



Techniques for NRA

How to solve Nonlinear Real Arithmetic

... and a lot of references



Stanford University



Contains mostly other peoples work!

Contains joint work with: Erika Ábrahám, Florian Corzilius, James Davenport, Matthew England, Rebecca Haehn, Jasper Nalbach



Satisfiability modulo theories

Let's skip that...



SMT for Nonlinear Real Arithmetic

Here: Theory of the Reals



SMT for Nonlinear Real Arithmetic

Here: Theory of the Reals

Nonlinear Real Arithmetic:

- ▶ real variables $v := x_i \in \mathbb{R}$
- ▶ constants $c := q \in \mathbb{Z}$
- ▶ terms $t := v \mid c \mid t + t \mid t \cdot t$
- ▶ atoms $a := t \sim 0, \sim \in \{<, >, \leq, \geq, =, \neq\}$



SMT for Nonlinear Real Arithmetic

Here: Theory of the Reals

Nonlinear Real Arithmetic:

- ▶ real variables $v := x_i \in \mathbb{R}$
- ▶ constants $c := q \in \mathbb{Z}$
- ▶ terms $t := v \mid c \mid t + t \mid t \cdot t$
- ▶ atoms $a := t \sim 0, \sim \in \{<, >, \leq, \geq, =, \neq\}$

Intuition: polynomials over real variables compared to zero.



SMT for Nonlinear Real Arithmetic

Here: Theory of the Reals

Nonlinear Real Arithmetic:

- ▶ real variables $v := x_i \in \mathbb{R}$
- ▶ constants $c := q \in \mathbb{Z}$
- ▶ terms $t := v \mid c \mid t + t \mid t \cdot t$
- ▶ atoms $a := t \sim 0, \sim \in \{<, >, \leq, \geq, =, \neq\}$

Intuition: polynomials over real variables compared to zero.

Does cover: $t > t$, rational constants, division (encoding with auxiliary variables)

Does not cover: transcendental constants, non-polynomial functions



SMT for Nonlinear Real Arithmetic

Here: Theory of the Reals

Nonlinear Real Arithmetic:

- ▶ real variables $v := x_i \in \mathbb{R}$
- ▶ constants $c := q \in \mathbb{Z}$
- ▶ terms $t := v \mid c \mid t + t \mid t \cdot t$
- ▶ atoms $a := t \sim 0, \sim \in \{<, >, \leq, \geq, =, \neq\}$

Intuition: polynomials over real variables compared to zero.

Does cover: $t > t$, rational constants, division (encoding with auxiliary variables)

Does not cover: transcendental constants, non-polynomial functions

Linear arithmetic: essentially a solved problem.

Use Simplex (or sometimes Fourier-Motzkin)



Theory of the Reals in a nutshell

- ▶ **complete** (we have decision procedures that are sound and complete)
- ▶ **admits quantifier elimination** (quantifiers are conceptually easy)



Theory of the Reals in a nutshell

- ▶ **complete** (we have decision procedures that are sound and complete)
- ▶ **admits quantifier elimination** (quantifiers are conceptually easy)

Some methods:

- ▶ [Tarski 1951] Tarski: first complete method, **non-elementary complexity**
- ▶ [Buchberger 1965] Gröbner bases: **limited applicability**, standard tool in CA
- ▶ [Collins 1974] CAD: **complete**, doubly exponential complexity
- ▶ [Weispfenning 1988] VS: **up to bounded degree**, singly exponential complexity
- ▶ [Gao et al. 2013] ICP: **heuristic interval reasoning**, incomplete
- ▶ [Fontaine et al. 2017] Subtropical satisfiability: incomplete **reduction to LRA**
- ▶ [Irfan 2018] Linearization: incomplete, **axiom instantiation**
- ▶ [Ábrahám et al. 2020] CDCAC: **conflict-driven CAD**
- ▶ and some more...



Overview

- 1 SMT for NRA
- 2 Linearization
- 3 Interval Constraint Propagation
- 4 Subtropical Satisfiability
- 5 Gröbner Bases
- 6 Virtual Substitution
- 7 Cylindrical Algebraic Decomposition
- 8 Conflict-Driven Cylindrical Algebraic Coverings
- 9 Related topics



Linearization by example

[Irfan 2018] [Cimatti et al. 2018]

- ▶ Linearize atoms
- ▶ Solve
- ▶ Identify conflicts
- ▶ Instantiate axioms
- ▶ Add as lemmas
- ▶ Repeat



Linearization by example

[Irfan 2018] [Cimatti et al. 2018]

- ▶ Linearize atoms
- ▶ Solve
- ▶ Identify conflicts
- ▶ Instantiate axioms
- ▶ Add as lemmas
- ▶ Repeat

$$x \cdot y \leq 0 \wedge x < 0 \wedge x + y = 0$$

$$\text{linearize: } z \leq 0 \wedge x < 0 \wedge x + y = 0 \quad z := x \cdot y$$

$$\text{atoms: } z \leq 0 \wedge x < 0 \wedge x + y = 0$$

$$\text{solve: } x \mapsto -1, y \mapsto 1, z \mapsto 0$$

$$\text{conflict: } 0 \neq -1 \cdot 1$$

$$\text{axiom: } z = 0 \Rightarrow (x = 0 \vee y = 0)$$

add axiom as lemma, proceed to next theory call

$$\text{atoms: } z \leq 0 \wedge x < 0 \wedge x + y = 0 \wedge z \neq 0$$

$$\text{solve: } x \mapsto -1, y \mapsto 1, z \mapsto -1$$

SAT!



Linearization

[Irfan 2018] [Cimatti et al. 2018]

Intuition: **iteratively teach the linear solver** about the nonlinear parts, add lemmas that cut away unsatisfiable regions.



Linearization

[Irfan 2018] [Cimatti et al. 2018]

Intuition: **iteratively teach the linear solver** about the nonlinear parts, add lemmas that cut away unsatisfiable regions.

More axioms: zeroes, monotonicity, commutativity, symmetry w.r.t. signs, tangent planes, ...



Linearization

[Irfan 2018] [Cimatti et al. 2018]

Intuition: **iteratively teach the linear solver** about the nonlinear parts, add lemmas that cut away unsatisfiable regions.

More axioms: zeroes, monotonicity, commutativity, symmetry w.r.t. signs, tangent planes, ...

Problems: difficult to identify models (linear solver only finds corners), linear solver only finds rational assignments ($x^2 = 2$)



Linearization

[Irfan 2018] [Cimatti et al. 2018]

Intuition: **iteratively teach the linear solver** about the nonlinear parts, add lemmas that cut away unsatisfiable regions.

More axioms: zeroes, monotonicity, commutativity, symmetry w.r.t. signs, tangent planes, ...

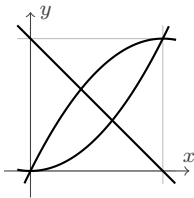
Problems: difficult to identify models (linear solver only finds corners), linear solver only finds rational assignments ($x^2 = 2$)

Extensions:

- ▶ **Repair model** (if easily possible)
- ▶ Transcendental functions
- ▶ extended operators in general



Interval Constraint Propagation by example

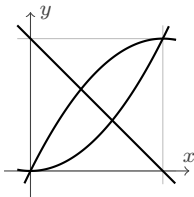


$$y > x^2 \quad \wedge \quad y < -x^2 + 2x \quad \wedge \quad y \leq 1 - x$$

$$x \times y$$



Interval Constraint Propagation by example

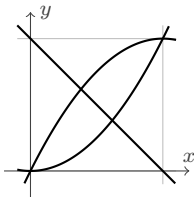


$$y > x^2 \wedge y < -x^2 + 2x \wedge y \leq 1 - x$$
$$y > x^2 \Rightarrow y \in (0, \infty)$$

$$x \times y$$
$$(-\infty, \infty) \times (0, \infty)$$



Interval Constraint Propagation by example



$$y > x^2 \quad \wedge \quad y < -x^2 + 2x \quad \wedge \quad y \leq 1 - x$$

$$y > x^2 \Rightarrow y \in (0, \infty)$$

$$x > 0.5x^2 + y \Rightarrow x \in (0, \infty)$$

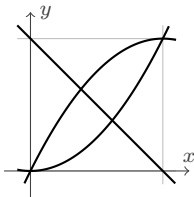
$$x \times y$$

$$(-\infty, \infty) \times (0, \infty)$$

$$(0, \infty) \times (0, \infty)$$



Interval Constraint Propagation by example



$$y > x^2 \wedge y < -x^2 + 2x \wedge y \leq 1 - x$$

$$y > x^2 \Rightarrow y \in (0, \infty)$$

$$x > 0.5x^2 + y \Rightarrow x \in (0, \infty)$$

$$x \leq -y + 1 \Rightarrow x \in (0, 1)$$

$$x \times y$$

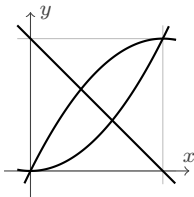
$$(-\infty, \infty) \times (0, \infty)$$

