Generalised Branch-and-Bound

and its Application in SAT Modulo Nonlinear Integer Arithmetic

Gereon Kremer
in cooperation with Florian Corzilius and Erika Ábrahám

RWTH Aachen University, Germany
LuFG Theory of Hybrid Systems

September 20, 2016

SC^2 (link to EU Project)

Satisfiability Checking and Symbolic Computation

Bridging Two Communities to Solve Real Problems

SUMMARY

The use of advanced methods to solve practical and industrially relevant problems by computers has a long history. Symbolic Computation is concerned with the algorithmic determination of exact solutions to complex mathematical problems; and more recent developments in the area of Satisfiability Checking are starting to tackle similar problems but with different algorithmic and technological solutions.

Though both communities have made remarkable progress in the last decades, they need to be further strengthened to tackle practical problems of ever increasing size and complexity. Their separate tools (computer algebra systems and SMT solvers) are urgently needed to address prevailing problems having a direct effect on our society. For example, Satisfiability Checking is an essential backend for assuring the security and the safety of computer systems. In various scientific areas, Symbolic Computation is able to deal with large mathematical problems far beyond the scope of pencil and paper solutions.

Currently the two communities are largely disjoint and unaware of the achievements of one another, despite there being strong reasons for them to discuss and collaborate, since they share many central interests. However, up to now, researchers from these two communities rarely interact, and also their tools lack common, mutual interfaces for unifying their strengths. Bridges between the communities in the form of common platforms and road-maps are necessary to initiate an exchange, and to support and direct their interaction. These are the main objectives of this CSA. We will initiate a wide range of activities to bring the two communities together, identify common challenges, propose joint standards, offer global events and bilateral visits, and so on. Besides the partners in our **EU Project**, **these people** have expressed an interest in these activities.

We believe that these activities will initiate cross-fertilization of both fields and bring mutual benefits. Combining the knowledge, experience and the technologies in these communities will enable the development of radically improved software tools.

News

On February 28th, 2016, the European Commission anounced an intention to fund our first project: the **SC-square**

Theory of Hybrid Systems Informatik 2

Contents

Basic framework

Satisfiability problem

Decide whether an existentially quantified formula $\varphi(x)$ is satisfiable

$$\exists x.\varphi(x) \equiv true$$

Basic framework

Satisfiability problem

Decide whether an existentially quantified formula $\varphi(x)$ is satisfiable

$$\exists x.\varphi(x) \equiv true$$

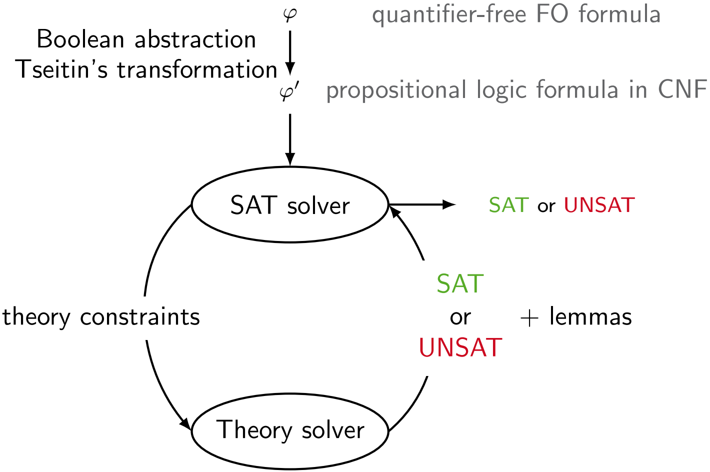
Satisfiability modulo theories

φ is from an existentially quantified first-order logic

- Fully automated solving
- Our focus: nonlinear integer arithmetic
- Example:

$$\exists x, y. y \geq \frac{x^2 - 2}{2} \wedge y \leq \frac{x}{2}$$

Fundamental idea: SAT vs. Theory



Branch-and-Bound [Land, Doig 1960] [Dutertre, de Moura 2006]

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`

Branch-and-Bound [Land, Doig 1960] [Dutertre, de Moura 2006]

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`
- If solution is integral: return `sat`

Theory of Hybrid Systems Informatik 2

Branch-and-Bound [Land, Doig 1960] [Dutertre, de Moura 2006]

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`
- If solution is integral: return `sat`
- If solution is fractional: return `unknown` and generate a lemma

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`
- If solution is integral: return `sat`
- If solution is fractional: return `unknown` and generate a lemma

What kind of lemma?

Branch-and-Bound [Land, Doig 1960] [Dutertre, de Moura 2006]

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`
- If solution is integral: return `sat`
- If solution is fractional: return `unknown` and generate a lemma

What kind of lemma?

- Must be a tautology in the theory
- Excludes only non-integral solutions

Branch-and-Bound [Land, Doig 1960] [Dutertre, de Moura 2006]

Fundamental idea: use a solver for $\mathbb{R}$ on problems from $\mathbb{Z}$

- Theory solver searches a solution for $C = \{c_1, ..., c_k\}$
- If no solution exists: return `unsat`
- If solution is integral: return `sat`
- If solution is fractional: return `unknown` and generate a lemma

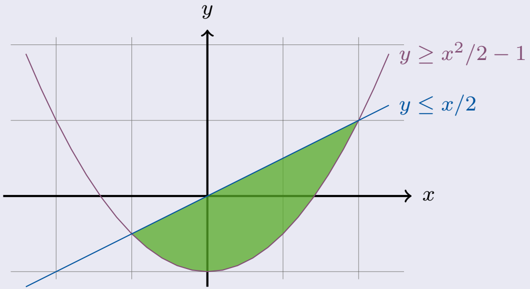
What kind of lemma?

