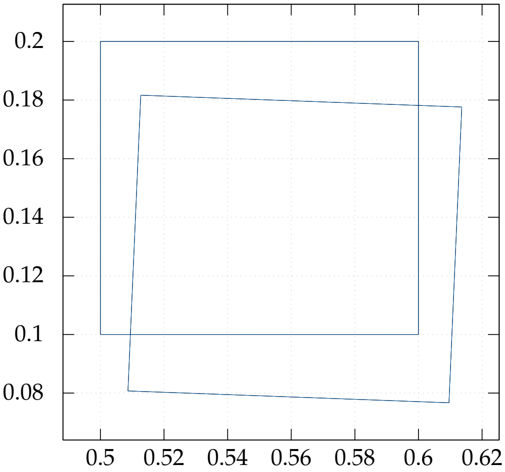
Reachability computation for LHA: Example





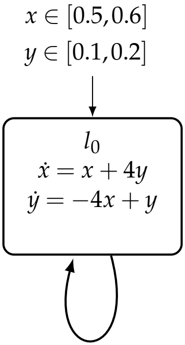
Reachability computation for LHA: Example

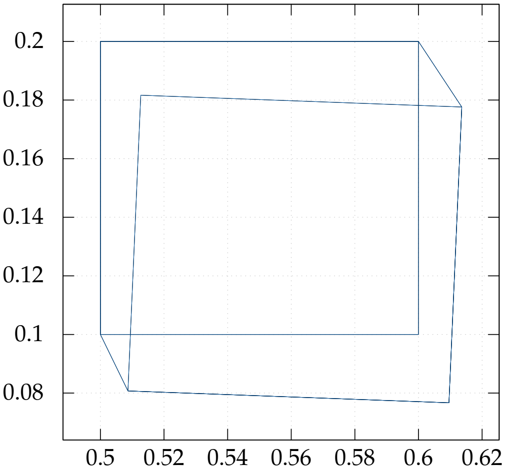




`linear transformation`

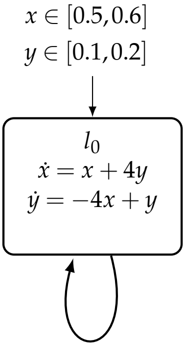
Reachability computation for LHA: Example

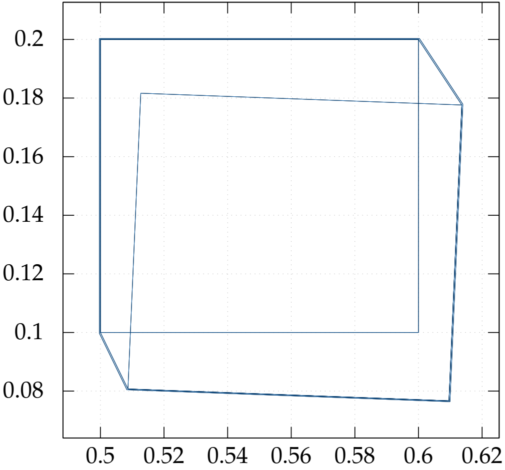




`union`

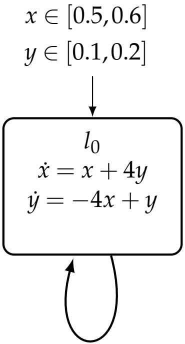
Reachability computation for LHA: Example

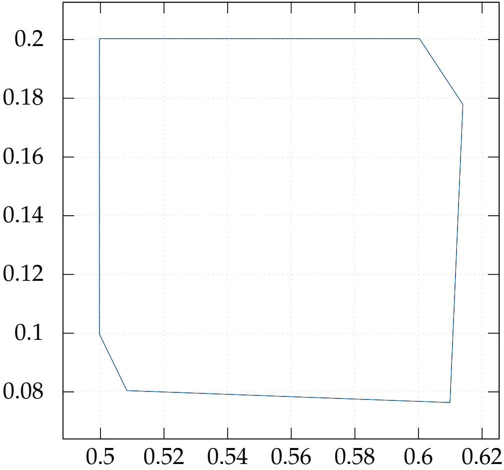




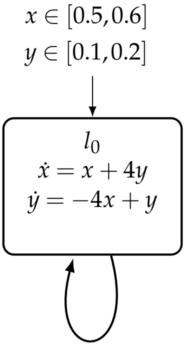
Minkowski sum

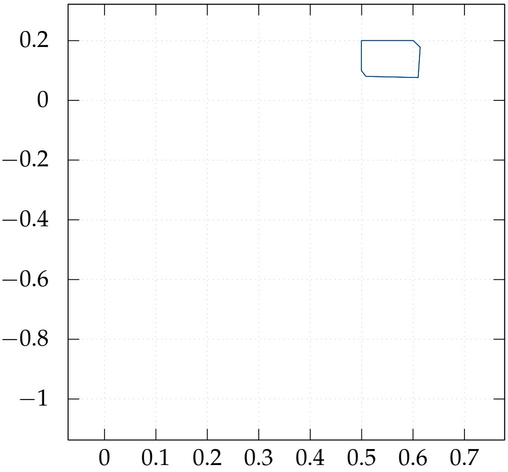
Reachability computation for LHA: Example



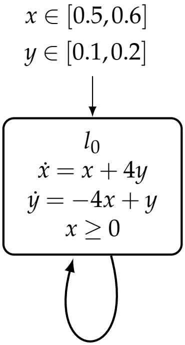


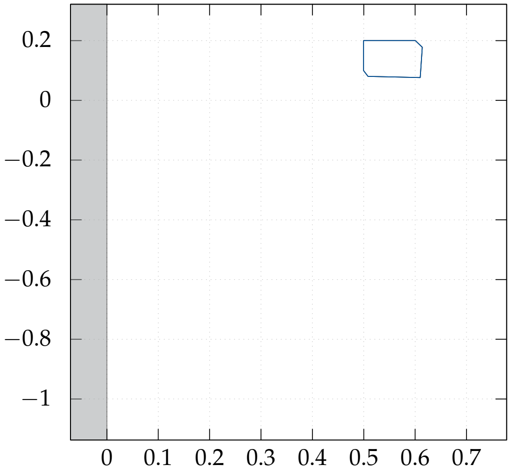
Reachability computation for LHA: Example





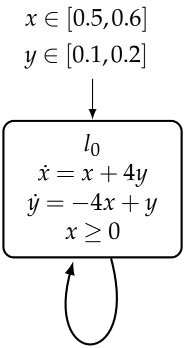
Reachability computation for LHA: Example

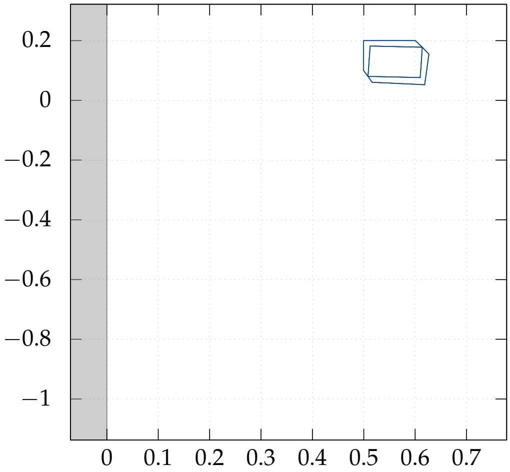




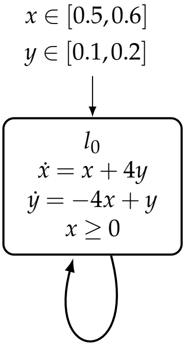
`intersection`

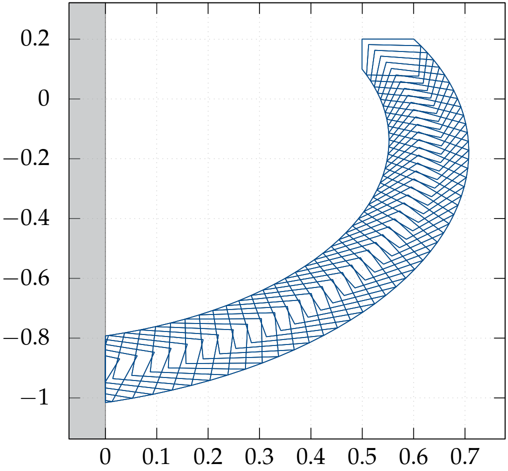
Reachability computation for LHA: Example





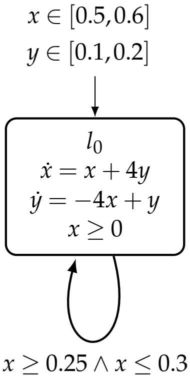
`linear transformation`

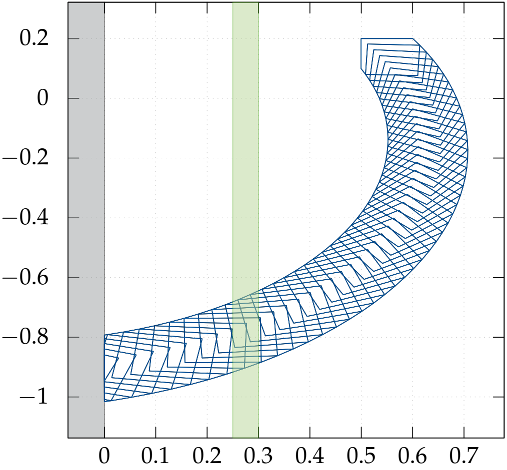




`linear transformation`

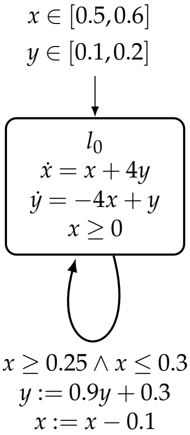
Reachability computation for LHA: Example

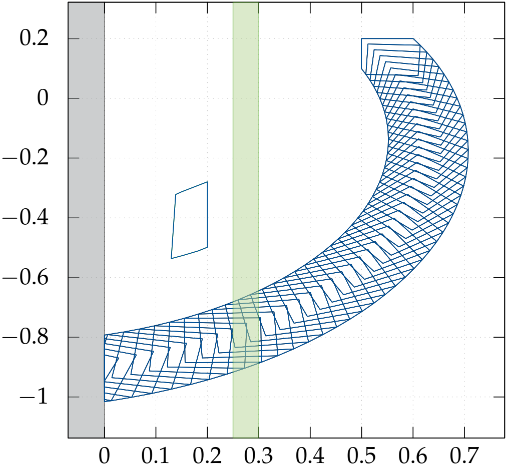




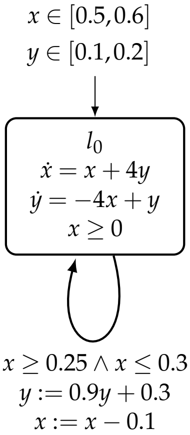
`intersection`

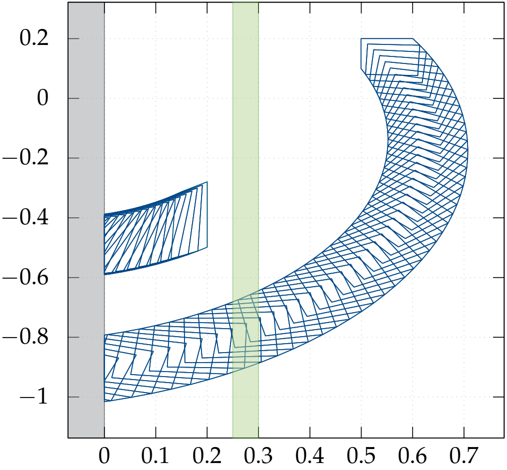
Reachability computation for LHA: Example





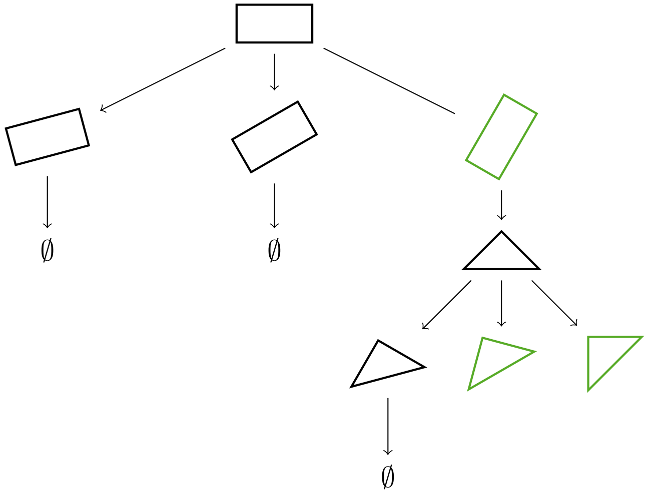
`linear transformation`



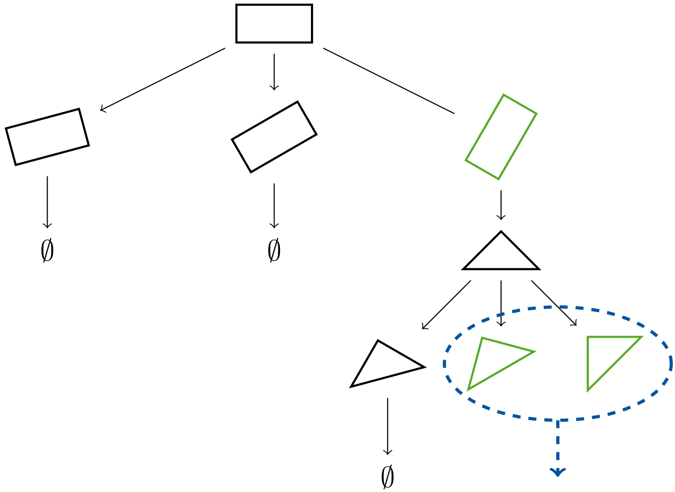


`linear transformation`

Reachability analysis search tree



Reachability analysis search tree



Flow* [Chen et al., CAV 2013]

- Taylor model-based approach
- non-linear dynamic
- adaptive refinement methods

Available at `https://flowstar.org/`

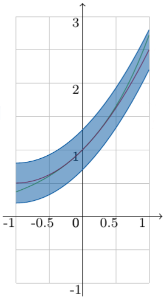


Image: Xin Chen

Has been used in a variety of verification tasks, e.g.