$$(0, \infty) \times (0, \infty)$$

$$(0, 1) \times (0, \infty)$$



Interval Constraint Propagation by example



$$y > x^2 \wedge y < -x^2 + 2x \wedge y \leq 1 - x$$

$$y > x^2 \Rightarrow y \in (0, \infty)$$

$$x > 0.5x^2 + y \Rightarrow x \in (0, \infty)$$

$$x \leq -y + 1 \Rightarrow x \in (0, 1)$$

$$y \leq -x + 1 \Rightarrow y \in (0, 1)$$

$$x \times y$$

$$(-\infty, \infty) \times (0, \infty)$$

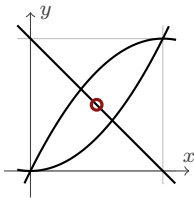
$$(0, \infty) \times (0, \infty)$$

$$(0, 1) \times (0, \infty)$$

$$(0, 1) \times (0, 1)$$



Interval Constraint Propagation by example



$$y > x^2 \wedge y < -x^2 + 2x \wedge y \leq 1 - x$$

$$y > x^2 \Rightarrow y \in (0, \infty)$$

$$x > 0.5x^2 + y \Rightarrow x \in (0, \infty)$$

$$x \leq -y + 1 \Rightarrow x \in (0, 1)$$

$$y \leq -x + 1 \Rightarrow y \in (0, 1)$$

$$\text{guess midpoint } (0.5, 0.5) \in (0, 1) \times (0, 1)$$

$$x \times y$$

$$(-\infty, \infty) \times (0, \infty)$$

$$(0, \infty) \times (0, \infty)$$

$$(0, 1) \times (0, \infty)$$

$$(0, 1) \times (0, 1)$$



Interval Constraint Propagation in a nutshell

[Benhamou et al. 2006] [Gao et al. 2013] [Scheibler et al. 2013] [Schupp 2013] [Tung et al. 2017]

Core idea:

- ▶ Maintain **interval assignment** (that represents the **current box**)
- ▶ Perform **over-approximating** contractions until
 - ▶ the current box is **empty** (UNSAT),
 - ▶ we can **guess a model** (SAT), or
 - ▶ we reach a **threshold**.
- ▶ When reaching a threshold
 - ▶ we terminate with **unknown** or
 - ▶ **split**: $x \in [0, 5] \rightsquigarrow (x < 3 \vee x \geq 3)$



Interval Constraint Propagation in a nutshell

[Benhamou et al. 2006] [Gao et al. 2013] [Scheibler et al. 2013] [Schupp 2013] [Tung et al. 2017]

Core idea:

- ▶ Maintain **interval assignment** (that represents the **current box**)
- ▶ Perform **over-approximating** contractions until
 - ▶ the current box is **empty** (UNSAT),
 - ▶ we can **guess a model** (SAT), or
 - ▶ we reach a **threshold**.
- ▶ When reaching a threshold
 - ▶ we terminate with **unknown** or
 - ▶ **split**: $x \in [0, 5] \rightsquigarrow (x < 3 \vee x \geq 3)$
- ▶ **Incomplete** solving procedure
- ▶ Used as **preprocessor** for other techniques [Loup et al. 2013]
- ▶ **Delicate tuning** of heuristics (splitting, thresholds, model guessing)



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**

For $n = 1$: $\lim_{x \rightarrow \infty} p(x) = \infty$ if $\text{lcoeff}(p) > 0$. Increase x as necessary.



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**

For $n = 1$: $\lim_{x \rightarrow \infty} p(x) = \infty$ if $\text{lcoeff}(p) > 0$. Increase x as necessary.

For $n \geq 2$: search **direction in exponent space** such that the **largest exponent in this direction** is positive. Increase x in **this direction** as necessary.



Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

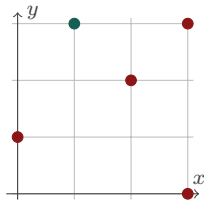
- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**

For $n = 1$: $\lim_{x \rightarrow \infty} p(x) = \infty$ if $\text{lcoeff}(p) > 0$. Increase x as necessary.

For $n \geq 2$: search **direction in exponent space** such that the **largest exponent in this direction** is positive. Increase x in this direction as necessary.

$$p = -y + 2xy^3 - 3x^2y^2 - x^3 - 4x^3y^3$$





Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

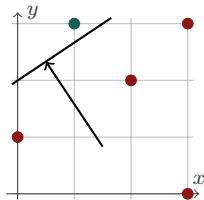
Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**

For $n = 1$: $\lim_{x \rightarrow \infty} p(x) = \infty$ if $\text{lcoeff}(p) > 0$. Increase x as necessary.

For $n \geq 2$: search **direction in exponent space** such that the **largest exponent in this direction** is positive. Increase x in this direction as necessary.

$$p = -y + 2xy^3 - 3x^2y^2 - x^3 - 4x^3y^3$$

Find hyperplane that separates a **positive** node





Subtropical satisfiability

[Fontaine et al. 2017] [Fontaine et al. 2018]

Core idea: reduce $p = 0$ to a **linear** problem in the **exponents of p**

- ▶ Assume $p(1, \dots, 1) < 0$ (otherwise consider $-p$)
- ▶ Find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$
- ▶ Solve $p(y) = 0$ with y on the line $(1, \dots, 1) - x$
- ▶ Incomplete (no such y exists, though $p = 0$ has a solution)

Core problem: **How to find $x \in \mathbb{R}_+^n$ such that $p(x) > 0$?**

For $n = 1$: $\lim_{x \rightarrow \infty} p(x) = \infty$ if $\text{lcoeff}(p) > 0$. Increase x as necessary.

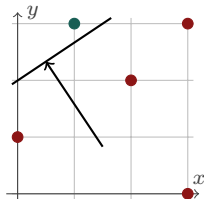
For $n \geq 2$: search **direction in exponent space** such that the **largest exponent in this direction** is positive. Increase x in this direction as necessary.

$$p = -y + 2xy^3 - 3x^2y^2 - x^3 - 4x^3y^3$$

Find hyperplane that separates a **positive** node

Encoding in QF_LRA

Growing degree only impacts coefficient size





Gröbner basis

[Buchberger 1965] [Junges 2012]

- ▶ **Canonical generators** for a polynomial ideal
- ▶ For us: **Normal form** for sets of polynomials
- ▶ Maintains set of **common complex roots**
- ▶ The workhorse of **computer algebra** for **polynomial equalities**
- ▶ **Mature implementations** (every CAS)
- ▶ Doubly exponential in worst case, but usually much faster.



Gröbner basis

[Buchberger 1965] [Junges 2012]

- ▶ **Canonical generators** for a polynomial ideal
- ▶ For us: **Normal form** for sets of polynomials
- ▶ Maintains set of **common complex roots**
- ▶ The workhorse of **computer algebra** for **polynomial equalities**
- ▶ **Mature implementations** (every CAS)
- ▶ Doubly exponential in worst case, but usually much faster.

Relevant for SMT: $\exists x \in \mathbb{C}^n. p(x) = 0$



Gröbner basis

[Buchberger 1965] [Junges 2012]

- ▶ **Canonical generators** for a polynomial ideal
- ▶ For us: **Normal form** for sets of polynomials
- ▶ Maintains set of **common complex roots**
- ▶ The workhorse of **computer algebra** for **polynomial equalities**
- ▶ **Mature implementations** (every CAS)
- ▶ Doubly exponential in worst case, but usually much faster.

Relevant for SMT: $\exists x \in \mathbb{C}^n. p(x) = 0$

But: What about **inequalities**? How to go **from \mathbb{C} to \mathbb{R}** ?
see [Junges 2012] for some approaches.