- Must be a tautology in the theory
- Excludes only non-integral solutions
- $x \to \alpha_x$: $(c_1 \wedge ... \wedge c_k) \Rightarrow (x \leq \lfloor \alpha_x \rfloor \vee x \geq \lceil \alpha_x \rceil)$

hybr2d Theory of Hybrid Systems Informatik 2

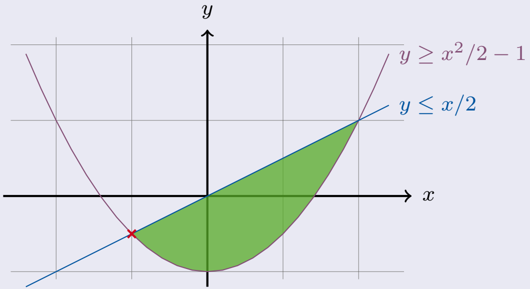
Branch-and-Bound – Example

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Branch-and-Bound – Example

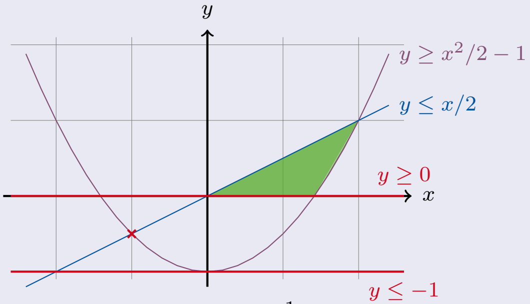
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



- Generate solution $x = -1, y = -\frac{1}{2}$

Branch-and-Bound – Example

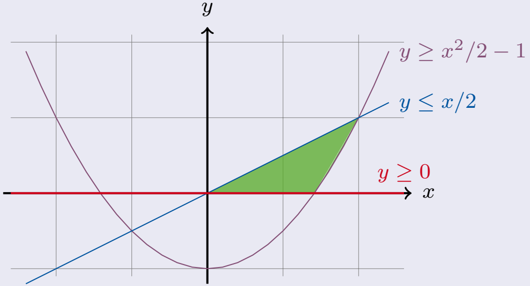
Example: $y \geq x^2/2 - 1 \land y \leq x/2$



- Generate solution $x = -1, y = -\frac{1}{2}$
- Not integral, generate $(c_1 \land c_2) \Rightarrow (y \leq -1 \lor y \geq 0)$

Branch-and-Bound – Example

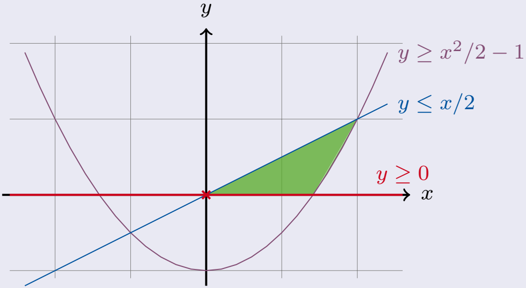
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



- Generate solution $x = -1, y = -\frac{1}{2}$
- Not integral, generate $(c_1 \wedge c_2) \Rightarrow (y \leq -1 \vee y \geq 0)$
- Pass lemma to SAT solver which selects one part

Branch-and-Bound – Example

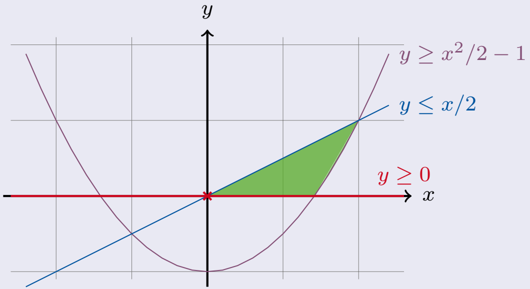
Example: $y \geq x^2/2 - 1 \land y \leq x/2$