- biological/medical systems (glucose control, spiking neurons, Lotka Volterra equations),
- circuits (oscillators, van der Pol circuit),
- mechanical systems (jet engine model)

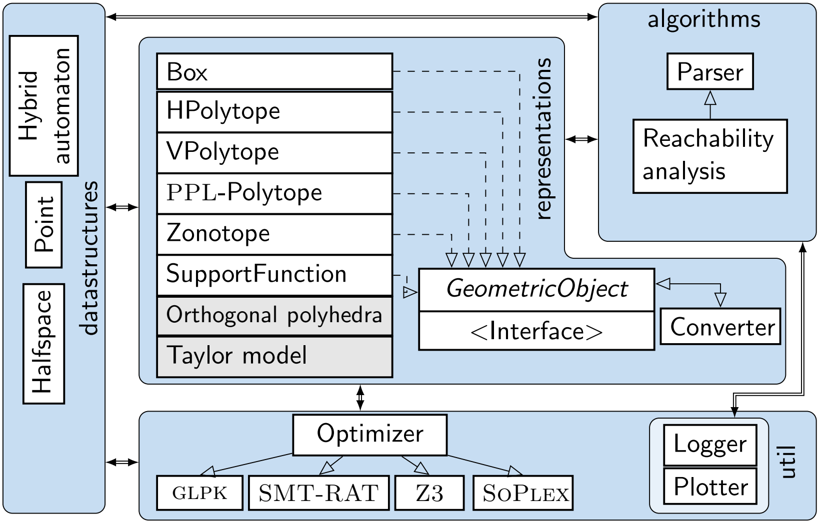
HyPro [Schupp et al., NFM 2017]

A free and open-source `C++` library for state set representations for the reachability analysis of hybrid systems.

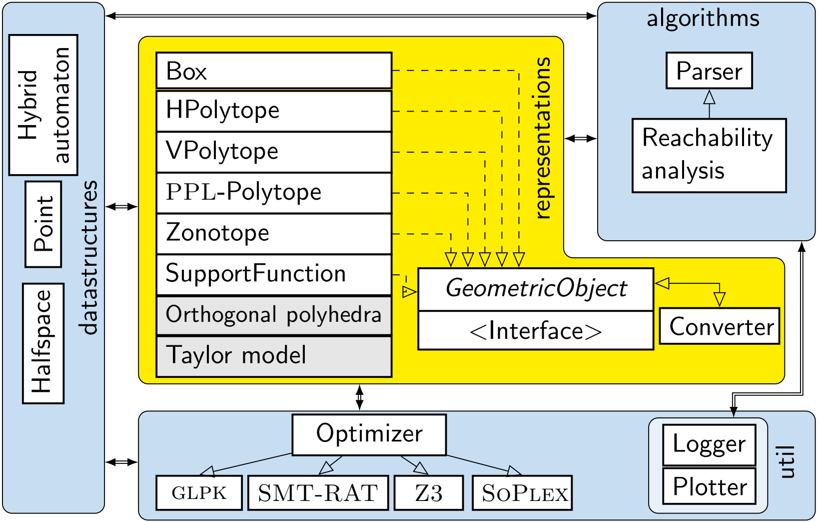
Available at `https://github.com/hypro/hypro`.

Allows the fast implementation of specialized reachability analysis methods.

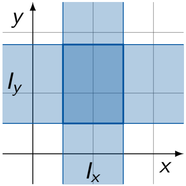
HyPro/HyDRA: Structure



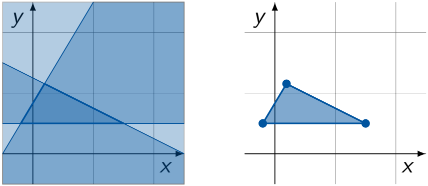
HyPro/HyDRA: Structure



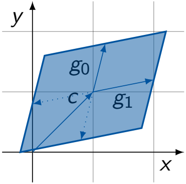
HyPro: State set representations



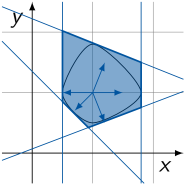
Boxes



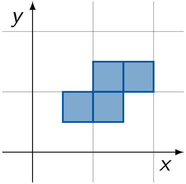
Convex polyhedra ($\mathcal{H}$, $\mathcal{V}$, PPL)



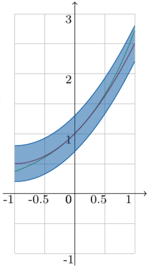
Zonotopes



Support functions



Orthogonal polyhedra



Taylor models

Source: Xin Chen

Main functionalities of `GeometricObject`

Set operations:

`X.affineTransformation(matrix A, vector b)`	$AX + b$
`X.minkowskiSum(geometricObject Y)`	$X \oplus Y$
`X.intersectHalfspaces(matrix A, vector b)`	$X \cap \{y \mid Ay \leq b\}$
`X.satisfiesHalfspaces(matrix A, vector b)`	$X \cap \{y \mid Ay \leq b\} \neq \emptyset$
`X.unite(geometricObject Y)`	$cl(X \cup Y)$

Main functionalities of `GeometricObject`

Set operations:

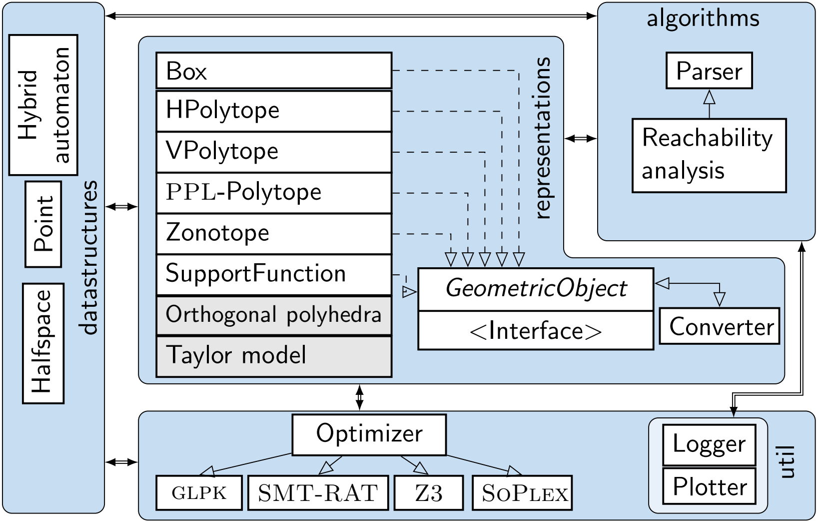
`X.affineTransformation(matrix A, vector b)`	$AX + b$
`X.minkowskiSum(geometricObject Y)`	$X \oplus Y$
`X.intersectHalfspaces(matrix A, vector b)`	$X \cap \{y \mid Ay \leq b\}$
`X.satisfiesHalfspaces(matrix A, vector b)`	$X \cap \{y \mid Ay \leq b\} \neq \emptyset$
`X.unite(geometricObject Y)`	$cl(X \cup Y)$

Set utility functions:

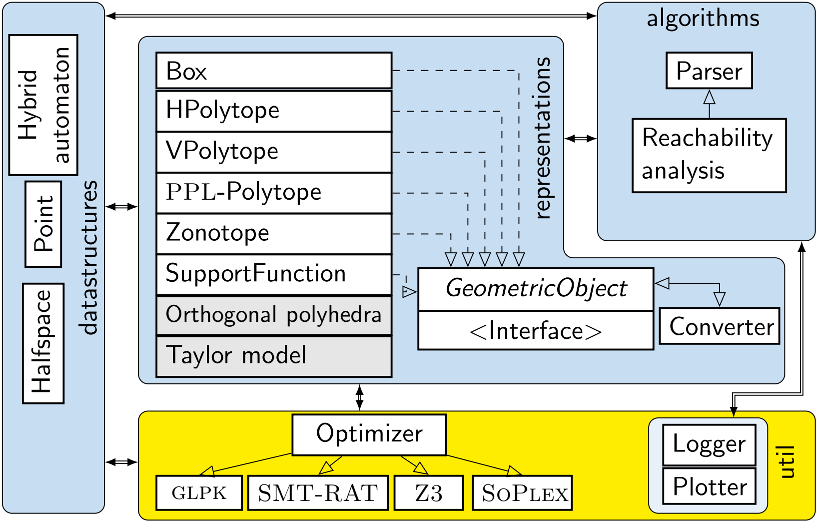
```
dimension()
empty()
vertices()
project(vector<dimensions> d)
contains(point p)
```

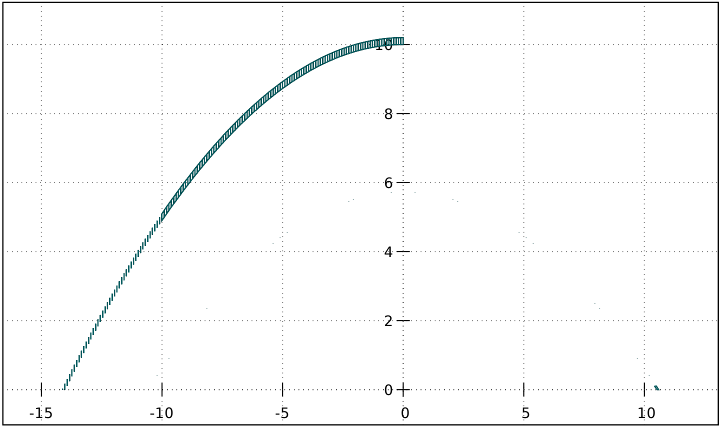
conversion operations
reduction functions

HyPro/HyDRA: Structure

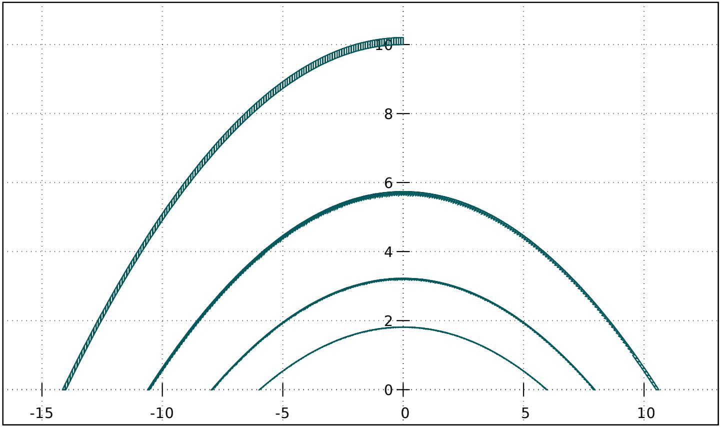


HyPro/HyDRA: Structure





10
8
6
4
2
0
-15
-10
-5
0
5
10



10
8
6
4
2
0
-15
-10
-5
0
5
10

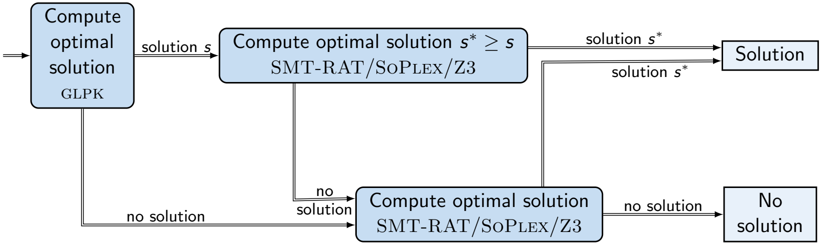
HyPro: Linear optimization

HyPro offers different number representations:
`cln::cl_RA`, `mpq_class`, `double`

Obstacles:

- inexact linear optimization not suitable
- exact linear optimization expensive

⇝ combined application

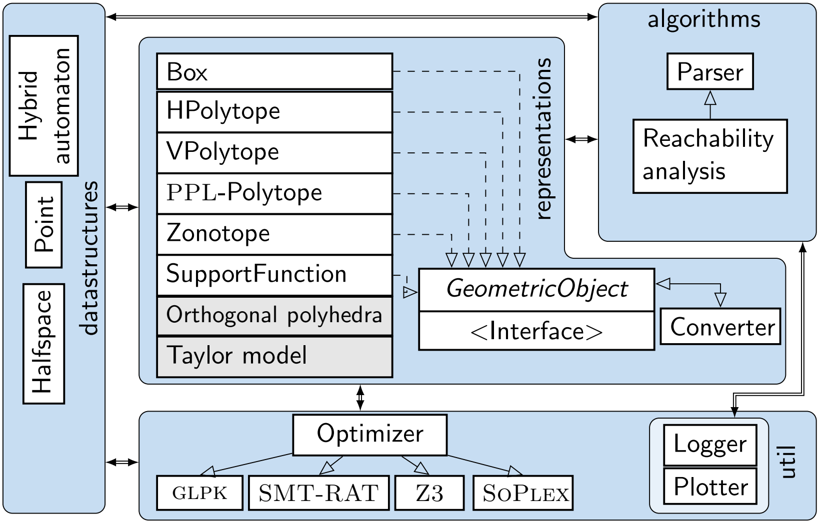


Utility

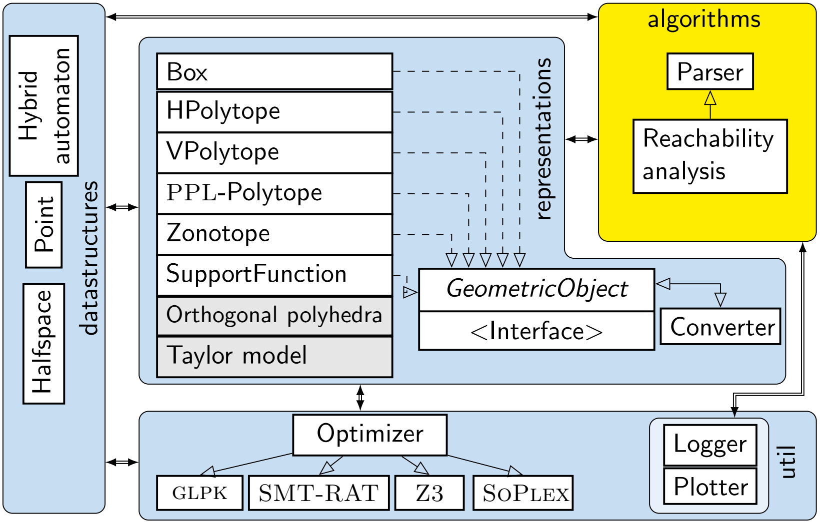
Further utility functions:

- datastructures for e.g. hybrid automata, point, halfspace
- parser for FLOW*-based syntax
- GNUPLOT plotting interface (pdf, eps and tex)
- logging

HyPro/HyDRA: Structure



HyPro/HyDRA: Structure



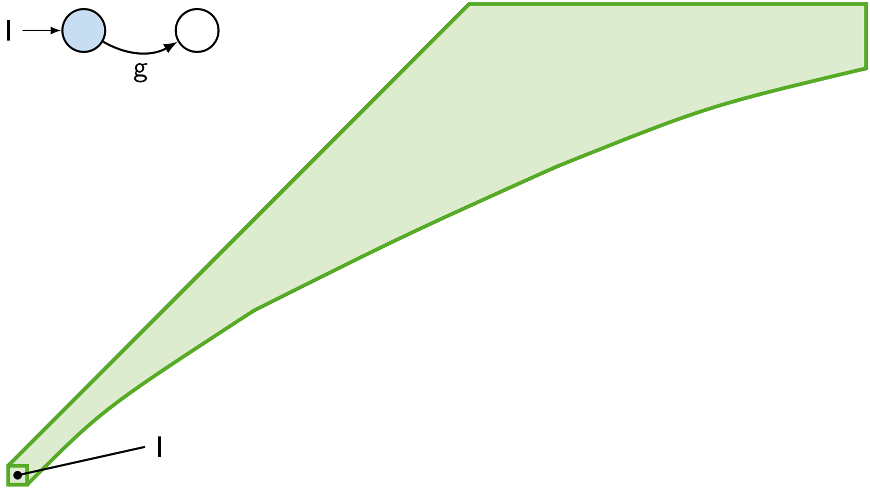
HyDRA techniques

1. Counterexample-guided abstraction refinement
2. Parallelization
3. Sub-space computations

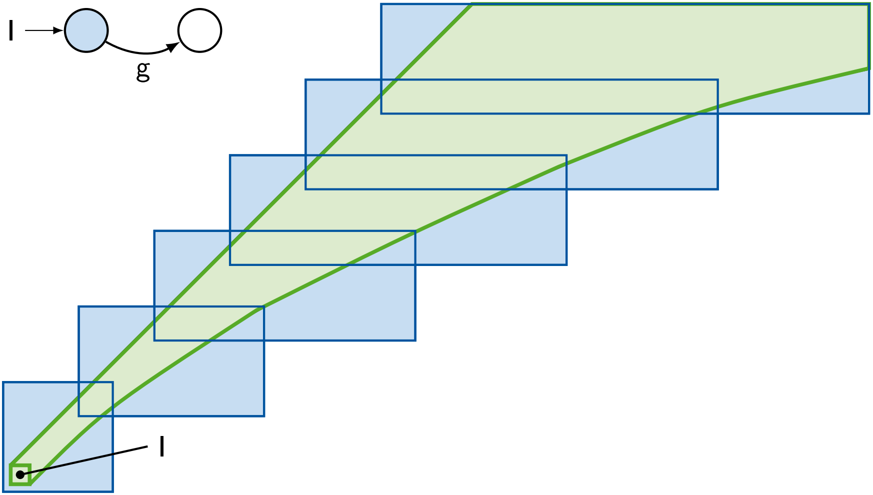
HyDRA techniques

1. Counterexample-guided abstraction refinement
2. Parallelization
3. Sub-space computations

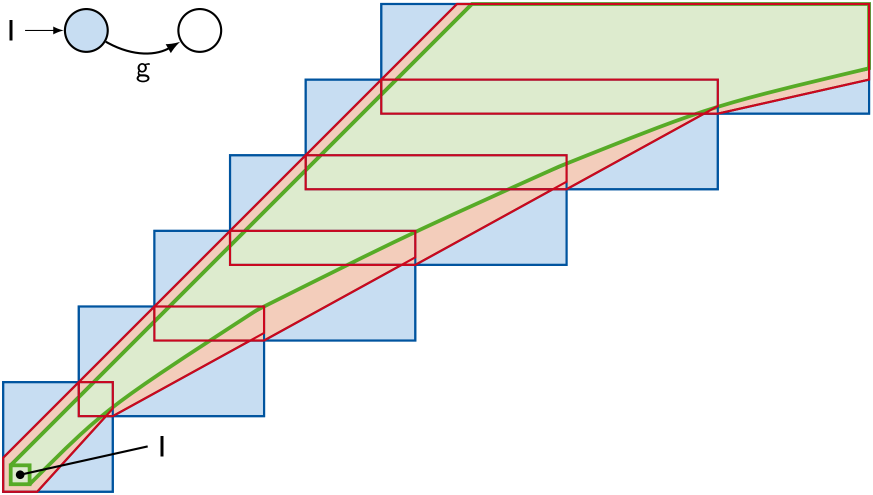
Choice of state set representation



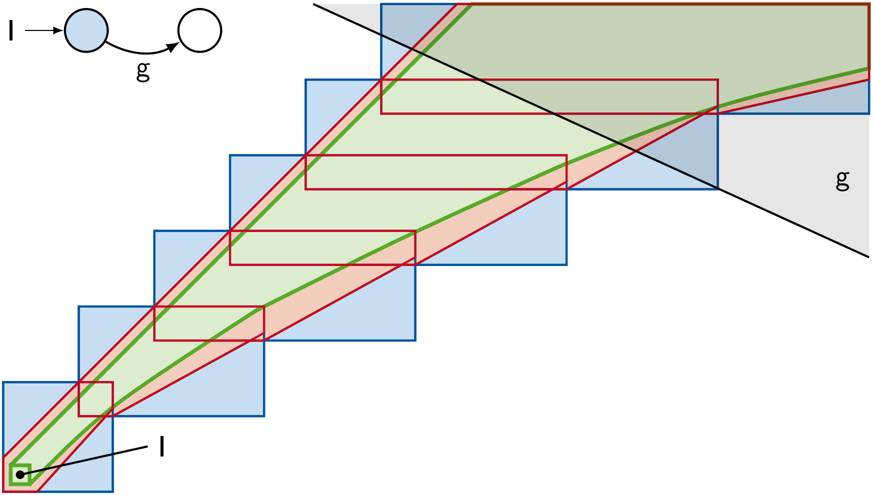
Choice of state set representation



Choice of state set representation

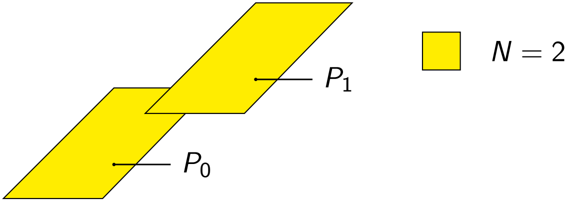


Choice of state set representation



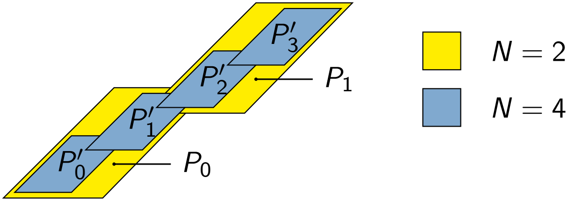
Time step length

Discretize time horizon T into N time segments:



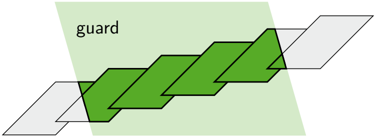
Time step length

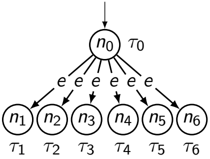
Discretize time horizon T into N time segments:



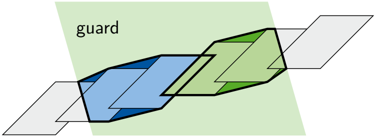
Discrete successors: Aggregation & clustering

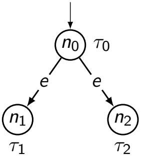
No aggregation/clustering



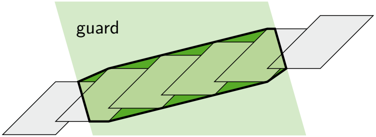


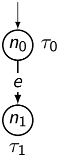
Clustering





Aggregation





Analysis parameters

Parameters such as

- state set representation,
- time step size δ,
- aggregation/clustering,
- ...

influence precision as well as computational effort.

Analysis parameters

Parameters such as

- state set representation,
- time step size δ,
- aggregation/clustering,
- ...

influence precision as well as computational effort.

Too precise $\rightarrow$ might not terminate within acceptable time
Too coarse $\rightarrow$ might fail to show safety

Analysis parameters

Parameters such as

- state set representation,
- time step size δ,
- aggregation/clustering,
- ...

influence precision as well as computational effort.

Too precise $\rightarrow$ might not terminate within acceptable time
Too coarse $\rightarrow$ might fail to show safety

Idea of dynamic configurations:
Use "coarse" configurations for fast analysis.
Use more "precise" configurations to falsify spurious counterexamples.

Analysis parameters

Parameters such as

- state set representation,
- time step size δ,
- aggregation/clustering,
- ...

influence precision as well as computational effort.

Too precise $\rightarrow$ might not terminate within acceptable time
Too coarse $\rightarrow$ might fail to show safety

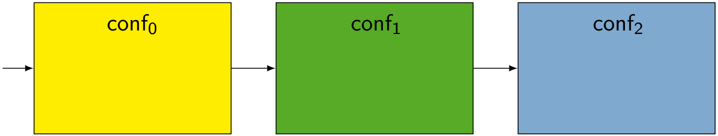
Idea of dynamic configurations:
Use "coarse" configurations for fast analysis.
Use more "precise" configurations to falsify spurious counterexamples.

Some tools use adaptive methods, but they are hard-wired and restricted to certain parameters.