Virtual Substitution

[Weispfenning 1988] [Weispfenning 1997] [Kořta et al. 2015] [Kořta 2016] [Nalbach 2017]

Core idea:

- ▶ Use **solution formula** to solve polynomial equation for x
- ▶ **Substitute value** for x into remaining equations
- ▶ Repeat for remaining variables



Virtual Substitution

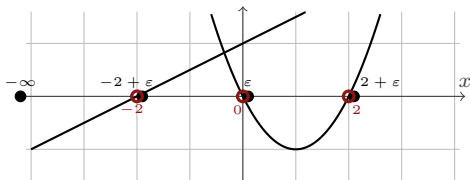
[Weispfenning 1988] [Weispfenning 1997] [Košta et al. 2015] [Košta 2016] [Nalbach 2017]

Core idea:

- ▶ Use **solution formula** to solve polynomial equation for x
- ▶ **Substitute value** for x into remaining equations
- ▶ Repeat for remaining variables

What about **inequalities**?

- ▶ Construct test candidates for all **sign-invariant** regions in x
- ▶ Always try the **roots** and the **smallest values of the intermediate intervals**



- ▶ Introduces special terms $t + \epsilon$ and $-\infty$



Virtual Substitution

Algorithmic core: a collection of substitution rules

Example: Substitute $e + \varepsilon$ for x into $a \cdot x^2 + b \cdot x + c > 0$:

$$\begin{array}{l}
 \vee \quad ((ax^2 + bx + c > 0)[e//x]) \\
 \vee \quad ((ax^2 + bx + c = 0)[e//x] \wedge (2ax + b > 0)[e//x]) \\
 \vee \quad ((ax^2 + bx + c = 0)[e//x] \wedge (2ax + b = 0)[e//x] \wedge (2a > 0)[e//x])
 \end{array}$$



Virtual Substitution

Algorithmic core: a collection of substitution rules

Example: Substitute $e + \varepsilon$ for x into $a \cdot x^2 + b \cdot x + c > 0$:

$$\begin{aligned} & ((ax^2 + bx + c > 0)[e//x]) \\ \vee & ((ax^2 + bx + c = 0)[e//x] \quad \wedge \quad (2ax + b > 0)[e//x]) \\ \vee & ((ax^2 + bx + c = 0)[e//x] \quad \wedge \quad (2ax + b = 0)[e//x] \quad \wedge \quad (2a > 0)[e//x]) \end{aligned}$$

Not always applicable:

- ▶ Solution formulas only exist **up to degree four**
- ▶ The above rule may introduce a **degree growth**
- ▶ **Efficient** if applicable
- ▶ [Košta et al. 2015] uses FO formulas, allows **arbitrary but fixed degrees**
(needs precomputed substitution rules obtained by quantifier elimination)



Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**



Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**

Construct a **sign-invariant decomposition** of \mathbb{R}^n :

$$\text{cell } C \subset \mathbb{R}^n : \forall a, b \in C : \varphi(a) = \varphi(b)$$

Abstraction: \mathbb{R}^n to **finite** set of cells, consider a single $a \in C$ per cell.



Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

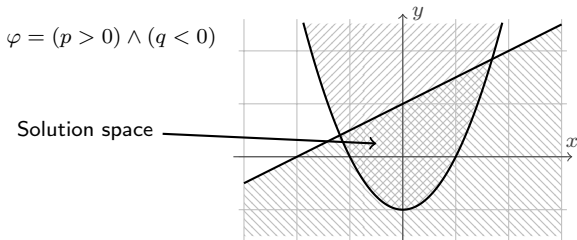
$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**

Construct a **sign-invariant decomposition** of \mathbb{R}^n :

$$\text{cell } C \subset \mathbb{R}^n : \forall a, b \in C : \varphi(a) = \varphi(b)$$

Abstraction: \mathbb{R}^n to **finite** set of cells, consider a single $a \in C$ per cell.





Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

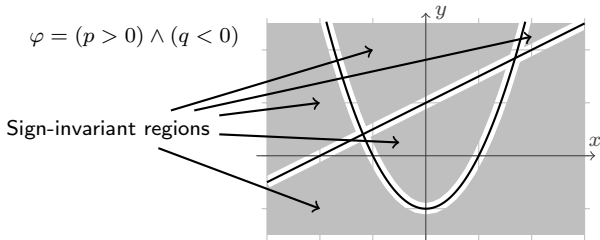
$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**

Construct a **sign-invariant decomposition** of \mathbb{R}^n :

$$\text{cell } C \subset \mathbb{R}^n : \forall a, b \in C : \varphi(a) = \varphi(b)$$

Abstraction: \mathbb{R}^n to **finite** set of cells, consider a single $a \in C$ per cell.





Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**

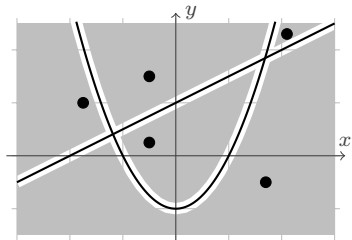
Construct a **sign-invariant decomposition of \mathbb{R}^n** :

$$\text{cell } C \subset \mathbb{R}^n : \forall a, b \in C : \varphi(a) = \varphi(b)$$

Abstraction: \mathbb{R}^n to **finite** set of cells, consider a single $a \in C$ per cell.

$$\varphi = (p > 0) \wedge (q < 0)$$

Sample points





Cylindrical Algebraic Decomposition

The core idea: **sign-invariance** (or rather **truth-table equivalence**)

$$\text{sgn}(p(a)) = \text{sgn}(p(b)) \quad \forall p \in \varphi \quad \Rightarrow \quad \varphi(a) = \varphi(b)$$

For our purpose, a and b are **equivalent!**

Construct a **sign-invariant decomposition** of \mathbb{R}^n :

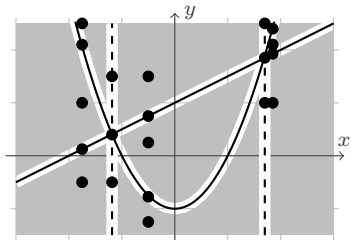
$$\text{cell } C \subset \mathbb{R}^n : \forall a, b \in C : \varphi(a) = \varphi(b)$$

Abstraction: \mathbb{R}^n to **finite** set of cells, consider a single $a \in C$ per cell.

$$\varphi = (p > 0) \wedge (q < 0)$$

Actual sample points

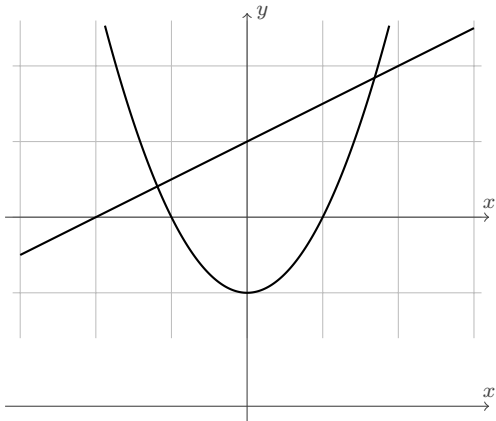
Arranged in **cylinders**





Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

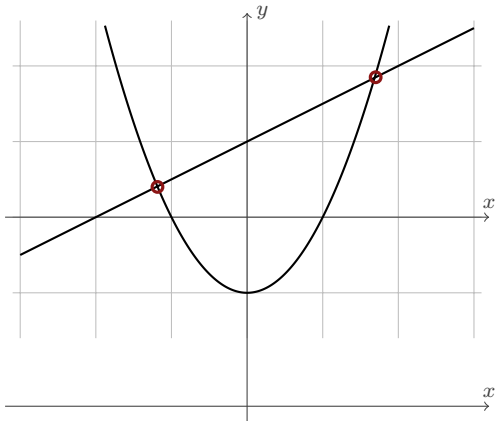


Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

Intuition

Critical points



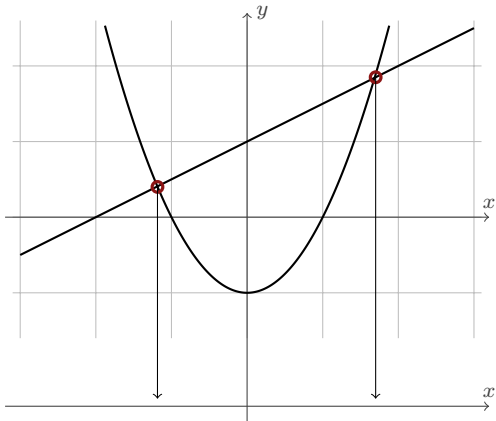
Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