- Generate solution $x = 0, y = 0$

Branch-and-Bound – Example

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

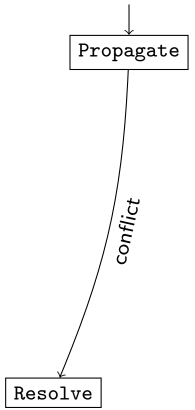


- Generate solution $x = 0, y = 0$
- Integral, return sat

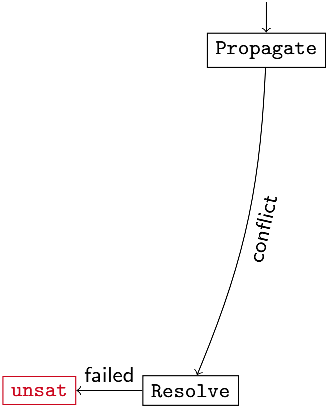
SMT solving with B&B

$$\downarrow$$
$$\boxed{\texttt{Propagate}}$$

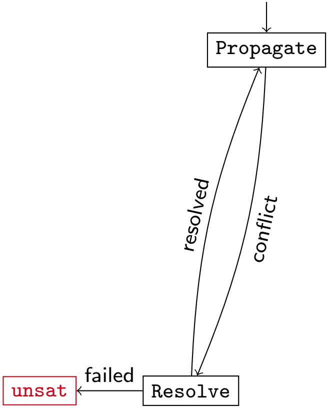
SMT solving with B&B



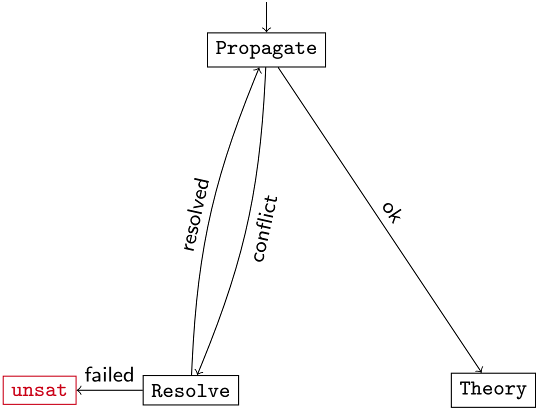
SMT solving with B&B



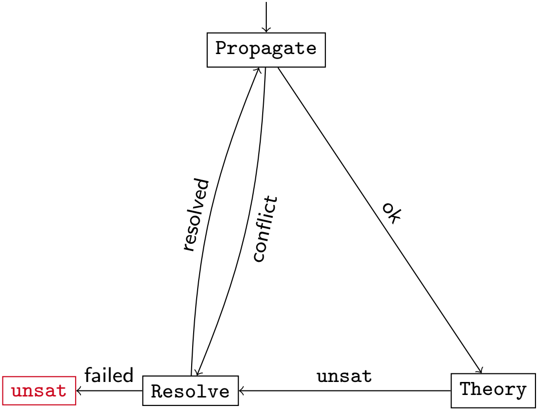
SMT solving with B&B



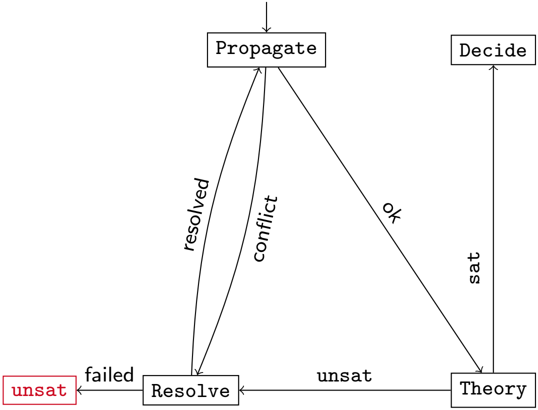
SMT solving with B&B



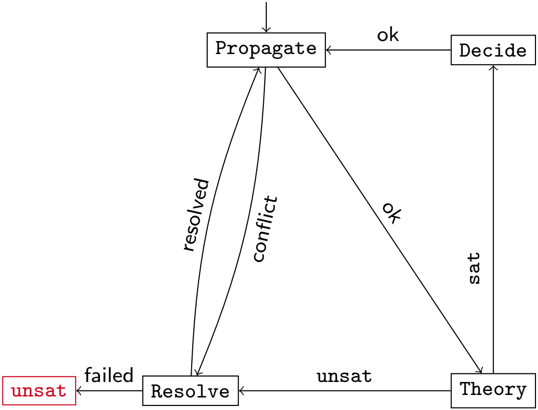
SMT solving with B&B



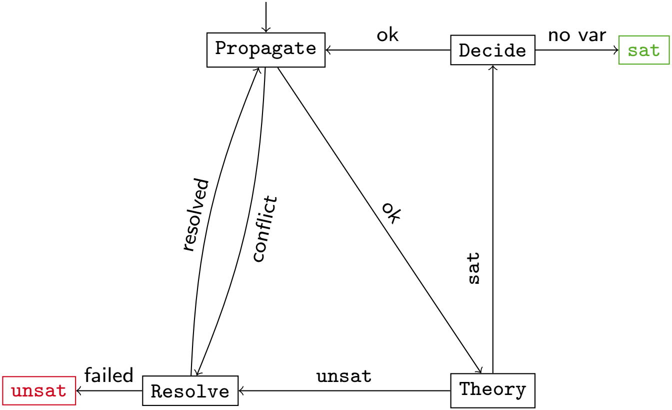
SMT solving with B&B



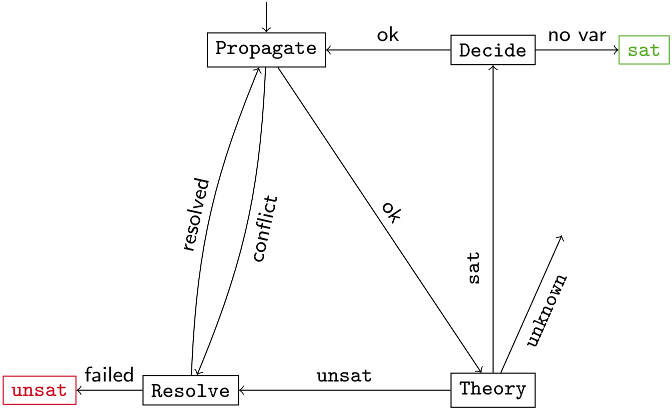
SMT solving with B&B



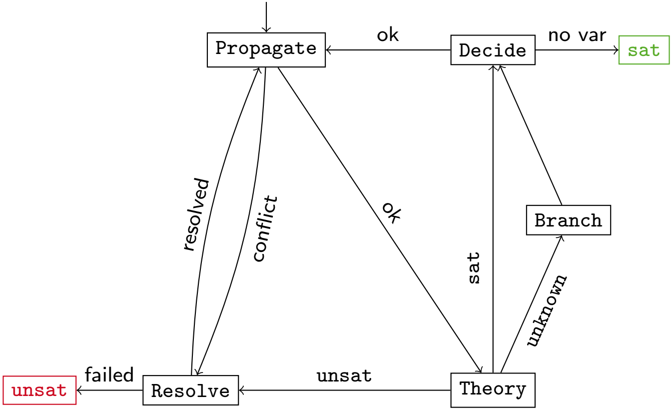
SMT solving with B&B



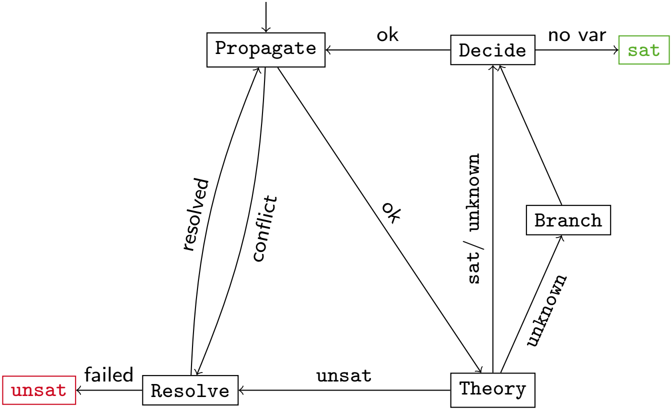
SMT solving with B&B



SMT solving with B&B



SMT solving with B&B



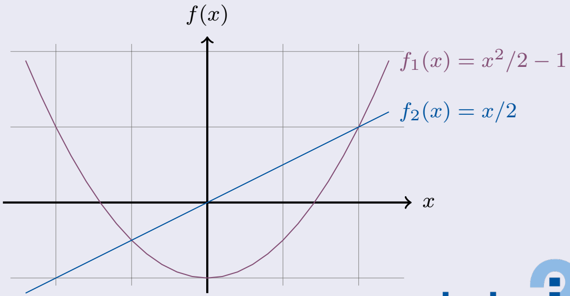
Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