HyDRA's CEGAR approach

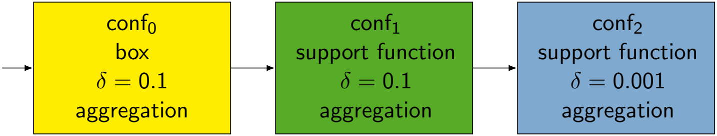
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



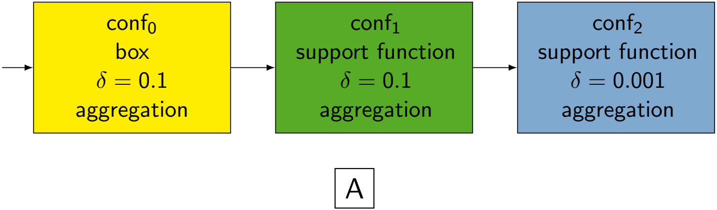
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



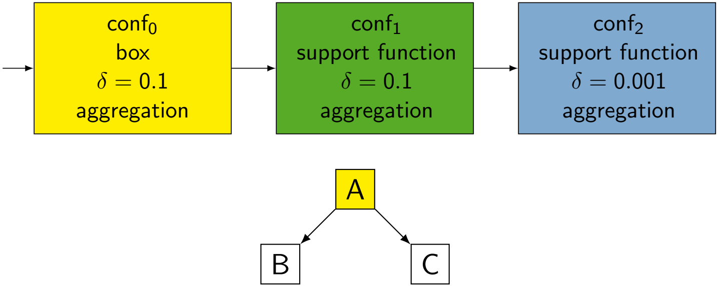
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



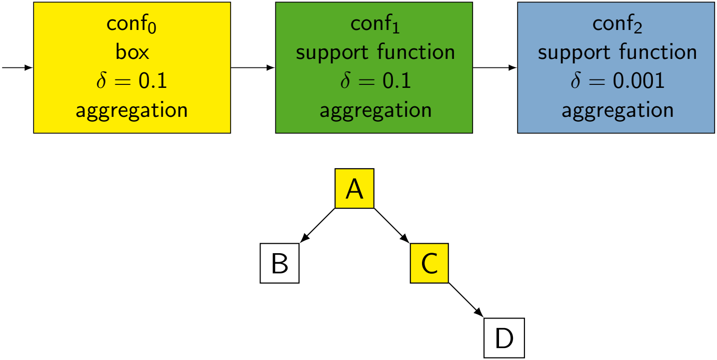
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



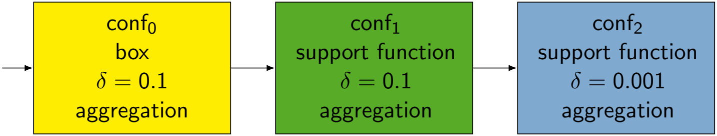
HyDRA's CEGAR approach

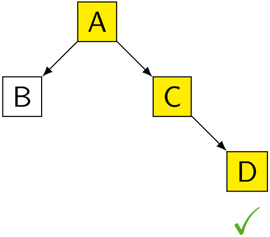
Strategy: Finite sequence of parameter configurations



HyDRA's CEGAR approach

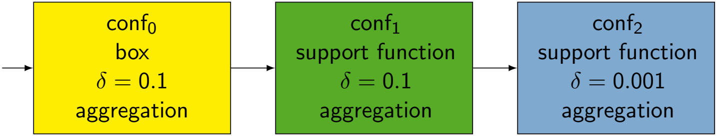
Strategy: Finite sequence of parameter configurations

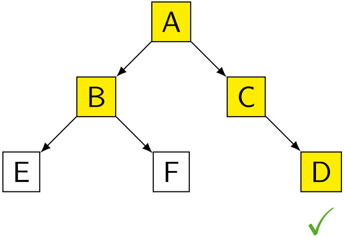




HyDRA's CEGAR approach

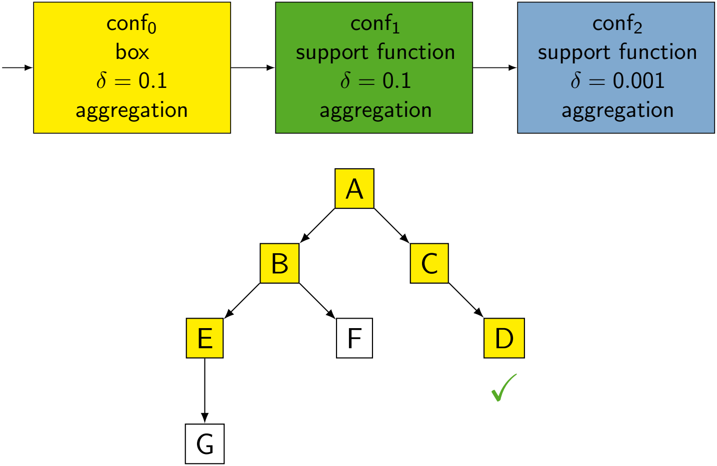
Strategy: Finite sequence of parameter configurations





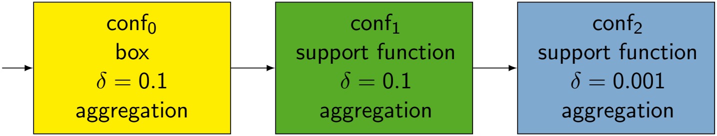
HyDRA's CEGAR approach

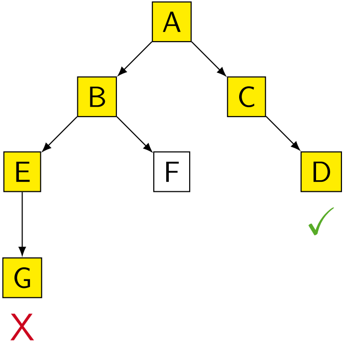
Strategy: Finite sequence of parameter configurations



HyDRA's CEGAR approach

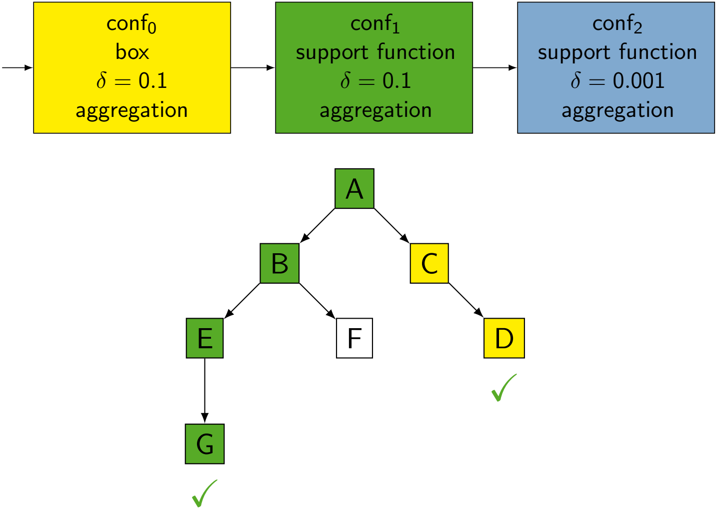
Strategy: Finite sequence of parameter configurations





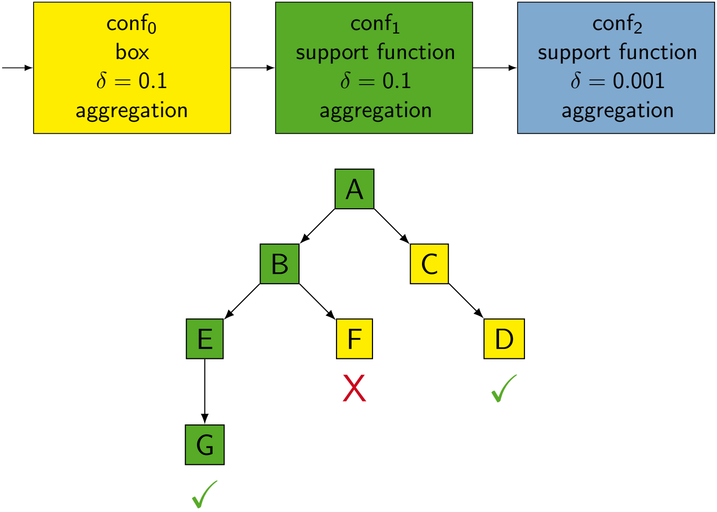
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



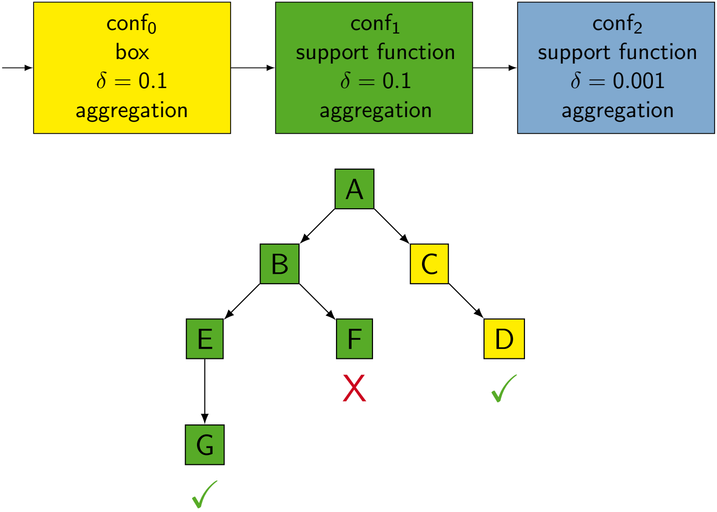
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



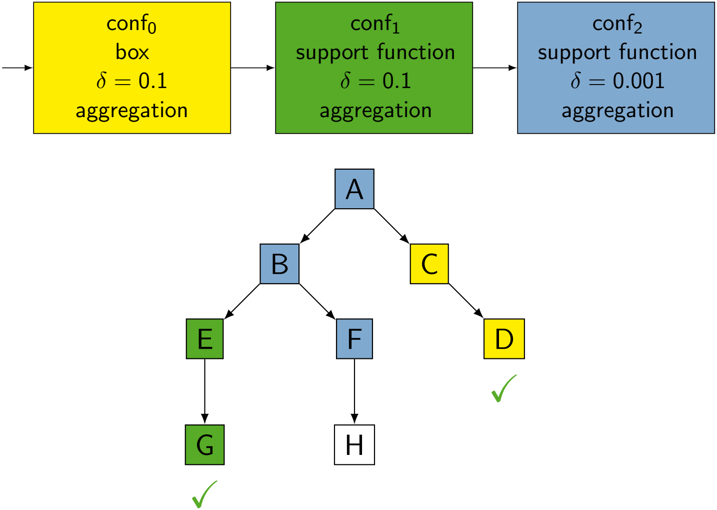
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



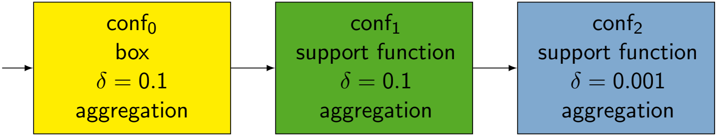
HyDRA's CEGAR approach

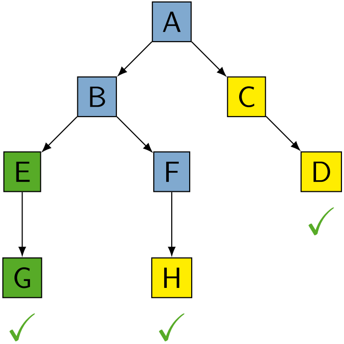
Strategy: Finite sequence of parameter configurations



HyDRA's CEGAR approach

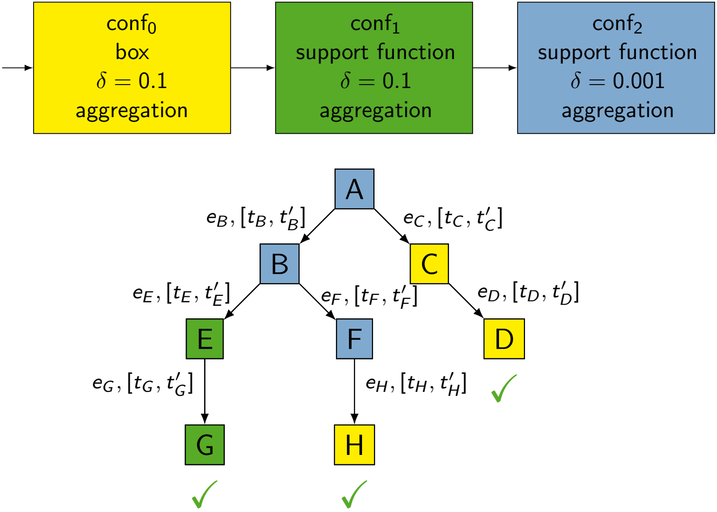
Strategy: Finite sequence of parameter configurations





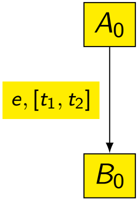
HyDRA's CEGAR approach

Strategy: Finite sequence of parameter configurations



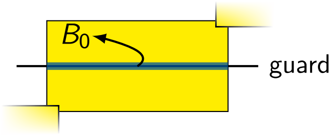
Dynamic search tree structure

No aggregation/clustering



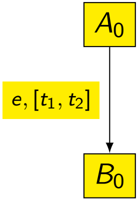
Time interval:

Flowpipe:



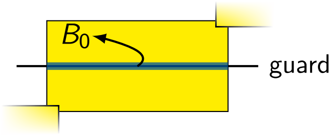
Dynamic search tree structure

No aggregation/clustering – reduce time step length

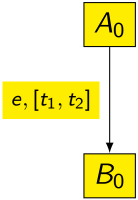


Time interval:

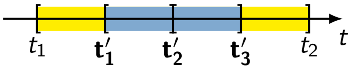
Flowpipe:



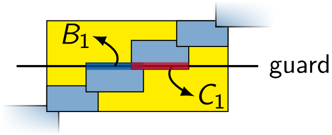
No aggregation/clustering – reduce time step length



Time interval:

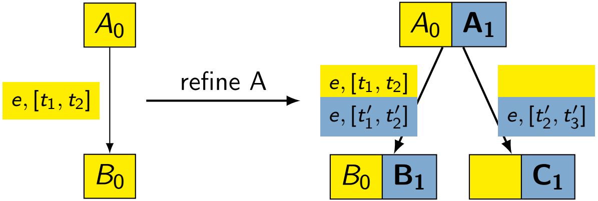


Flowpipe:

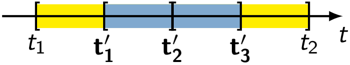


Dynamic search tree structure

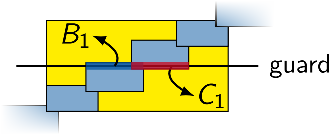
No aggregation/clustering – reduce time step length



Time interval:

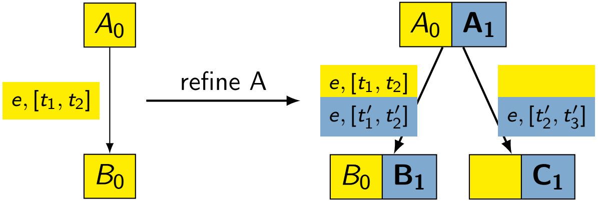


Flowpipe:

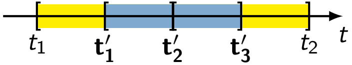


Dynamic search tree structure

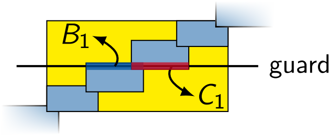
No aggregation/clustering – reduce time step length



Time interval:

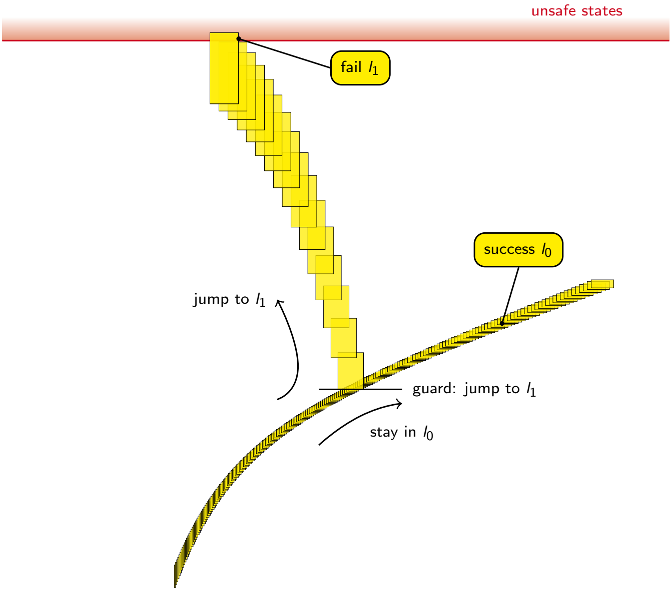


Flowpipe:

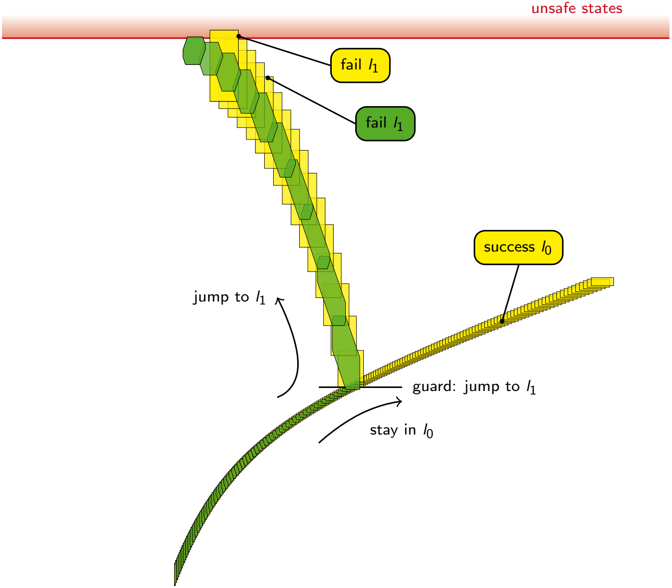


Reuse and **refine** transition timing information.

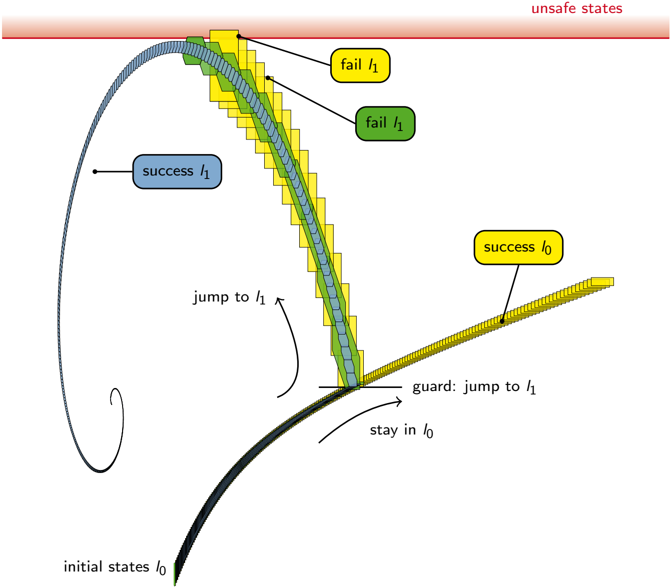
Example computation



Example computation



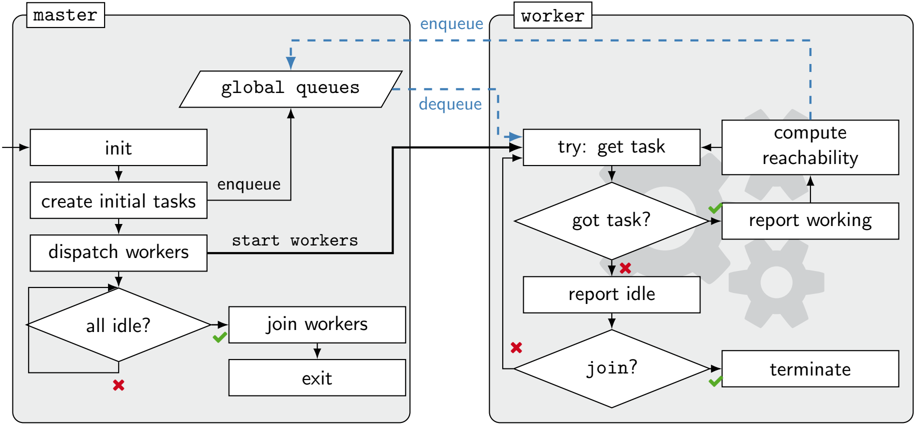
Example computation



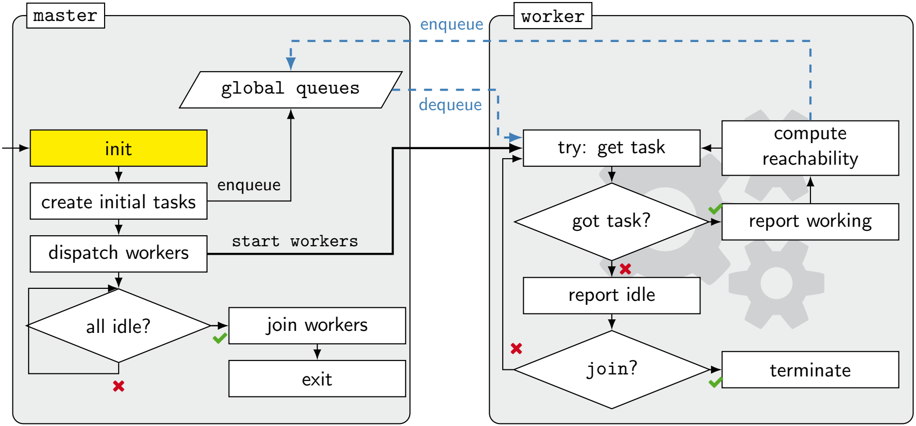
HyDRA techniques

1. Counterexample-guided abstraction refinement
2. Parallelization
3. Sub-space computations

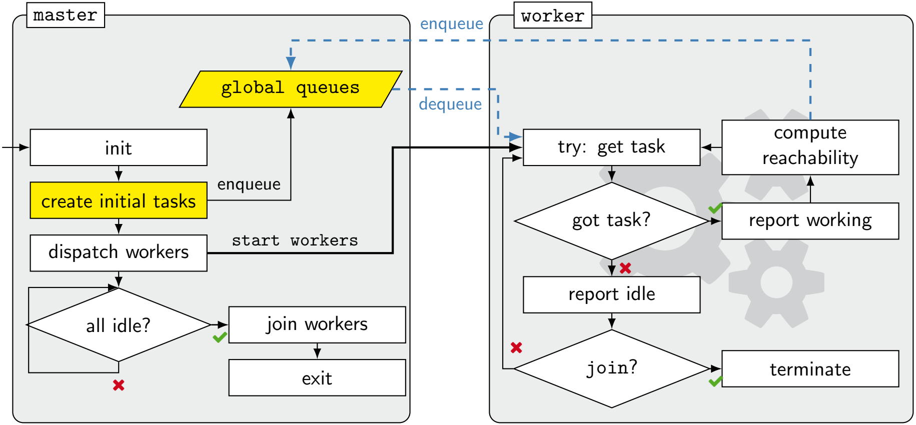
Parallel CEGAR



Parallel CEGAR



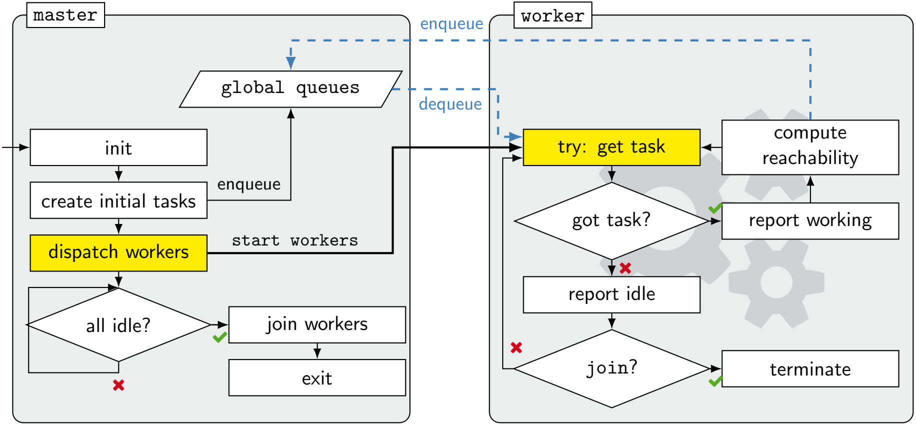
Parallel CEGAR



Queues: (1) non-refinement (2) refinement

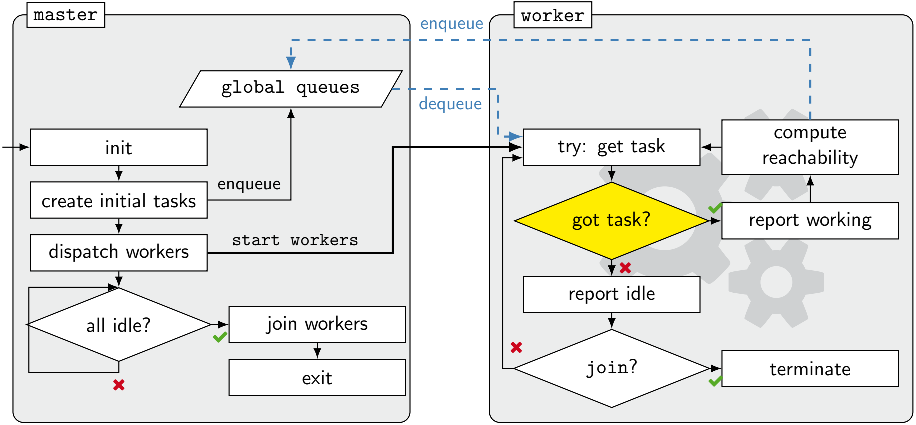
Tasks: node, refinement level, symbolic path

Parallel CEGAR

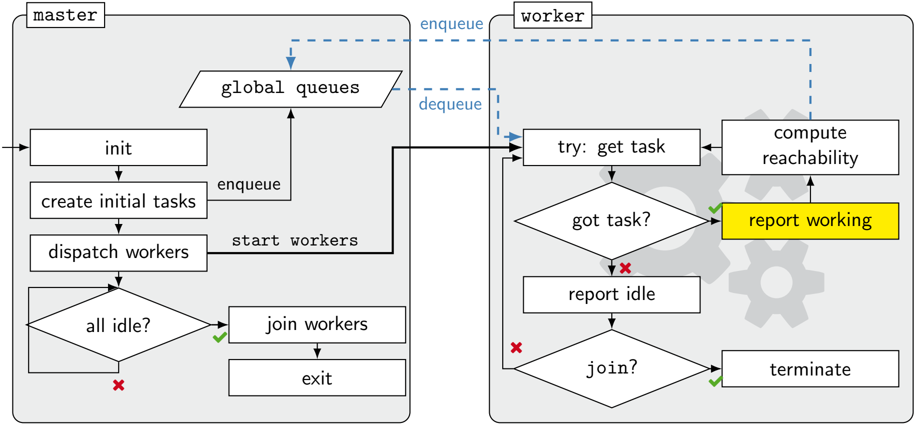


Synchronization on global queues.