Intuition

Critical points

Project sample





Cylindrical Algebraic Decomposition in \mathbb{R}^2

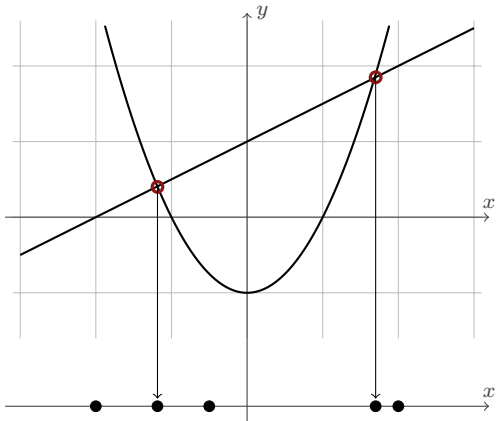
Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

Intuition

Critical points

Project sample

Solve 1-dim



Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

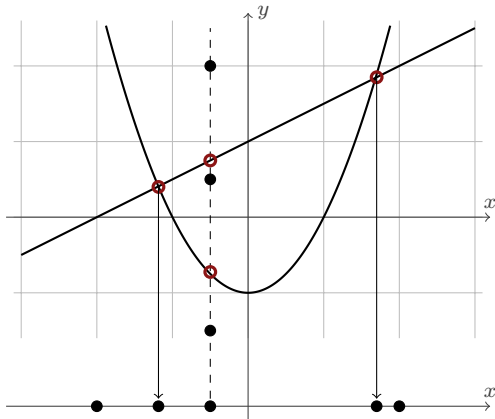
Intuition

Critical points

Project sample

Solve 1-dim

Lift to 2-dim





Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

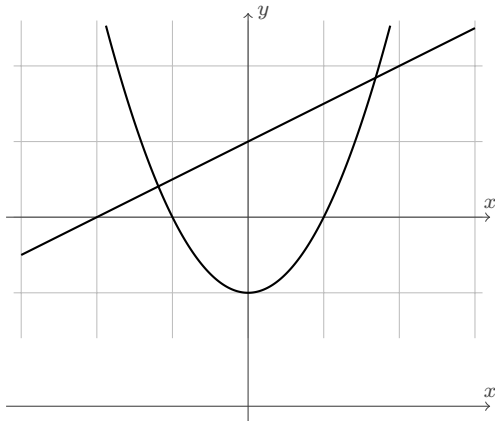
Intuition

Critical points

Project sample

Solve 1-dim

Lift to 2-dim



Implementation

Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

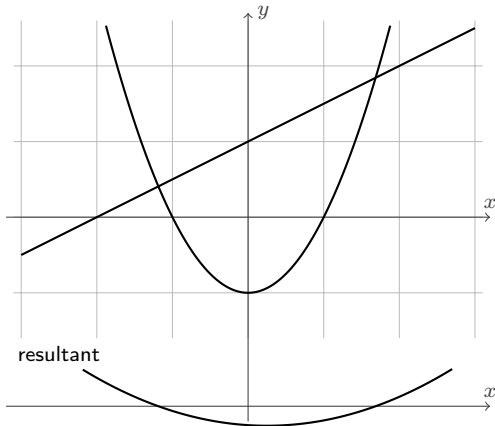
Intuition

Critical points

Project sample

Solve 1-dim

Lift to 2-dim

**Implementation**

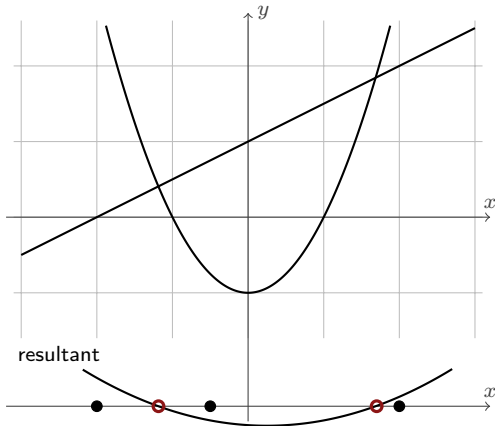
Project polynomials

Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

Intuition

Critical points
Project sample
Solve 1-dim
Lift to 2-dim

**Implementation**

Project polynomials
Solve 1-dim

Cylindrical Algebraic Decomposition in \mathbb{R}^2

Proceed **dimension-wise**: project to lower-dimensional problem, lift results.

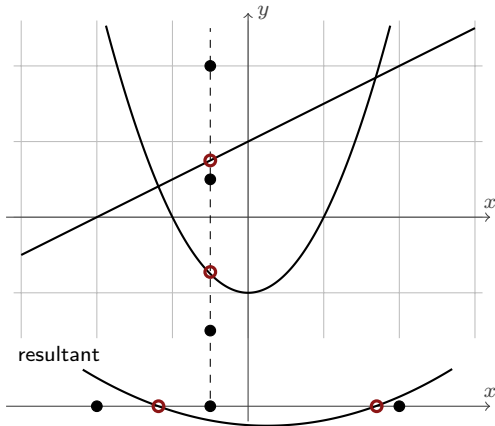
Intuition

Critical points

Project sample

Solve 1-dim

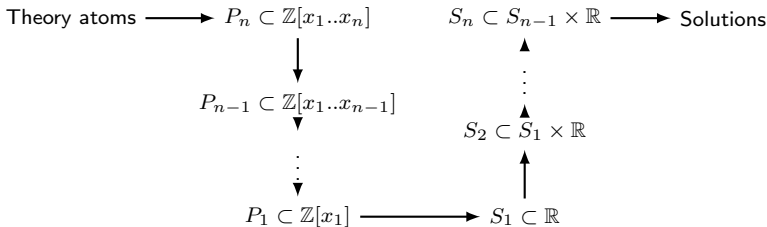
Lift to 2-dim

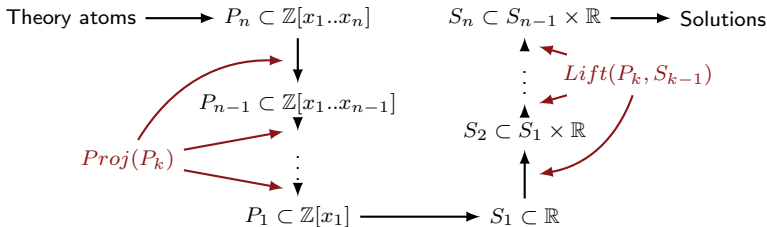
**Implementation**

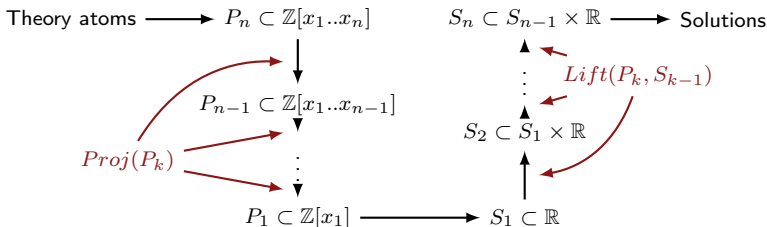
Project polynomials

Solve 1-dim

Lift to 2-dim

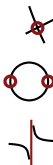
Cylindrical Algebraic Decomposition in \mathbb{R}^n 

Cylindrical Algebraic Decomposition in \mathbb{R}^n 

Cylindrical Algebraic Decomposition in \mathbb{R}^n 

Projection:

- Intersections (resultants)
- Flipping points (discriminants)
- Singularities (coefficients)





Final notes on CAD

- ▶ Asymptotic complexity: $(n \cdot m)^{2^r}$ (r variables, m polynomials of degree n)
- ▶ Oftentimes way faster, but worst-case occurs in practice!
- ▶ Best complete method that is known and implemented. [Hong 1991]
- ▶ Active research:
 - ▶ Projection [McCallum 1984] [McCallum 1988] [Hong 1990] [Lazard 1994] [Brown 2001] [McCallum 2001] [McCallum et al. 2016] [McCallum et al. 2019]; [Strzeboński 2000] [Seidl et al. 2003] [Jovanović et al. 2012] [Brown 2013] [Strzeboński 2014] [Brown et al. 2015]
 - ▶ Lifting [Collins 1974] [Lazard 1994] [McCallum et al. 2016] [McCallum et al. 2019]
 - ▶ Equational constraints [Collins 1998] [McCallum 1999] [McCallum 2001] [England et al. 2015] [Haehn et al. 2018] [Nair et al. 2019]
 - ▶ Variable ordering [England et al. 2014] [Huang et al. 2014] [Nalbach et al. 2019] [Florescu et al. 2019]
 - ▶ Adaptions [Jovanović et al. 2012] [Brown 2013] [Brown 2015] [Ábrahám et al. 2020]
- ▶ Implementation needs groundwork: polynomial computation (resultants, multivariate gcd, optionally multivariate factorization), real algebraic numbers (representation, multivariate root isolation)



Conflict-Driven Cylindrical Algebraic Coverings

[Ábrahám et al. 2020]

Core idea: use **CAD techniques** in a **conflict-driven** way.