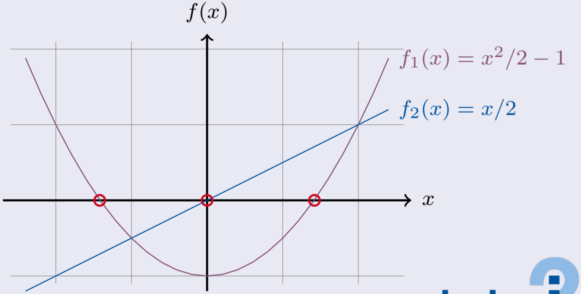
1-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

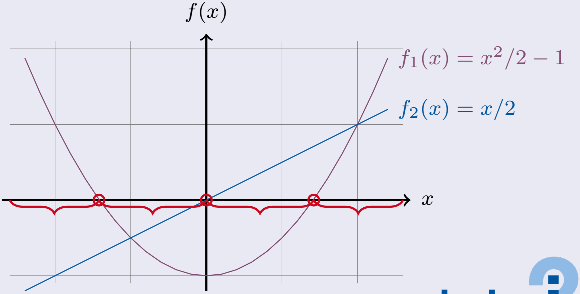
1-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

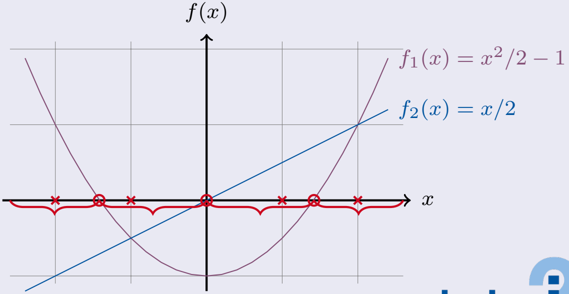
1-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

1-dimensional case



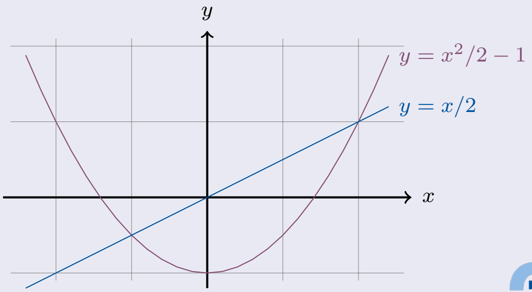
Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

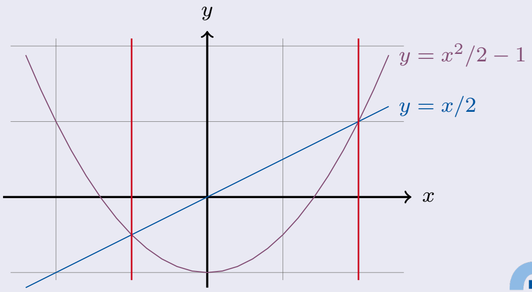
2-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

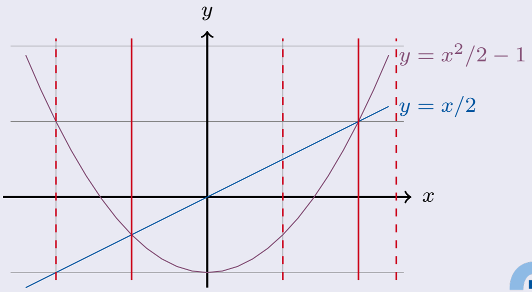
2-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

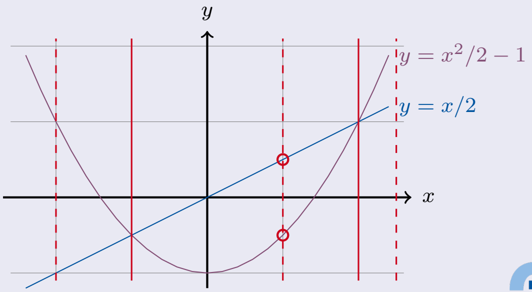
2-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

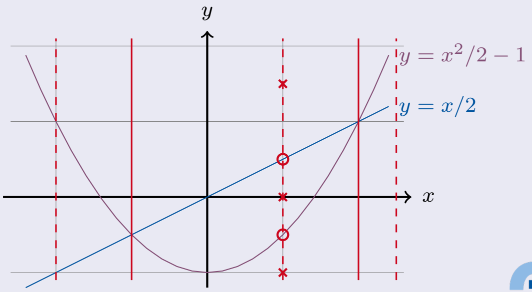
2-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

2-dimensional case



Solution strategy for Nonlinear Arithmetic

- Eliminate one variable at a time
- Separate solution space into satisfiability equivalent regions
- Check a single representative for each region

Open questions:

- How to generate roots and samples for 1-dimensional case?
- How to create k-dimensional points from $(k-1)$-dimensional points?
- How to make sure, that $(k-1)$-dimensional samples properly separate k-dimensional regions?