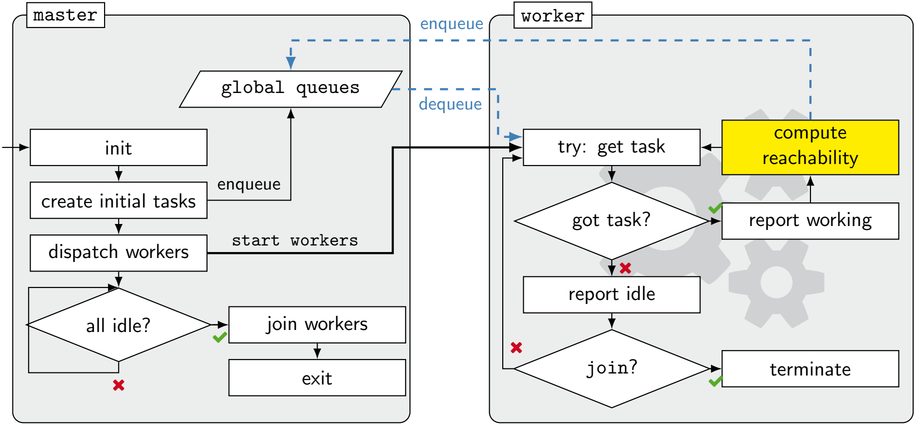
Parallel CEGAR



Parallel CEGAR

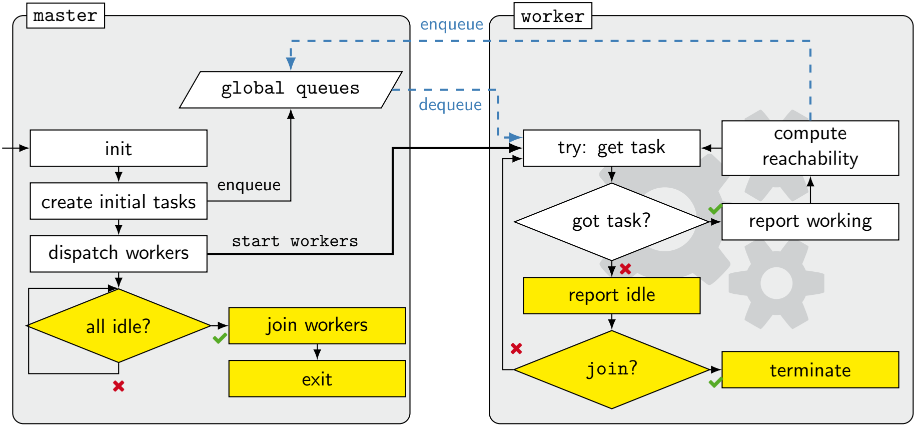


Parallel CEGAR

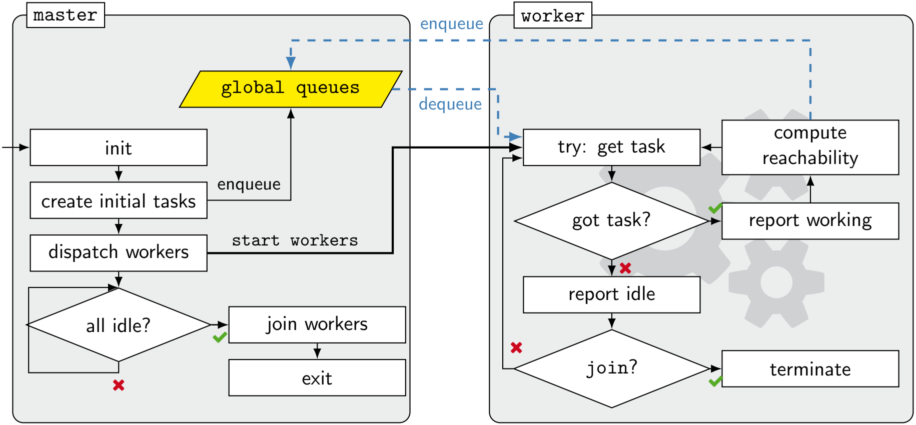


Synchronization on nodes for refinements.

Parallel CEGAR

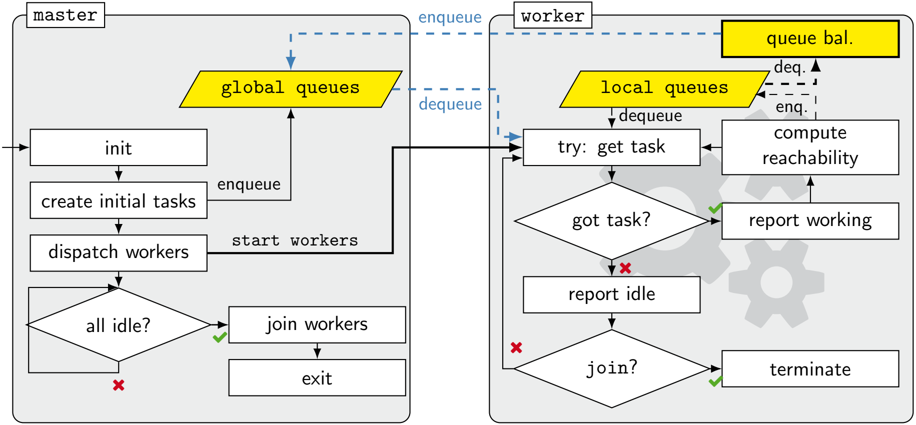


Parallel CEGAR



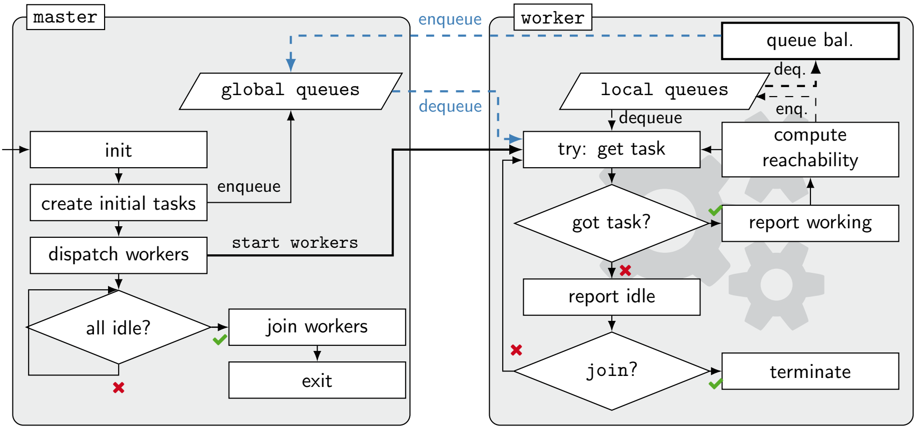
Use balanced local and global queues.

Parallel CEGAR



Use balanced local and global queues.

Parallel CEGAR



HyDRA techniques

1. Counterexample-guided abstraction refinement
2. Parallelization
3. Sub-space computations

HyPro application: Sub-space computations

- Motivation: PLC-controlled plants
- High-dimensional models

HyPro application: Sub-space computations

- Motivation: PLC-controlled plants
- High-dimensional models
- Relevant number of discrete variables
- Clocks for cycle synchronisation

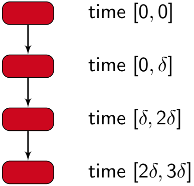
HyPro application: Sub-space computations

- Motivation: PLC-controlled plants
- High-dimensional models
- Relevant number of discrete variables
- Clocks for cycle synchronisation

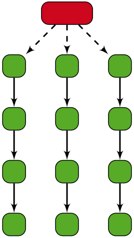
Idea:

- Partition variable set $\rightsquigarrow$ sub-spaces
- Compute reachability in sub-spaces
- Synchronise on time

Usual:



Wanted:



3. Sub-space computations

Partition the variable set into syntactically independent subsets.

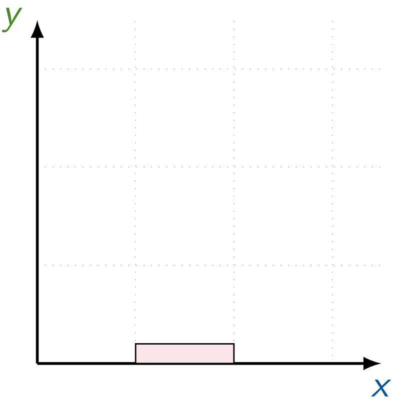
$$(\dot{x}, \dot{y}) = A \cdot (x, y)^T + B \cdot u \;\longrightarrow\; \dot{x} = A_x \cdot x^T + B_x \cdot u \;\wedge\; \dot{y} = A_y \cdot y^T + B_y \cdot u$$

$$Inv \;\longrightarrow\; Inv_x \;\wedge\; Inv_y$$

$$guard \;\longrightarrow\; guard_x \;\wedge\; guard_y$$

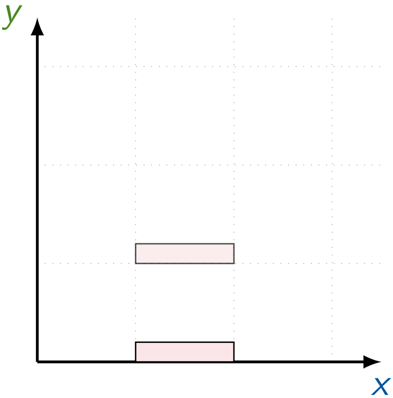
$$reset \;\longrightarrow\; reset_x \;\wedge\; reset_y$$