My intuition: MCSAT turned into a theory solver.



Conflict-Driven Cylindrical Algebraic Coverings

[Ábrahám et al. 2020]

Core idea: use **CAD techniques** in a **conflict-driven** way.

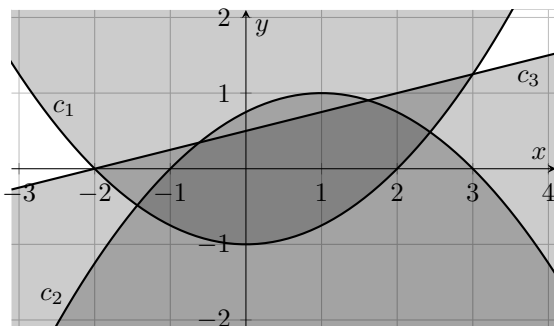
My intuition: MCSAT turned into a theory solver.

- ▶ Fix a **variable ordering**
- ▶ For the k th variable
 - ▶ Use constraints to **exclude unsatisfiable intervals**
 - ▶ **Guess** a value for the k th variable
 - ▶ Recurse to $k + 1$ st variable and obtain
 - ▶ a **full variable assignment** (\rightarrow return SAT)
 - ▶ or a **covering for the $k + 1$ st variable**
 - ▶ Use **CAD machinery** to infer an interval for the k th variable
- ▶ Until the collected intervals form a **covering** for the k th variable



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$

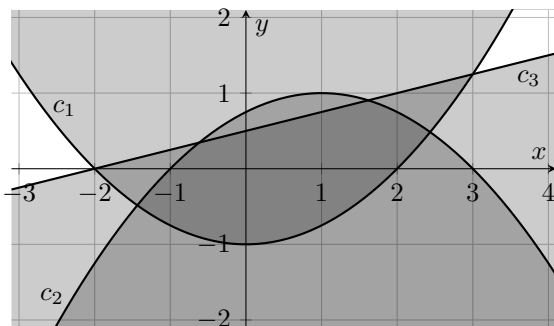




An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$

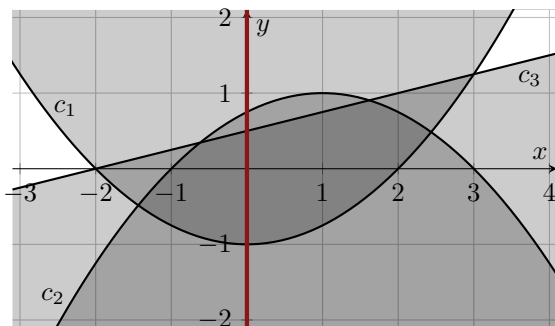
No constraint for x





An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$

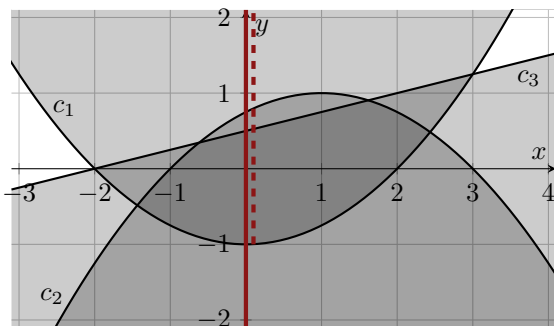


No constraint for x
Guess $x \mapsto 0$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

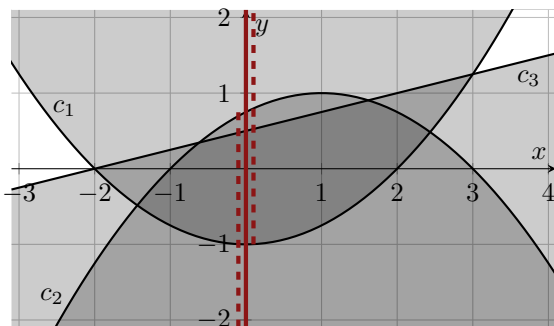
Guess $x \mapsto 0$

$c_1 \rightarrow y \notin (-1, \infty)$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

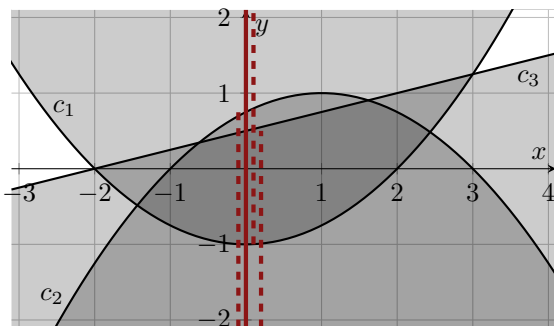
$$c_1 \rightarrow y \notin (-1, \infty)$$

$$c_2 \rightarrow y \notin (-\infty, 0.75)$$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

$$c_1 \rightarrow y \notin (-1, \infty)$$

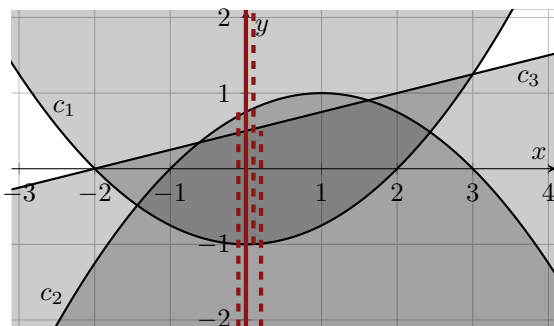
$$c_2 \rightarrow y \notin (-\infty, 0.75)$$

$$c_3 \rightarrow y \notin (-\infty, 0.5)$$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

$$c_1 \rightarrow y \notin (-1, \infty)$$

$$c_2 \rightarrow y \notin (-\infty, 0.75)$$

$$c_3 \rightarrow y \notin (-\infty, 0.5)$$

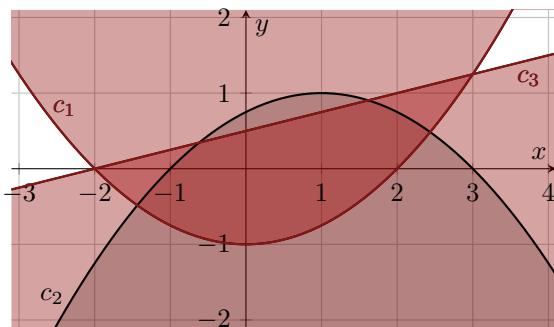
Construct covering

$$(-\infty, 0.5), (-1, \infty)$$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

$$c_1 \rightarrow y \notin (-1, \infty)$$

$$c_2 \rightarrow y \notin (-\infty, 0.75)$$

$$c_3 \rightarrow y \notin (-\infty, 0.5)$$

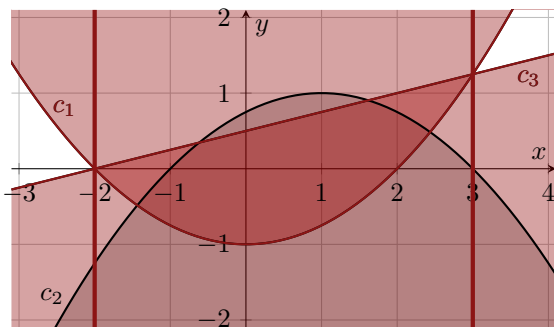
Construct covering

$$(-\infty, 0.5), (-1, \infty)$$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

$$c_1 \rightarrow y \notin (-1, \infty)$$

$$c_2 \rightarrow y \notin (-\infty, 0.75)$$

$$c_3 \rightarrow y \notin (-\infty, 0.5)$$

Construct covering

$$(-\infty, 0.5), (-1, \infty)$$

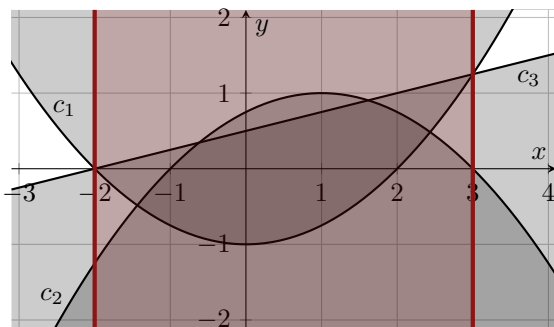
Construct interval for x

$$x \notin (-2, 3)$$



An example

$$c_1 : 4 \cdot y < x^2 - 4 \quad c_2 : 4 \cdot y > 4 - (x - 1)^2 \quad c_3 : 4 \cdot y > x + 2$$