Virtual Substitution [Weispfenning 1988,1993] [Corzilius$^+$ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

Theory of Hybrid Systems Informatik 2

Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

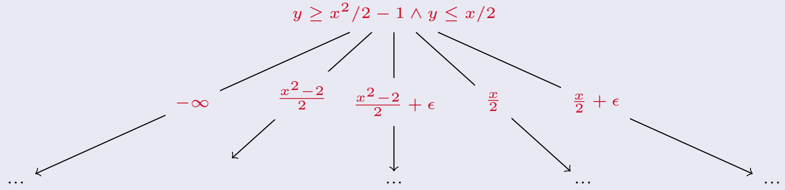
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$y \geq x^2/2 - 1 \wedge y \leq x/2$

Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

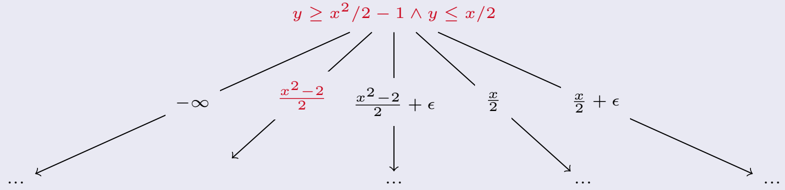


Theory of Hybrid Systems Informatik 2

Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

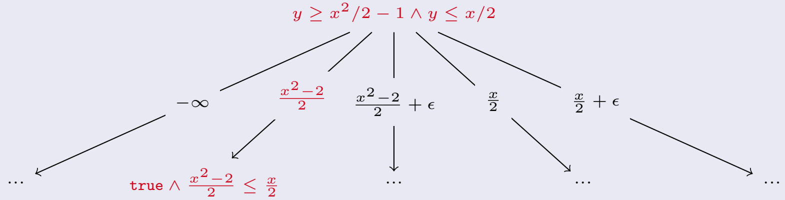
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

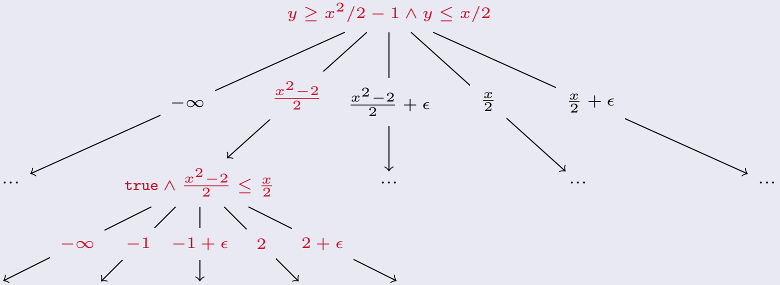
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

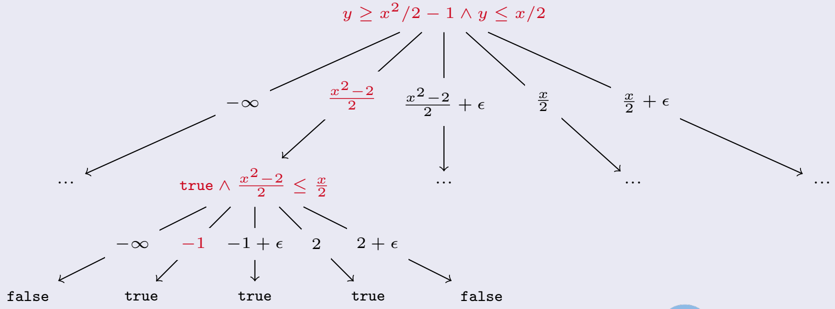
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

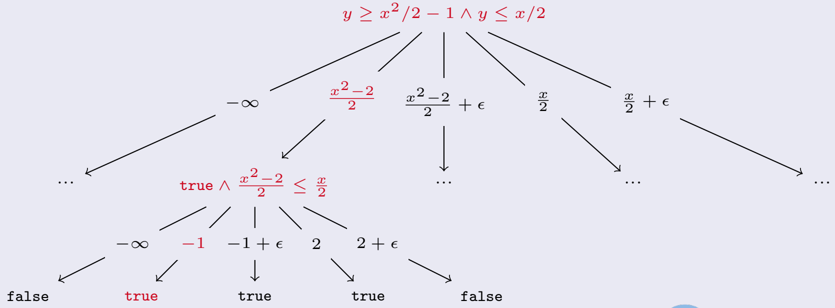
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Virtual Substitution [Weispfenning 1988,1993] [Corzilius^{+} 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

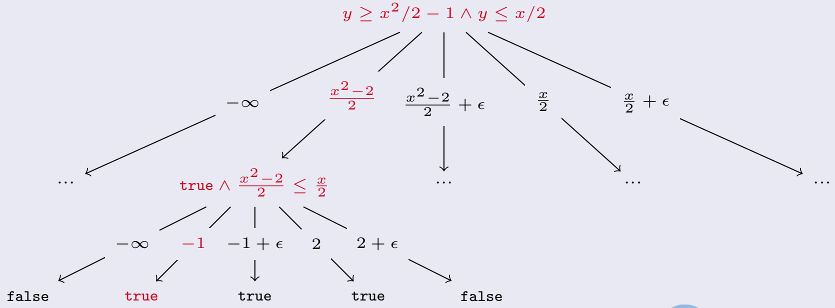
Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Virtual Substitution [Weispfenning 1988,1993] [Corzilius+ 2011]

- Uses symbolic representation of roots (e.g. using solution formula)
- Only for polynomials up to degree two*
- Substitutes roots into polynomials

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Satisfying solution: $x = -1, y = \frac{x^2-2}{2} = -\frac{1}{2}$

hybr2d Theory of Hybrid Systems Informatik 2

Virtual Substitution and B&B

- If no solution exists: return `unsat`
- Otherwise follow path upwards

$$y \geq x^2/2 - 1 \land y \leq x/2$$

$$\uparrow$$

$$\frac{x^2-2}{2}$$

$$\uparrow$$

$$\texttt{true} \land \frac{x^2-2}{2} \leq \frac{x}{2}$$

$$\uparrow$$

$$-1$$

$$\uparrow$$

$$\texttt{true}$$

Theory of Hybrid Systems Informatik 2

Virtual Substitution and B&B

- If no solution exists: return `unsat`
- Otherwise follow path upwards

- $x = -1$: integral

$$y \geq x^2/2 - 1 \wedge y \leq x/2$$
$$\uparrow$$
$$\frac{x^2-2}{2}$$
$$\uparrow$$
$$\texttt{true} \wedge \frac{x^2-2}{2} \leq \frac{x}{2}$$
$$\uparrow$$
$$-1$$
$$\uparrow$$
$$\texttt{true}$$