Discrete variables



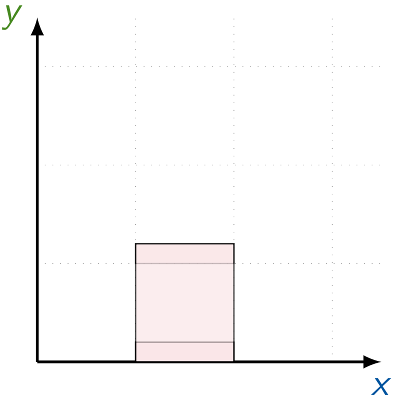
$$\dot{x} = 0$$
$$\dot{y} = 1$$

Discrete variables



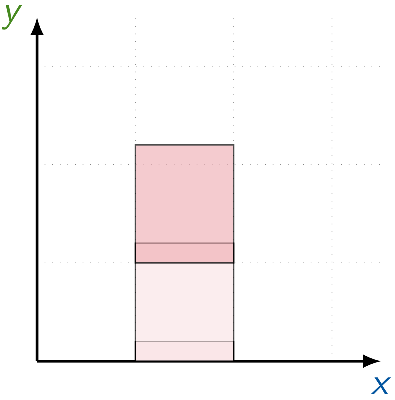
$$\dot{x} = 0$$
$$\dot{y} = 1$$

Discrete variables



$$\dot{x} = 0$$
$$\dot{y} = 1$$

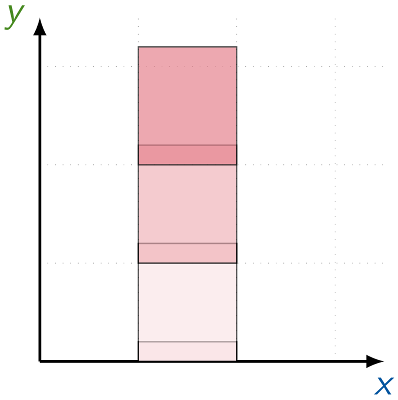
Discrete variables



$\dot{x} = 0$

$\dot{y} = 1$

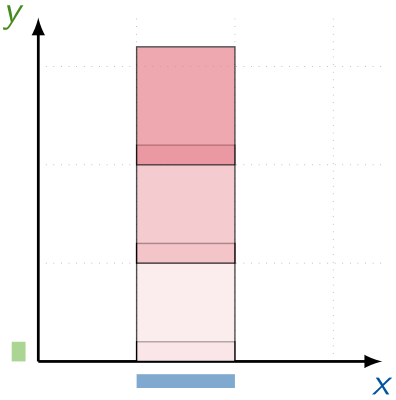
Discrete variables



$\dot{x} = 0$

$\dot{y} = 1$

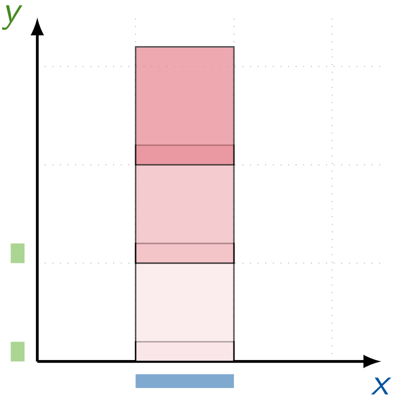
Discrete variables



$\dot{x} = 0$

$\dot{y} = 1$

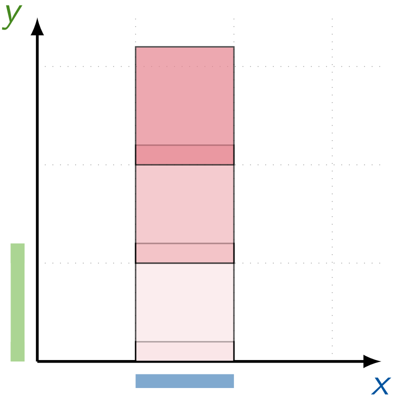
Discrete variables



$\dot{x} = 0$

$\dot{y} = 1$

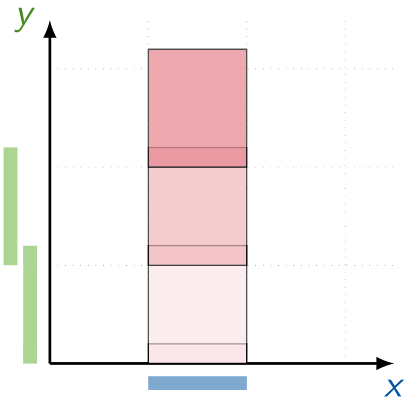
Discrete variables



$\dot{x} = 0$

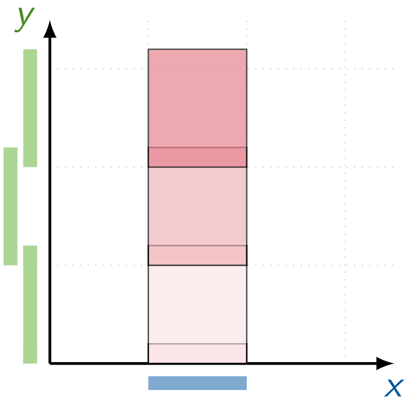
$\dot{y} = 1$

Discrete variables



$$\dot{x} = 0$$
$$\dot{y} = 1$$

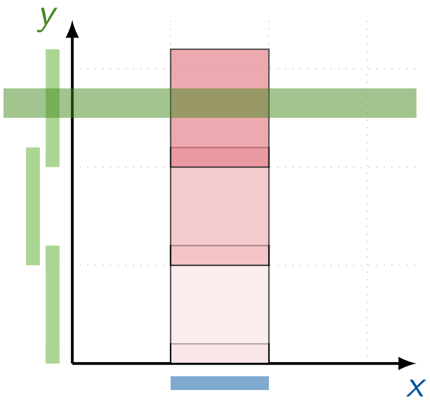
Discrete variables



$\dot{x} = 0$

$\dot{y} = 1$

Discrete variables

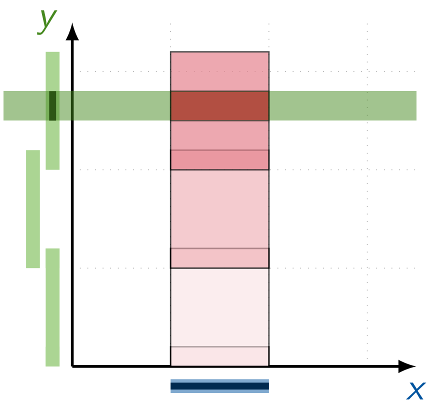


$2.5 \leq y \leq 2.8$

$\dot{x} = 0$

$\dot{y} = 1$

Discrete variables

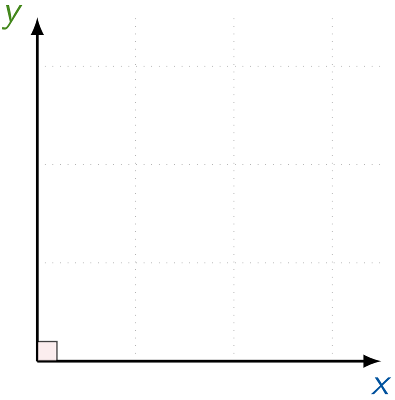


$2.5 \leq y \leq 2.8$

$\dot{x} = 0$

$\dot{y} = 1$

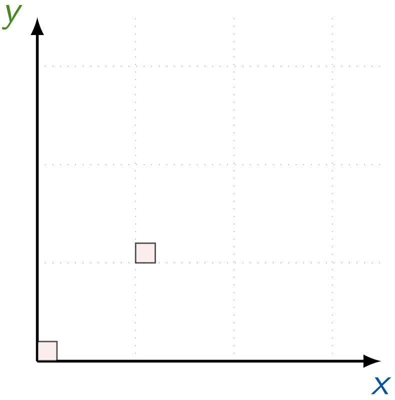
Representation: Boxes



$\dot{x} = 1$

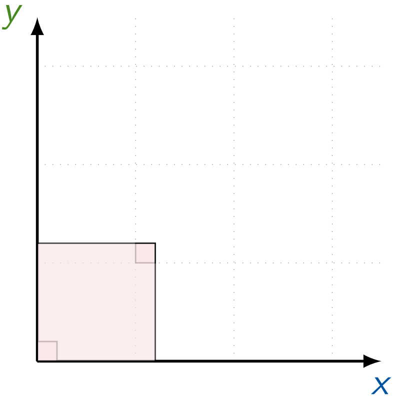
$\dot{y} = 1$

Representation: Boxes



$$\dot{x} = 1$$
$$\dot{y} = 1$$

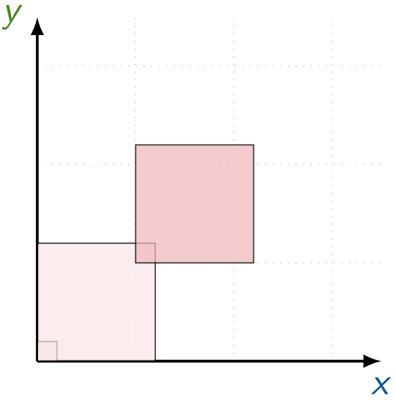
Representation: Boxes



$\dot{x} = 1$

$\dot{y} = 1$

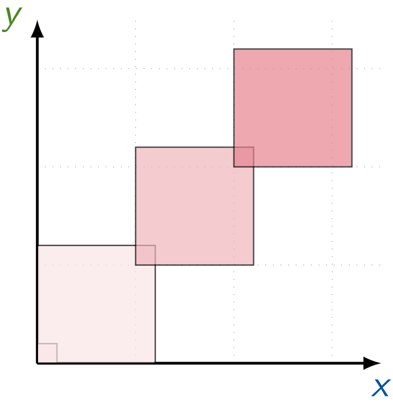
Representation: Boxes



$\dot{x} = 1$

$\dot{y} = 1$

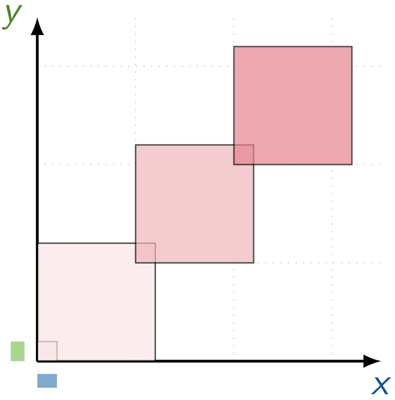
Representation: Boxes



$\dot{x} = 1$

$\dot{y} = 1$

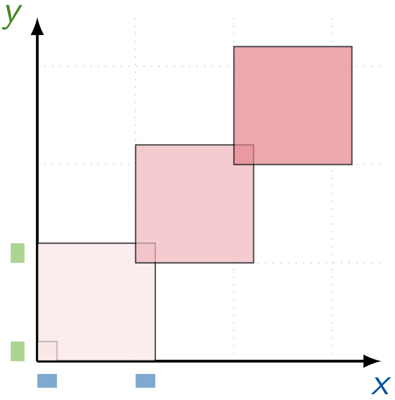
Representation: Boxes



$\dot{x} = 1$

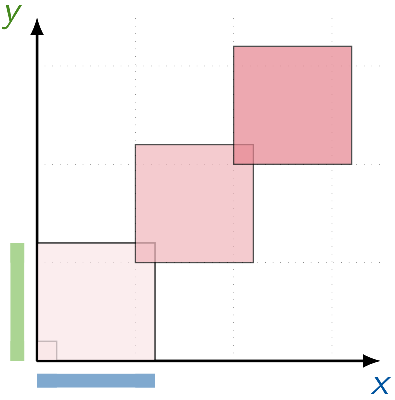
$\dot{y} = 1$

Representation: Boxes



$$\dot{x} = 1$$
$$\dot{y} = 1$$

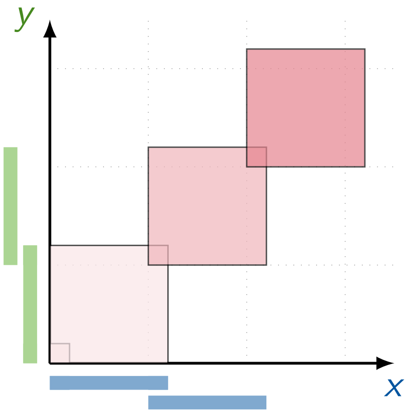
Representation: Boxes



$\dot{x} = 1$

$\dot{y} = 1$

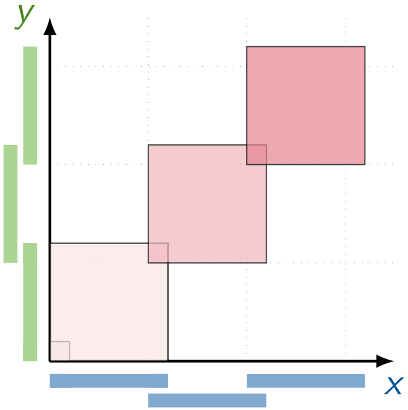
Representation: Boxes



$\dot{x} = 1$

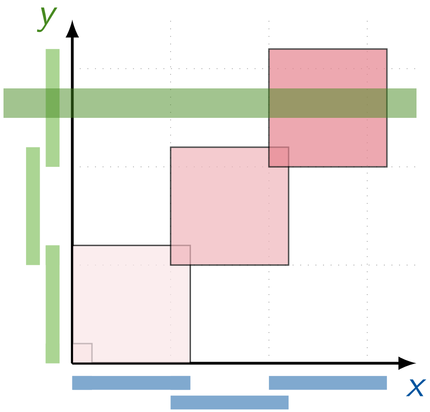
$\dot{y} = 1$

Representation: Boxes



$\dot{x} = 1$
$\dot{y} = 1$

Representation: Boxes

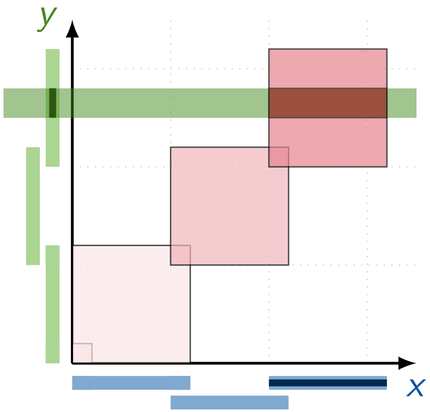


$2.5 \leq y \leq 2.8$

$\dot{x} = 1$

$\dot{y} = 1$

Representation: Boxes

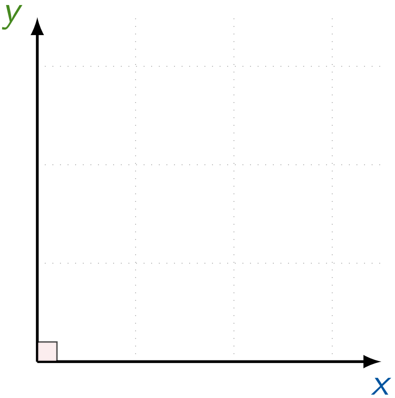


$2.5 \leq y \leq 2.8$

$\dot{x} = 1$

$\dot{y} = 1$

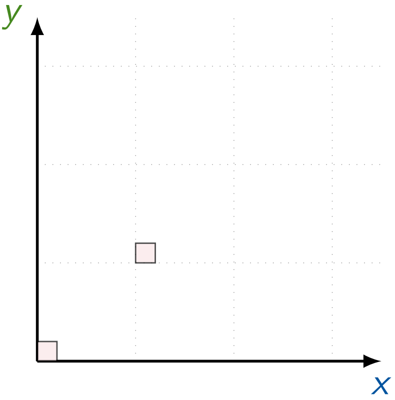
Representation: Polytopes



$\dot{x} = 1$

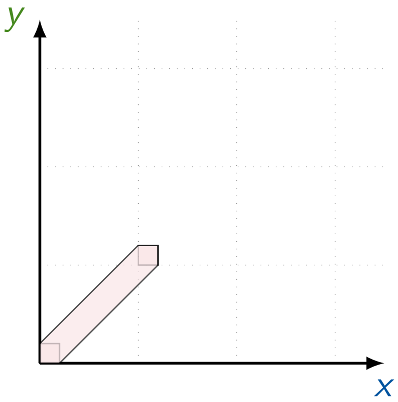
$\dot{y} = 1$

Representation: Polytopes



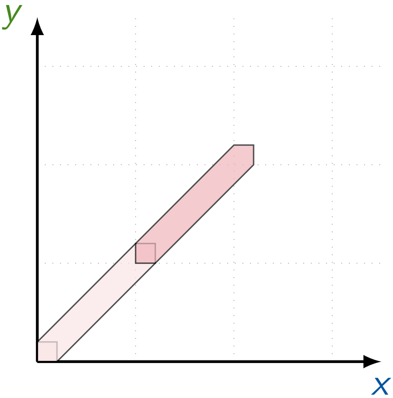
$$\dot{x} = 1$$
$$\dot{y} = 1$$

Representation: Polytopes



$$\dot{x} = 1$$
$$\dot{y} = 1$$

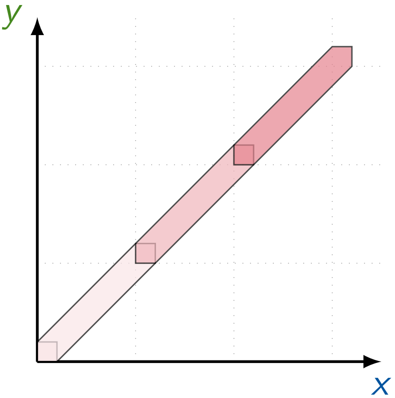
Representation: Polytopes



$\dot{x} = 1$

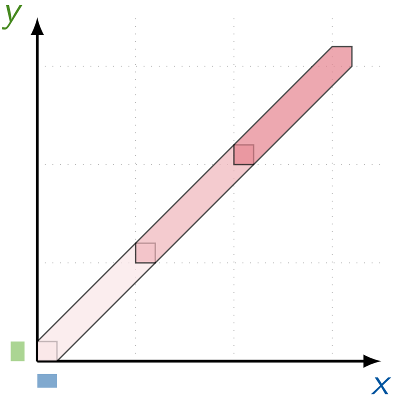
$\dot{y} = 1$

Representation: Polytopes



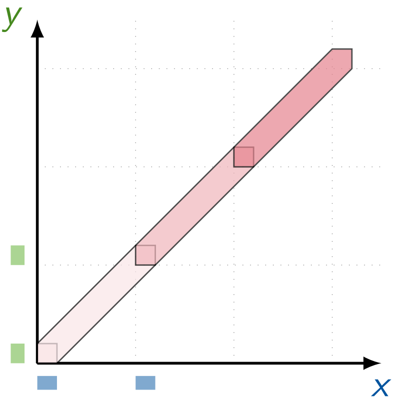
$\dot{x} = 1$
$\dot{y} = 1$

Representation: Polytopes



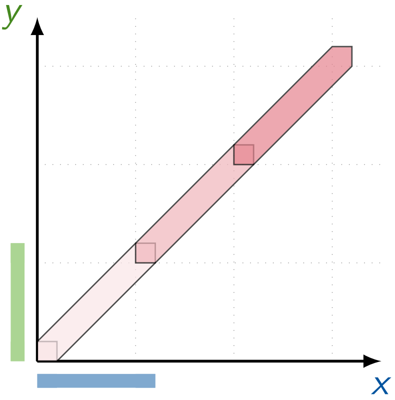
$$\dot{x} = 1$$
$$\dot{y} = 1$$

Representation: Polytopes



$$\dot{x} = 1$$
$$\dot{y} = 1$$

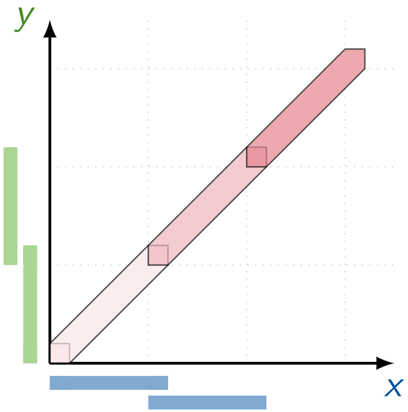
Representation: Polytopes



$\dot{x} = 1$

$\dot{y} = 1$

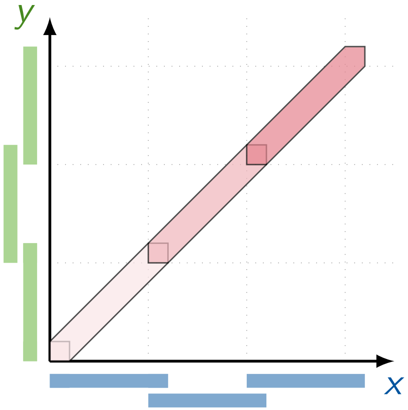
Representation: Polytopes



$\dot{x} = 1$

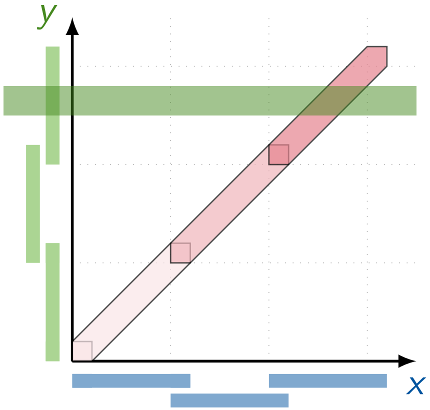
$\dot{y} = 1$

Representation: Polytopes



$$\dot{x} = 1$$
$$\dot{y} = 1$$

Representation: Polytopes

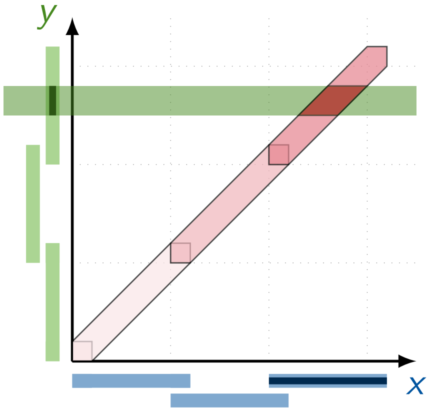


$2.5 \leq y \leq 2.8$

$\dot{x} = 1$

$\dot{y} = 1$

Representation: Polytopes



$2.5 \leq y \leq 2.8$

$\dot{x} = 1$

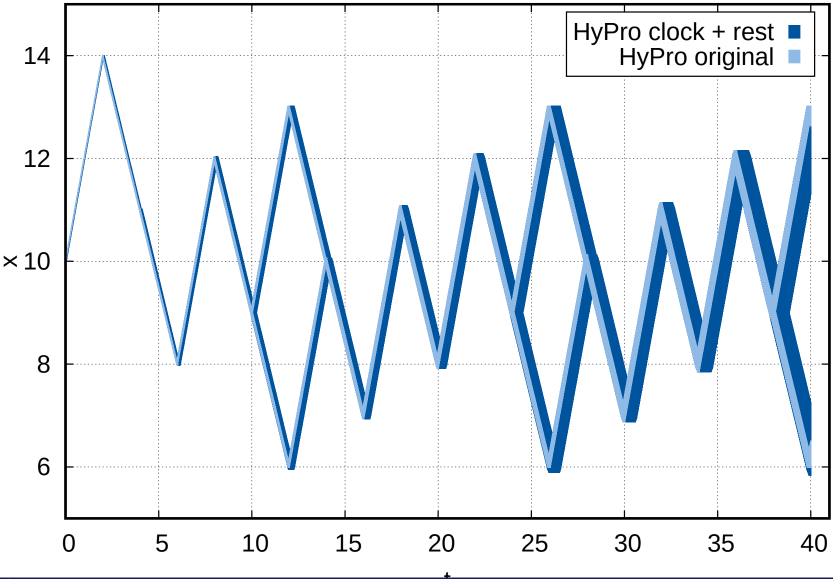
$\dot{y} = 1$