No constraint for x

Guess $x \mapsto 0$

$$c_1 \rightarrow y \notin (-1, \infty)$$

$$c_2 \rightarrow y \notin (-\infty, 0.75)$$

$$c_3 \rightarrow y \notin (-\infty, 0.5)$$

Construct covering

$$(-\infty, 0.5), (-1, \infty)$$

Construct interval for x

$$x \notin (-2, 3)$$

New guess for x



The main algorithm

```
function get_unsat_cover( $(s_1, \dots, s_{i-1})$ )
```

```
   $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
  while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
     $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
    if  $i = n$  then return (SAT,  $(s_1, \dots, s_{i-1}, s_i)$ )
```

```
     $(f, O) := \text{get\_unsat\_cover}((s_1, \dots, s_{i-1}, s_i))$ 
```

```
    if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
    else if  $f = \text{UNSAT}$  then
```

```
       $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i), O)$ 
```

```
       $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}), s_i, R)$ 
```

```
       $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
    end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```




The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i), O)$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}), s_i, R)$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point



The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i), O)$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}), s_i, R)$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point

Select sample from $\mathbb{R} \setminus \mathbb{I}$



The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i), O)$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}), s_i, R)$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point

Select sample from $\mathbb{R} \setminus \mathbb{I}$

Recurse to next variable



The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i))$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}, s_i), R)$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point

Select sample from $\mathbb{R} \setminus \mathbb{I}$

Recurse to next variable

CAD-style projection:
Roots of polynomials restrict where covering is still applicable



The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i))$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}, s_i), R)$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point

Select sample from $\mathbb{R} \setminus \mathbb{I}$

Recurse to next variable

CAD-style projection:
Roots of polynomials restrict where covering is still applicable

Extract interval from polynomials



The main algorithm

```
function get_unsat_cover(( $s_1, \dots, s_{i-1}$ ))
```

```
 $\mathbb{I} := \text{get\_unsat\_intervals}(s)$ 
```

```
while  $\bigcup_{I \in \mathbb{I}} I \neq \mathbb{R}$  do
```

```
   $s_i := \text{sample\_outside}(\mathbb{I})$ 
```

```
  if  $i = n$  then return (SAT, ( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  ( $f, O$ ) := get_unsat_cover(( $s_1, \dots, s_{i-1}, s_i$ ))
```

```
  if  $f = \text{SAT}$  then return (SAT,  $O$ )
```

```
  else if  $f = \text{UNSAT}$  then
```

```
     $R := \text{construct\_characterization}((s_1, \dots, s_{i-1}, s_i))$ 
```

```
     $J := \text{interval\_from\_characterization}((s_1, \dots, s_{i-1}, s_i))$ 
```