Virtual Substitution and B&B

- If no solution exists: return `unsat`
- Otherwise follow path upwards

- $x = -1$: integral
- $y = \frac{x^2-2}{2} = -\frac{1}{2}$: fractional

$$\begin{array}{c} y \geq x^2/2 - 1 \land y \leq x/2 \\ \uparrow \\ \frac{x^2-2}{2} \\ \uparrow \\ \texttt{true} \land \frac{x^2-2}{2} \leq \frac{x}{2} \\ \uparrow \\ -1 \\ \uparrow \\ \texttt{true} \end{array}$$

Virtual Substitution and B&B

- If no solution exists: return `unsat`
- Otherwise follow path upwards

- $x = -1$: integral
- $y = \frac{x^2-2}{2} = -\frac{1}{2}$: fractional
- Generate $(y \leq -1 \vee y \geq 0)$

$$y \geq x^2/2 - 1 \wedge y \leq x/2$$

$$\uparrow$$

$$\frac{x^2-2}{2}$$

$$\uparrow$$

$$\texttt{true} \wedge \frac{x^2-2}{2} \leq \frac{x}{2}$$

$$\uparrow$$

$$-1$$

$$\uparrow$$

$$\texttt{true}$$

Cylindrical Algebraic Decomposition [Collins 1975] [Loup+ 2012]

- Complete, but doubly exponential runtime
- two-phase approach: Projection and Lifting

Cylindrical Algebraic Decomposition [Collins 1975] [Loup+ 2012]

- Complete, but doubly exponential runtime
- two-phase approach: Projection and Lifting

Projection

Given $P_k \subset \mathbb{Q}[x_1, ..., x_k]$ construct $P_{k-1} \subset \mathbb{Q}[x_1, ..., x_{k-1}]$ such that:

$$\xi_1, ..., \xi_k \text{ is a root of } P_k \Rightarrow \xi_1, ..., \xi_{k-1} \text{ is a root of } P_{k-1}$$

Cylindrical Algebraic Decomposition [Collins 1975] [Loup+ 2012]

- Complete, but doubly exponential runtime
- two-phase approach: Projection and Lifting

Projection

Given $P_k \subset \mathbb{Q}[x_1, ..., x_k]$ construct $P_{k-1} \subset \mathbb{Q}[x_1, ..., x_{k-1}]$ such that:

$$\xi_1, ..., \xi_k \text{ is a root of } P_k \Rightarrow \xi_1, ..., \xi_{k-1} \text{ is a root of } P_{k-1}$$

Lifting

Given P_k and $\xi_1, ..., \xi_{k-1}$, we can obtain all roots of P_k by

- substituting all ξ_i and
- calculating univariate roots of $P_k[x_i/\xi_i]$.

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\}$

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$$\begin{array}{c} P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\} \\ \downarrow \\ P_1 : \{x^2 - x - 2, x^2 - 2, x\} \end{array}$$

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$$
\begin{array}{c}
P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\} \\
\downarrow \\
P_1 : \{x^2 - x - 2, x^2 - 2, x\}
\end{array}
$$

$$roots(\{x^2 - x - 2, x^2 - 2, x\})$$

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$$P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\}$$

$$\downarrow$$

$$P_1 : \{x^2 - x - 2, x^2 - 2, x\}$$

$$roots(\{x^2 - x - 2, x^2 - 2, x\})$$

$$-\sqrt{2} \qquad -1 \qquad 0 \qquad \sqrt{2} \qquad 2$$

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$$P_2 : \{y - \tfrac{x^2}{2} - 1, y - \tfrac{x}{2}\}$$

$$\downarrow$$

$$P_1 : \{x^2 - x - 2, x^2 - 2, x\}$$

$$roots(\{x^2 - x - 2, x^2 - 2, x\})$$

$-\sqrt{2}$ $\quad$ -1 $\quad$ 0 $\quad$ $\sqrt{2}$ $\quad$ 2

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$

$$P_2 : \{y - \tfrac{x^2}{2} - 1, y - \tfrac{x}{2}\}$$

$$\downarrow$$

$$P_1 : \{x^2 - x - 2, x^2 - 2, x\}$$

$$roots(\{x^2 - x - 2, x^2 - 2, x\})$$

$-\sqrt{2}$ $\quad$ -1 $\quad$ 0 $\quad$ $\sqrt{2}$ $\quad$ 2

Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \land y \leq x/2$

$P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\} \xrightarrow{y - \frac{3}{2}, y + \frac{1}{2}}$

$\downarrow$

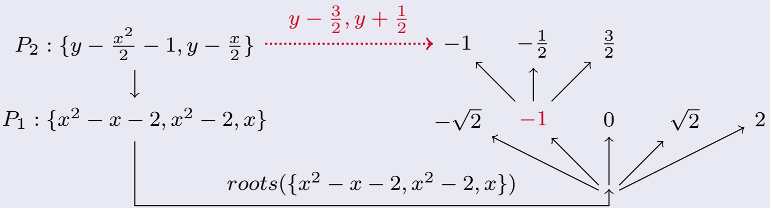
$P_1 : \{x^2 - x - 2, x^2 - 2, x\}$

$roots(\{x^2 - x - 2, x^2 - 2, x\})$

$-\sqrt{2} \quad -1 \quad 0 \quad \sqrt{2} \quad 2$

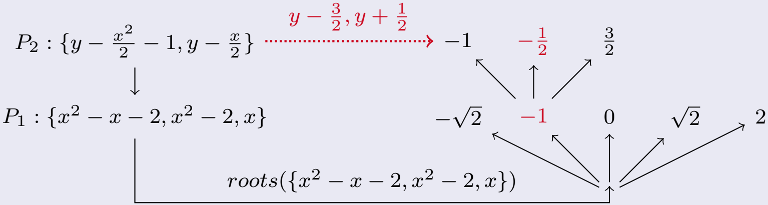
Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \land y \leq x/2$