Pros and cons

+ Reduced computational effort
+ Subspace-local configurations are possible
+ Even subspace-local reachability analysis algorithms are possible

- Additional over-approximation

Leaking tank example



Conclusion

- HyPro: open-source programming library
- Available at `https://github.com/hypro/hypro`
- State set representations for the implementation of hybrid systems reachability analysis algorithms
- Exact as well as inexact number representations
- Flexibility to deviate from standard methods
- HyDRA: HyPro-based reachability analysis
  - Counterexample-guided abstraction refinement
  - Parallelization
  - Sub-space computations

HyPro benchmark collection [NFM'15]

http://ths.rwth-aachen.de/research/projects/hypro/benchmarks-of-continuous-and-hybrid-systems/

Two tanks

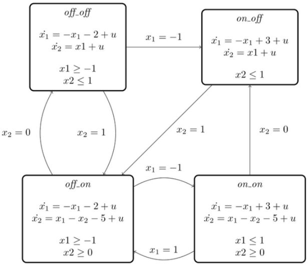
Classification

Type	Continuous dynamics	Guards & Invariants	Resets
hybrid	linear polynomial	hyperplane & half-space	identity

Model Description (flow*_files, spaceEx_files)

The considered benchmark presented in Fig.1(a) consists of two tanks. The liquid in the first tank comes from two outside sources: a constant inflow source and a second source equipped with a controlled valve valve1, with flows Q_0 and Q_1 respectively. A drain placed at the bottom of tank 1 allows the liquid to flow into tank 2 with flow Q_A.

Tank 2 is itself equipped with two drains. In the first one a pump is placed to assure a constant liquid outflow Q_B whereas the flow in the second one Q_2 is controlled by an electro-valve valve2.Both valves can take the states On/Off. This results consequently in four possible discrete modes for the hybrid automaton. The liquid levels in tank i is given by x_i.



$$\dot{x}_1 = \begin{cases} -x_1 - 2 + [-0.1, 0.1] & \text{if } valve_1 \text{ is Off} \\ -x_1 + 3 + [-0.1, 0.1] & \text{if } valve_1 \text{ is On} \end{cases}$$

$$\dot{x}_2 = \begin{cases} x_1 + [-0.1, 0.1] & \text{if } valve_2 \text{ is Off} \\ x_1 - x_2 - 5 + [-0.1, 0.1] & \text{if } valve_2 \text{ is On} \end{cases}$$

Reachability Setting

Settings in the model files

We consider an initial set of

- $x_1 \in [1.5, 2.5]$
- $x_2 = 1$

and the starting location off_off, a time horizon $T = 5s$, and a time step $r = 0.1s$. The set of bad states are all states, where $x_2 <= -0.7$.

Results

The results given in Fig. 2 shows tightness difference between the two obtained flowpipes although both proposed tools are based on same reachability technique based on support functions. Fig.2 shows the resulting flowpipe given the abovementioned parameters using in a) HyReach, and in b) SpaceEx.

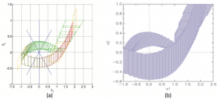


Figure 2: Flowpipe of the two-tank system

References

[1] J. Lygeros. *Lecture notes on hybrid systems.* Technical Report, 2004.

[2] I. A. Hiskens. *Stability of limit cycles in hybrid systems.* In Proceedings of the 34th Hawaii International Conference on System Sciences (HICSS'01), pages 163-328, IEEE, 2001.

[3] A. Girard. *Reachability of Uncertain Linear Systems Using Zonotopes.* In Proceedings

Further challenges

- State set representation: context-sensitive approaches, non-convex representations, under-approximation
- Precision: automated dynamic error reduction
- Large uncertainties / initial sets
- Zeno behaviour
- Unbounded verification: efficient fixed-point recognition
- More expressive models: non-convex invariants, urgent transitions/locations, communication, random behaviour, hierarchical models
- Counterexamples
- Compositionality

- Standard input language, more benchmarks, competitions