```
     $\mathbb{I} := \mathbb{I} \cup \{J\}$ 
```

```
  end
```

```
end
```

```
return (UNSAT,  $\mathbb{I}$ )
```

Real root isolation over a partial sample point

Select sample from $\mathbb{R} \setminus \mathbb{I}$

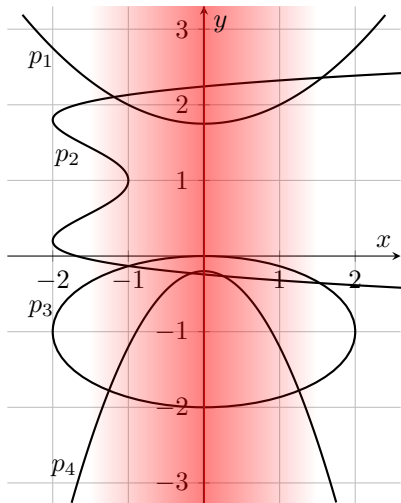
Recurse to next variable

CAD-style projection:
Roots of polynomials restrict where covering is still applicable

Extract interval from polynomials



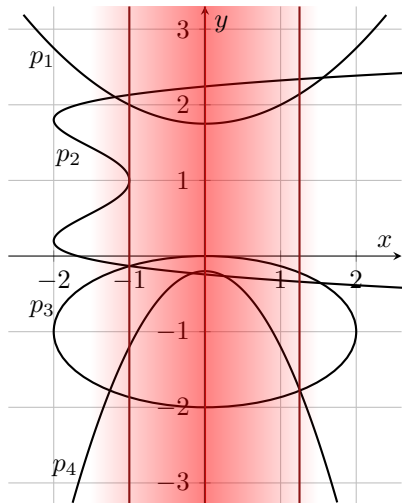
construct_characterization



Identify region around sample



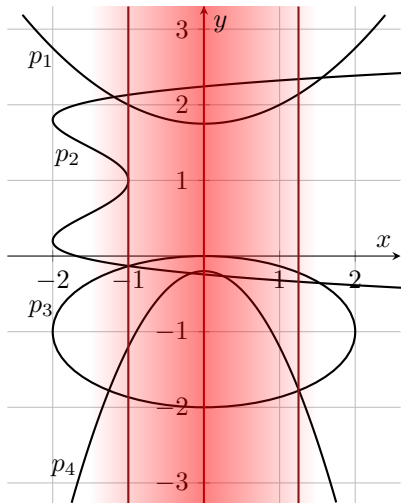
construct_characterization



Identify region around sample



construct_characterization



Identify region around sample

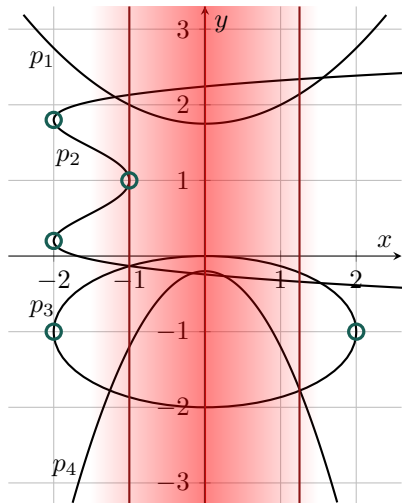
CAD projection:

Discriminants (and coefficients)

Resultants



construct_characterization



Identify region around sample

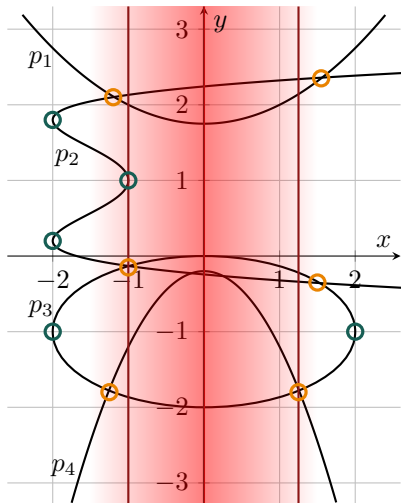
CAD projection:

Discriminants (and coefficients)

Resultants



construct_characterization



Identify region around sample

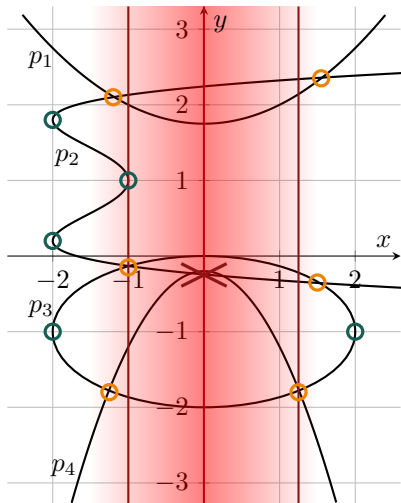
CAD projection:

Discriminants (and coefficients)

Resultants



construct_characterization



Identify region around sample
CAD projection:

Discriminants (and coefficients)
Resultants

Improvement over CAD:

Resultants between
neighbouring intervals only!



Other methods for (QF_)NRA

- ▶ Numerical methods [Kremer 2013]:
focus on **good approximation**, but no **formal guarantees**
- ▶ Tarski's method [Tarski 1951]:
theoretical breakthrough only, non-elementary complexity
- ▶ Grigor'ev and Vorobjov [Grigor'ev et al. 1988], Renegar [Renegar 1988]:
singly exponential, but impractical (see [Hong 1991])
- ▶ Basu, Pollack and Roy [Basu et al. 1996]:
"realizable sign conditions", **has not been implemented** (yet)
- ▶ Other CAD-based methods:
Regular Chains [Chen et al. 2009], NuCAD [Brown 2015]



Beyond QF_NRA

- ▶ Quantifiers:
 - ▶ Theory of the Reals **admits quantifier elimination**
 - ▶ CAD constructs φ' for $Q_x \varphi(x, y) \Leftrightarrow \varphi'(y)$
- ▶ Theory combination with Array, BV, FP, String, ... [Nelson et al. 1979]
- ▶ **Transcendentals**: extend linearization [Cimatti et al. 2018] [Irfan 2018]
- ▶ **Optimization**: CAD can **optimize for an objective** [Kremer 2020]
- ▶ **Integers**: Branch&Bound complements BitBlasting [Kremer et al. 2016]



Beyond CDCL(T)-style SMT

Other approaches for (QF_)NRA:

- ▶ MCSAT / NLSAT:
 - ▶ Theory model construction integrated in the core solver
 - ▶ SMT-RAT, yices, z3 [Jovanović et al. 2012] [Jovanović et al. 2013] [Moura et al. 2013] [Nalbach et al. 2019] [Kremer 2020]
- ▶ CAD is a **stand-alone tool**:
 - ▶ Maple / RegularChains [Chen et al. 2009]
 - ▶ Mathematica [Strzeboński 2014]
 - ▶ QEPCAD B [Brown 2003]
 - ▶ Redlog / Reduce [Dolzmann et al. 1997]

These can be **integrated as theory solvers** [Fontaine et al. 2018] [Kremer 2018]



in CVC4

- ▶ Linearization (`--nl-ext`)
- ▶ CDCAC (`--nl-cad`)
- ▶ Also: ICP-style propagations (`--nl-icp`)



in CVC4

- ▶ Linearization (`--nl-ext`)
- ▶ CDCAC (`--nl-cad`)
- ▶ Also: ICP-style propagations (`--nl-icp`)

in progress / future work:

- ▶ Better **integration** of Linearization, CDCAC and ICP
- ▶ **Preprocessing** for nonlinear arithmetic
- ▶ Add **proofs**
- ▶ Improve **incrementality** (in particular CDCAC)
- ▶ Improvements **within CDCAC** (heuristics, factorization, ...)



References I

- ▶ Erika Ábrahám, James H. Davenport, Matthew England, and Gereon Kremer. “Deciding the Consistency of Non-Linear Real Arithmetic Constraints with a Conflict Driven Search Using Cylindrical Algebraic Coverings”. In: *arXiv e-prints* (2020). Accepted at *Journal of Logical and Algebraic Methods in Programming*. arXiv: 2003.05633.
- ▶ Saugata Basu, Richard Pollack, and Marie-Françoise Roy. “On the Combinatorial and Algebraic Complexity of Quantifier Elimination”. In: *Journal of the ACM* 43 (6 1996), pp. 1002–1045. DOI: 10.1145/235809.235813.
- ▶ Frédéric Benhamou and Laurent Granvilliers. “Continuous and Interval Constraints”. In: *Handbook of Constraint Programming*, Vol. 2. 2006, pp. 571–603. DOI: 10.1016/S1574-6526(06)80020-9.
- ▶ Christopher W. Brown. “Improved Projection for Cylindrical Algebraic Decomposition”. In: *Journal of Symbolic Computation* 32 (5 2001), pp. 447–465. DOI: 10.1006/jsc.2001.0463.
- ▶ Christopher W. Brown. “Qepcad b: A program for computing with semi-algebraic sets using CADs”. In: *ACM SIGSAM Bulletin* 37 (4 2003), pp. 97–108. doi: 10.1145/968708.968710.
- ▶ Christopher W. Brown. “Constructing a Single Open Cell in a Cylindrical Algebraic Decomposition”. In: *ISSAC*. 2013, pp. 133–140. doi: 10.1145/2465506.2465952.
- ▶ Christopher W. Brown. “Open Non-uniform Cylindrical Algebraic Decompositions”. In: *ISSAC*. 2015, pp. 85–92. doi: 10.1145/2755996.2756654.
- ▶ Christopher W. Brown and Marek Košta. “Constructing a single cell in cylindrical algebraic decomposition”. In: *Journal of Symbolic Computation* 70 (2015), pp. 14–48. doi: 10.1016/j.jsc.2014.09.024.



References II

- ▶ [Bruno Buchberger](#). “Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal”. PhD thesis. University of Innsbruck, 1965.
- ▶ [Changbo Chen, Marc Moreno Maza, Bican Xia, and Lu Yang](#). “Computing Cylindrical Algebraic Decomposition via Triangular Decomposition”. In: *ISSAC*. 2009, pp. 95–102. doi: [10.1145/1576702.1576718](https://doi.org/10.1145/1576702.1576718).
- ▶ [Alessandro Cimatti, Alberto Griggio, Ahmed Irfan, Marco Roveri, and Roberto Sebastiani](#). “Incremental Linearization for Satisfiability and Verification Modulo Nonlinear Arithmetic and Transcendental Functions”. In: *ACM Transactions on Computational Logic* 19 (3 2018), 19:1–19:52. doi: [10.1145/3230639](https://doi.org/10.1145/3230639).
- ▶ [George E. Collins](#). “Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition—Preliminary Report”. In: *ACM SIGSAM Bulletin* 8 (3 1974), pp. 80–90. doi: [10.1145/1086837.1086852](https://doi.org/10.1145/1086837.1086852).
- ▶ [George E. Collins](#). “Quantifier Elimination by Cylindrical Algebraic Decomposition — Twenty Years of Progress”. In: *Quantifier Elimination and Cylindrical Algebraic Decomposition*. 1998, pp. 8–23. doi: [10.1007/978-3-7091-9459-1_2](https://doi.org/10.1007/978-3-7091-9459-1_2).
- ▶ [Andreas Dolzmann and Thomas Sturm](#). “REDLOG: Computer Algebra Meets Computer Logic”. In: *ACM SIGSAM Bulletin* 31 (2 1997), pp. 2–9. doi: [10.1145/261320.261324](https://doi.org/10.1145/261320.261324).
- ▶ [Matthew England, Russell Bradford, and James H. Davenport](#). “Improving the Use of Equational Constraints in Cylindrical Algebraic Decomposition”. In: *ISSAC*. 2015, pp. 165–172. doi: [10.1145/2755996.2756678](https://doi.org/10.1145/2755996.2756678).



References III

- ▶ Matthew England, Russell Bradford, James H. Davenport, and David Wilson. “Choosing a Variable Ordering for Truth-Table Invariant Cylindrical Algebraic Decomposition by Incremental Triangular Decomposition”. In: **ICMS**. Vol. 8592. 2014. doi: 10.1007/978-3-662-44199-2_68.
- ▶ Dorian Florescu and Matthew England. “Algorithmically Generating New Algebraic Features of Polynomial Systems for Machine Learning”. In: **SC²**. SIAM AG. Vol. 2460. 2019. url: <http://ceur-ws.org/Vol-2460/paper4.pdf>.
- ▶ Pascal Fontaine, Mizuhito Ogawa, Thomas Sturm, Van Khanh To, and Xuan Tung Vu. “Wrapping Computer Algebra is Surprisingly Successful for Non-Linear SMT”. In: **SC²**. FLoC. Vol. 2189. 2018, pp. 110–117. url: <http://ceur-ws.org/Vol-2189/paper3.pdf>.