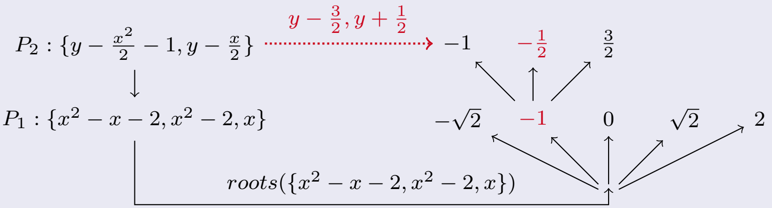
Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Cylindrical Algebraic Decomposition

Example: $y \geq x^2/2 - 1 \wedge y \leq x/2$



Satisfying solution: $x = -1, y = -\frac{1}{2}$

Example: $y \geq x^2/2 - 1 \land y \leq x/2$

$P_2 : \{y - \frac{x^2}{2} - 1, y - \frac{x}{2}\} \xrightarrow{y - \frac{3}{2}, y + \frac{1}{2}} -1 \quad -\frac{1}{2} \quad \frac{3}{2}$

$\downarrow$

$P_1 : \{x^2 - x - 2, x^2 - 2, x\}$ $\qquad -\sqrt{2} \quad -1 \quad 0 \quad \sqrt{2} \quad 2$

$roots(\{x^2 - x - 2, x^2 - 2, x\})$

Satisfying solution: $x = -1, y = -\frac{1}{2}$

Note: intermediate samples were skipped

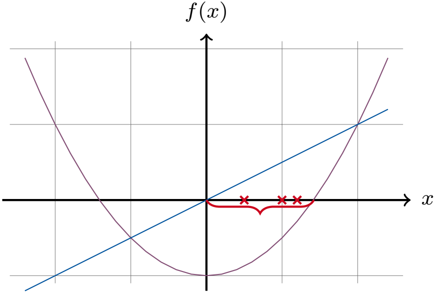
Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

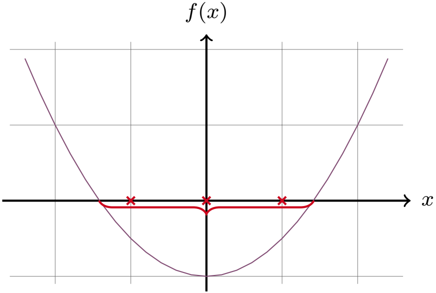
- Try to select integers



Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

- Try to select integers
- Select integers from the middle of interval



Cylindrical Algebraic Decomposition and B&B

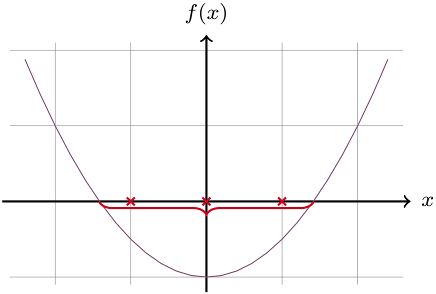
Some heuristics (that worked for us)

- Try to select integers
- Select integers from the middle of interval
- Continue lifting to avoid splitting on `unsat` samples

Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

- Try to select integers
- Select integers from the middle of interval
- Continue lifting to avoid splitting on `unsat` samples
- Do not lift multiple integers from a single interval



Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

- Try to select integers
- Select integers from the middle of interval
- Continue lifting to avoid splitting on `unsat` samples
- Do not lift multiple integers from a single interval
- Be careful with more involved backtracking schemes

Cylindrical Algebraic Decomposition and B&B

Some heuristics (that worked for us)

- Try to select integers
- Select integers from the middle of interval
- Continue lifting to avoid splitting on `unsat` samples
- Do not lift multiple integers from a single interval
- Be careful with more involved backtracking schemes
- Different lemmas if multiple assignments are rational:
  - Smallest variable
  - Largest variable
  - Activity-based

$\rightarrow$ does not matter

hybr2d Theory of Hybrid Systems Informatik 2

What about existing tools?

- Linearization [Borralleras+ 2009]

  Approximation and incremental refinement by an LIA formula

What about existing tools?

- Linearization [Borralleras+ 2009]
  Approximation and incremental refinement by an LIA formula
- Interval Constraint Propagation (`iSAT3`[Scheibler+ 2013], `raSAT`[Van Kanh+ 2014])
  Use bounds on variables to shrink solution space

What about existing tools?

- Linearization [Borralleras+ 2009]

  Approximation and incremental refinement by an LIA formula

- Interval Constraint Propagation (`iSAT3`[Scheibler+ 2013], `raSAT`[Van Kanh+ 2014])

  Use bounds on variables to shrink solution space

- Bit-blasting (`AProVE`[Giesl+ 2014], `CVC4`[Barrett+ 2011], `Z3`[de Moura 2012])

  Encode integers as vectors of boolean variables

hybr2d Theory of Hybrid Systems Informatik 2

Our approach

Strategies

$$\mathtt{RAT}_{\mathbb{Z}}: \quad \mathtt{M}_{\mathsf{SAT}} \to \mathtt{M}_{\mathsf{LRA}} \to \mathtt{M}_{\mathsf{VS}_{\mathbb{Z}}} \to \mathtt{M}_{\mathsf{CAD}_{\mathbb{Z}}}$$

$$\mathtt{RAT}_{\text{blast}}: \quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}}$$

$$\mathtt{RAT}_{\text{blast}.\mathbb{Z}}: \quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}} \to \mathtt{RAT}_{\mathbb{Z}}$$

Our approach

Strategies

$$\begin{aligned}
\mathtt{RAT}_{\mathbb{Z}} &: \quad \mathtt{M}_{\mathsf{SAT}} \to \mathtt{M}_{\mathsf{LRA}} \to \mathtt{M}_{\mathsf{VS}_{\mathbb{Z}}} \to \mathtt{M}_{\mathsf{CAD}_{\mathbb{Z}}} \\
\mathtt{RAT}_{\mathrm{blast}} &: \quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}} \\
\mathtt{RAT}_{\mathrm{blast}.\mathbb{Z}} &: \quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}} \to \mathtt{RAT}_{\mathbb{Z}}
\end{aligned}$$

- $\mathtt{RAT}_{\mathbb{Z}}$: uses branch-and-bound approach as described
- $\mathtt{M}_{\mathsf{IntBlast}}$: used bit-blasting approach
- $\mathtt{M}_{\mathsf{IncWidth}}$: creates and widens artificial bounds on all variables
- $\mathtt{RAT}_{\mathsf{blast}.\mathbb{Z}}$: uses $\mathtt{M}_{\mathsf{IncWidth}}$ up to 4 bits and uses $\mathtt{RAT}_{\mathbb{Z}}$ afterwards

Our approach

Strategies

$$
\begin{aligned}
\mathtt{RAT}_{\mathbb{Z}}: &\quad \mathtt{M}_{\mathsf{SAT}} \to \mathtt{M}_{\mathsf{LRA}} \to \mathtt{M}_{\mathsf{VS}_{\mathbb{Z}}} \to \mathtt{M}_{\mathsf{CAD}_{\mathbb{Z}}} \\
\mathtt{RAT}_{\mathrm{blast}}: &\quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}} \\
\mathtt{RAT}_{\mathrm{blast}.\mathbb{Z}}: &\quad \mathtt{M}_{\mathsf{IncWidth}} \to \mathtt{M}_{\mathsf{IntBlast}} \to \mathtt{RAT}_{\mathbb{Z}}
\end{aligned}
$$

- $\mathtt{RAT}_{\mathbb{Z}}$: uses branch-and-bound approach as described
- $\mathtt{M}_{\mathsf{IntBlast}}$: used bit-blasting approach
- $\mathtt{M}_{\mathsf{IncWidth}}$: creates and widens artificial bounds on all variables
- $\mathtt{RAT}_{\mathsf{blast}.\mathbb{Z}}$: uses $\mathtt{M}_{\mathsf{IncWidth}}$ up to 4 bits and uses $\mathtt{RAT}_{\mathbb{Z}}$ afterwards

- Rationale: find small solutions fast, use $\mathtt{RAT}_{\mathbb{Z}}$ for large solutions

Theory of Hybrid Systems Informatik 2

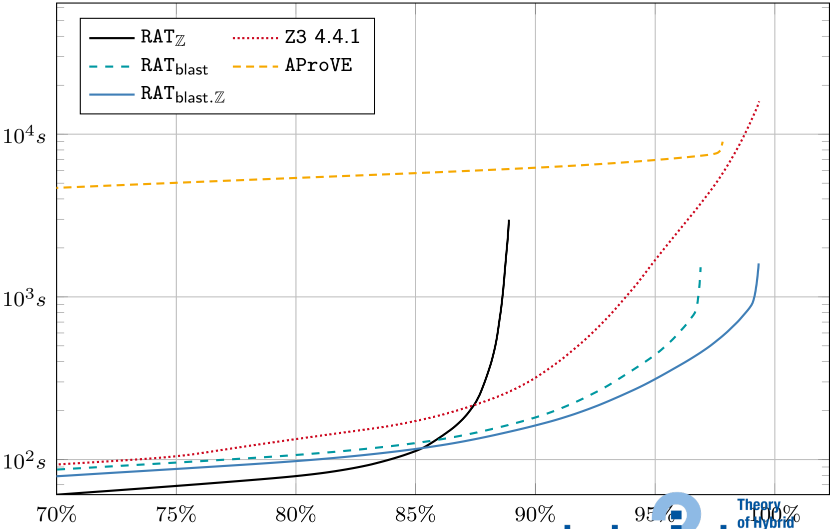
Experimental results

Benchmark→ Solver↓		APROVE (8129) #	time	CALYPTO (138) #	time	LEIPZIG (167) #	time	CALYPTO$_\infty$ (138) #	time	all (8572) #	time
RAT$_\mathbb{Z}$	sat	7283	2294.8	67	71.2	9	260.4	133	298.9	7492	2925.3
	unsat	73	14.3	52	40.7	0	0.0	**3**	**< 0.1**	128	55.1
RAT$_\text{blast}$	sat	8025	866.3	21	35.6	156	603.3	87	16.0	8289	1521.2
	unsat	12	0.4	5	0.1	0	0.0	0	0.0	17	0.5
RAT$_{\text{blast}.\mathbb{Z}}$	sat	**8025**	**780.7**	**79**	**122.3**	156	511.5	**134**	**21.8**	**8394**	**1436.3**
	unsat	71	42.6	46	127.5	0	0.0	3	0.1	120	170.2
Z3	sat	7992	14695.5	78	19.1	158	427.6	126	57.3	8354	15199.5
	unsat	**102**	**595.9**	**57**	**117.6**	0	0.0	3	2.3	**162**	**715.8**
AProVE	sat	8025	7052.2	74	559.1	**159**	**696.5**	127	685.2	8385	8993.0
	unsat	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0

- Comparing against `Z3` and `AProVE`:
  - Good on `sat`, lagging behind for `unsat` (preprocessing?)
  - Very fast (we start with a smaller bit-width)
- Comparing our strategies:
  - RAT$_\mathbb{Z}$ complements RAT$_\text{blast}$ nicely on many examples
  - RAT$_\mathbb{Z}$ finds large solution fast in many cases

hybr2d Theory of Hybrid Systems Informatik 2

Experimental results



Conclusion

- Branch-and-Bound is applied naturally to nonlinear arithmetic
- Comparably easy implementation
- Many possibilities for heuristics

Conclusion

- Branch-and-Bound is applied naturally to nonlinear arithmetic
- Comparably easy implementation
- Many possibilities for heuristics
- Nicely complements bit-blasting approach
- Benefits from improvements on NRA