- ▶ Pascal Fontaine, Mizuhito Ogawa, Thomas Sturm, and Xuan Tung Vu. “Subtropical Satisfiability”. In: **FroCoS**. 2017, pp. 189–206.
- ▶ Sicun Gao, Soonho Kong, and Edmund M. Clarke. “dReal: An SMT Solver for Nonlinear Theories over the Reals”. In: **CADE-24**. Vol. 7898. 2013, pp. 208–214. doi: 10.1007/978-3-642-38574-2_14.
- ▶ D. Yu. Grigor'ev and N.N. Vorobjov. “Solving Systems of Polynomial Inequalities in Subexponential Time”. In: **Journal of Symbolic Computation** 5 (1–2 1988), pp. 37–64. doi: 10.1016/S0747-7171(88)80005-1.
- ▶ Rebecca Haehn, Gereon Kremer, and Erika Ábrahám. “Evaluation of Equational Constraints for CAD in SMT Solving”. In: **SC²**. FLoC. Vol. 2189. 2018, pp. 19–32. url: <http://ceur-ws.org/Vol-2189/paper10.pdf>.



References IV

- ▶ **Hoon Hong**. “An Improvement of the Projection Operator in Cylindrical Algebraic Decomposition”. In: *ISSAC*. 1990, pp. 261–264. doi: 10.1145/96877.96943.
- ▶ **Hoon Hong**. *Comparison of Several Decision Algorithms for the Existential Theory of the Reals*. Research rep. Johannes Kepler University, 1991, pp. 1–33.
- ▶ **Zongyan Huang, Matthew England, David Wilson, James H. Davenport, Lawrence C. Paulson, and James Bridge**. “Applying Machine Learning to the Problem of Choosing a Heuristic to Select the Variable Ordering for Cylindrical Algebraic Decomposition”. In: *CICM*. 2014, pp. 92–107. doi: 10.1007/978-3-319-08434-3_8.
- ▶ **Ahmed Irfan**. “Incremental Linearization for Satisfiability and Verification Modulo Nonlinear Arithmetic and Transcendental Functions”. *PhD thesis*. University of Trento, 2018. url: <http://eprints-phd.biblio.unitn.it/2952/>.
- ▶ **Dejan Jovanović, Clark Barrett, and Leonardo de Moura**. “The Design and Implementation of the Model Constructing Satisfiability Calculus”. In: *FMCAD*. 2013, pp. 173–180. doi: 10.1109/FMCAD.2013.7027033.
- ▶ **Dejan Jovanović and Leonardo de Moura**. “Solving Non-linear Arithmetic”. In: *IJCAR*. Vol. 7364. 2012, pp. 339–354. doi: 10.1007/978-3-642-31365-3_27.
- ▶ **Sebastian Junges**. “On Gröbner Bases in SMT-Compliant Decision Procedures”. *Bachelor’s thesis*. RWTH Aachen University, 2012.
- ▶ **Marek Košta**. “New Concepts for Real Quantifier Elimination by Virtual Substitution”. *PhD thesis*. Saarland University, Saarbrücken, Germany, 2016. doi: 10.22028/D291-26679.



References V

- ▶ **Marek Košta and Thomas Sturm.** “A Generalized Framework for Virtual Substitution”. In: **arXiv e-prints** (2015). arXiv: 1501.05826.
- ▶ **Gereon Kremer.** “Isolating Real Roots Using Adaptable-Precision Interval Arithmetic”. **Master's thesis.** RWTH Aachen University, 2013.
- ▶ **Gereon Kremer.** “Computer Algebra and Computer Science”. In: **ACA. Abstract.** 2018, p. 27. doi: 10.15304/9788416954872.
- ▶ **Gereon Kremer.** “Cylindrical Algebraic Decomposition for Nonlinear Arithmetic Problems”. **PhD thesis.** RWTH Aachen University, 2020. url: <http://aib.informatik.rwth-aachen.de/2020/2020-04.pdf>.
- ▶ **Gereon Kremer, Florian Corzilius, and Erika Ábrahám.** “A Generalised Branch-and-Bound Approach and Its Application in SAT Modulo Nonlinear Integer Arithmetic”. In: **CASC.** Vol. 9890. 2016, pp. 315–335. doi: 10.1007/978-3-319-45641-6_21.
- ▶ **Daniel Lazard.** “An Improved Projection for Cylindrical Algebraic Decomposition”. In: **Algebraic Geometry and its Applications.** 1994. Chap. 29, pp. 467–476. doi: 10.1007/978-1-4612-2628-4_29.
- ▶ **Ulrich Loup, Karsten Scheibler, Florian Corzilius, Erika Ábrahám, and Bernd Becker.** “A Symbiosis of Interval Constraint Propagation and Cylindrical Algebraic Decomposition”. In: **CADE-24.** Vol. 7898. 2013, pp. 193–207. doi: 10.1007/978-3-642-38574-2_13.



References VI

- ▶ **Scott McCallum**. “An Improved Projection Operation for Cylindrical Algebraic Decomposition”. PhD thesis. University of Wisconsin-Madison, 1984. url: <https://research.cs.wisc.edu/techreports/1985/TR578.pdf>.
- ▶ **Scott McCallum**. “An Improved Projection Operation for Cylindrical Algebraic Decomposition of Three-dimensional Space”. In: *Journal of Symbolic Computation* 5 (1–2 1988), pp. 141–161. doi: 10.1016/S0747-7171(88)80010-5.
- ▶ **Scott McCallum**. “On Projection in CAD-based Quantifier Elimination with Equational Constraint”. In: *ISSAC*. 1999, pp. 145–149. doi: 10.1145/309831.309892.
- ▶ **Scott McCallum**. “On Propagation of Equational Constraints in CAD-based Quantifier Elimination”. In: *ISSAC*. 2001, pp. 223–231. doi: 10.1145/384101.384132.
- ▶ **Scott McCallum and Hoon Hong**. “On using Lazard’s projection in CAD construction”. In: *Journal of Symbolic Computation* 72 (2016), pp. 65–81. doi: 10.1016/j.jsc.2015.02.001.
- ▶ **Scott McCallum, Adam Parusiński, and Laurentiu Paunescu**. “Validity proof of Lazard’s method for CAD construction”. In: *Journal of Symbolic Computation* 92 (2019), pp. 52–69. doi: 10.1016/j.jsc.2017.12.002.
- ▶ **Leonardo de Moura and Dejan Jovanović**. “A Model-Constructing Satisfiability Calculus”. In: *VMCAI*. Vol. 7737. 2013, pp. 1–12. doi: 10.1007/978-3-642-35873-9_1.
- ▶ **Akshar Nair, James Davenport, and Gregory Sankaran**. “On Benefits of Equality Constraints in Lex-Least Invariant CAD”. In: *SC²*. SIAM AG. Vol. 2460. 2019. url: <http://ceur-ws.org/Vol-2460/paper6.pdf>.



References VII

- ▶ **Jasper Nalbach**. “Embedding the Virtual Substitution in the MCSAT Framework”. *Bachelor's thesis*. RWTH Aachen University, 2017.
- ▶ **Jasper Nalbach, Gereon Kremer, and Erika Ábrahám**. “On Variable Orderings in MCSAT for Non-linear Real Arithmetic (extended abstract)”. In: *SC²*. SIAM AG. Vol. 2460. 2019. url: <http://ceur-ws.org/Vol-2460/paper5.pdf>.
- ▶ **Greg Nelson and Derek C. Oppen**. “Simplification by Cooperating Decision Procedures”. In: *ACM Transactions on Programming Languages and Systems* 1 (2 1979), pp. 245–257. doi: 10.1145/357073.357079.
- ▶ **James Renegar**. “A Faster PSPACE Algorithm for Deciding the Existential Theory of the Reals”. In: *SFCS*. 1988, pp. 291–295. doi: 10.1109/SFCS.1988.21945.
- ▶ **Karsten Scheibler, Stefan Kupferschmid, and Bernd Becker**. “Recent Improvements in the SMT Solver iSAT”. In: *MBMV*. Vol. 13. 2013, pp. 231–241.
- ▶ **Stefan Schupp**. “Interval Constraint Propagation in SMT Compliant Decision Procedures”. *Master's thesis*. RWTH Aachen University, 2013.
- ▶ **Andreas Seidl and Thomas Sturm**. “A Generic Projection Operator for Partial Cylindrical Algebraic Decomposition”. In: *ISSAC*. 2003, pp. 240–247. doi: 10.1145/860854.860903.
- ▶ **Adam W. Strzeboński**. “Solving Systems of Strict Polynomial Inequalities”. In: *Journal of Symbolic Computation* 29 (3 2000), pp. 471–480. doi: 10.1006/jsco.1999.0327.
- ▶ **Adam W. Strzeboński**. “Cylindrical Algebraic Decomposition Using Local Projections”. In: *ISSAC*. 2014, pp. 389–396. doi: 10.1145/2608628.2608633.



References VIII

- ▶ Alfred Tarski. A Decision Method for Elementary Algebra and Geometry. Research rep. RAND Corporation, 1951. url: <https://www.rand.org/pubs/reports/R109.html>.
- ▶ Vu Xuan Tung, To Van Khanh, and Mizuhito Ogawa. “raSAT: an SMT solver for polynomial constraints”. In: *Formal Methods in System Design* 51 (3 2017), pp. 462–499. doi: 10.1007/s10703-017-0284-9.
- ▶ Volker Weispfenning. “The Complexity of Linear Problems in Fields”. In: *Journal of Symbolic Computation* 5 (1–2 1988), pp. 3–27. doi: 10.1016/S0747-7171(88)80003-8.
- ▶ Volker Weispfenning. “Quantifier Elimination for Real Algebra — the Quadratic Case and Beyond”. In: *Applicable Algebra in Engineering, Communication and Computing* 8 (2 1997), pp. 85–101. doi: 10.1007/s002000050055.