MCSAT BASED APPROACHES FOR NON-LINEAR MODULAR ARITHMETIC

Jakob Rath[1] Clemens Eisenhofer[1] Thomas Hader [1]
Daniela Kaufmann[1] Laura Kovács[1] Nikolaj Bjørner[2]
[1] TU Wien [2] Microsoft Research

Satisfiability: Theory, Practice, and Beyond

Extended Reunion: Satisfiability

Simons Institute, Berkeley, CA, US

April 20, 2023

Modulo 5

$$x^2 - 1 = 0$$

$$xy - y - 1 = 0$$

$$xy - 2 \neq 0$$

Modulo 5

$$x^2 - 1 = 0$$

$$xy - y - 1 = 0$$

$$xy - 2 \neq 0$$

$\cdot$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Modulo 5

$$x^2 - 1 = 0$$
$$xy - y - 1 = 0$$
$$xy - 2 \neq 0$$

·	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

$$x \mapsto 4, y \mapsto 2$$

Solving polynomial equations

Solving polynomial equations is one of the oldest and hardest problem in mathematics.

Algebraic closed fields: decidable (Gröbner bases)

Finite domains are not algebraically closed

Solving polynomial equations

Solving polynomial equations is one of the oldest and hardest problem in mathematics.

Algebraic closed fields: decidable (Gröbner bases)
Finite domains are not algebraically closed

Non-linear polynomial reasoning over finite domains is currently of interest in automated reasoning over cryptosystems:

- Finite field $\mathbb{F}_q[X]$: Zero-knowledge proofs, elliptic curve cryptography
- $\mathbb{Z}/2^k\mathbb{Z}[X]$: Bit-vector solving, e.g. in smart contracts

Solving polynomial equations

Computer Algebra

- Until recently solving polynomial constraints was the sole domain of computer algebra.
- Powerful in finding all solutions
- High computational overhead

Solving polynomial equations

Computer Algebra

- Until recently solving polynomial constraints was the sole domain of computer algebra.
- Powerful in finding all solutions
- High computational overhead

We are typically not interested in finding all solutions!

Solving polynomial equations

Computer Algebra

- Until recently solving polynomial constraints was the sole domain of computer algebra.
- Powerful in finding all solutions
- High computational overhead

Model Constructing Satisfiability (MCSat) **[Jovanović et al., VMCAI'13]**

- Finding satisfiable instances of polynomial arithmetic.
- Combines CDCL-style search with algebraic decompositions.

MCSAT based approaches for non-linear modular arithmetic

1. Constraints in $\mathbb{F}_q[X]$

- Finite field
- Not algebraically closed
- Constraints: $=, \neq$

Modulo 5

$$\begin{aligned} x^2 - 1 &= 0 \\ xy - y - 1 &= 0 \\ xy - 2 &\neq 0 \end{aligned}$$

$\Rightarrow$ FFSat
Thomas Hader, Daniela Kaufmann,
Laura Kovács

MCSAT based approaches for non-linear modular arithmetic

1. Constraints in $\mathbb{F}_q[X]$

- Finite field
- Not algebraically closed
- Constraints: $=, \neq$

Modulo 5

$$x^2 - 1 = 0$$
$$xy - y - 1 = 0$$
$$xy - 2 \neq 0$$

$\Rightarrow$ FFSat

Thomas Hader, Daniela Kaufmann, Laura Kovács

2. Constraints in $\mathbb{Z}/2^k\mathbb{Z}[X]$

- Finite commutative ring
- Not algebraically closed
- Constraints: $=, \neq, <, >, \Omega^*(x, y)$

Modulo 2^4

$$xy + y \leq y + 3$$
$$2y + z = 10$$
$$3x + 6yz + 3z^2 = 1$$

$\Rightarrow$ PolySat

Nikolaj Bjørner, Clemens Eisenhofer, Daniela Kaufmann, Laura Kovács, Jakob Rath

MCSat

MCSat **[Jovanović et al., VMCAI'13]**

- abstract CDCL decision procedure
- integrates theory reasoning in the boolean search engine
- incrementally constructs model while searching
- propagated literals are justified by model assignments

MCSat

MCSat **[Jovanović et al., VMCAI'13]**

- abstract CDCL decision procedure
- integrates theory reasoning in the boolean search engine
- incrementally constructs model while searching
- propagated literals are justified by model assignments

Successfully applied in the theories of

- non-linear arithmetic constraints over reals **[Jovanović et al., IJCAR'12]**
- linear integer constraints **[Jovanović et al., CADE'11]**
- bitvectors **[Zeljić et al., SAT'16]**

MCSat - Idea

From a given set of clauses $\mathcal{C}$, generate a trail Γ with decided and propagated literals and theory variable assignments that leads to one of the two terminal states $\textsc{unsat}$ or $\textsc{sat}$.

Polynomial system is a set of unit clauses.

MCSat - Idea

From a given set of clauses $\mathcal{C}$, generate a trail Γ with decided and propagated literals and theory variable assignments that leads to one of the two terminal states UNSAT or SAT.

Main components:

- Trail Γ records assignments and reasons
- For each variable x, keep track of viable values V_x
- Conflict C: set of constraints that contradicts Γ
- Conflict analysis learn a new constraint to avoid the conflict in the future

MCSat - Idea

From a given set of clauses $\mathcal{C}$, generate a trail Γ with decided and propagated literals and theory variable assignments that leads to one of the two terminal states UNSAT or SAT.

Variables $x_1 < x_2 < \ldots < x_n$

$$\Gamma = [\![F_1, \ldots, F_l, x_1 \mapsto \alpha_1, G_1, \ldots, G_m, x_2 \mapsto \alpha_2, H_1, \ldots, H_n, \ldots]\!]$$

literals F_i over $[x_1]$, G_i over $[x_1, x_2]$, H_i over $[x_1, x_2, x_3]$.

MCSat - Idea

From a given set of clauses $\mathcal{C}$, generate a trail Γ with decided and propagated literals and theory variable assignments that leads to one of the two terminal states UNSAT or SAT.

Variables $x_1 < x_2 < \ldots < x_n$

$$\Gamma = [\![F_1, \ldots, F_l, x_1 \mapsto \alpha_1, G_1, \ldots, G_m, x_2 \mapsto \alpha_2, H_1, \ldots, H_n, \ldots]\!]$$

literals F_i over $[x_1]$, G_i over $[x_1, x_2]$, H_i over $[x_1, x_2, x_3]$.

Regular **boolean propagation**:

Clause $\mathcal{C}_1 = \{\neg F_1, \neg G_2, H\}$
Add literal H to the trail with justification $\mathcal{C}_1$.

MCSat - Idea

From a given set of clauses $\mathcal{C}$, generate a trail Γ with decided and propagated literals and theory variable assignments that leads to one of the two terminal states UNSAT or SAT.

Variables $x_1 < x_2 < \ldots < x_n$

$$\Gamma = [\![F_1, \ldots, F_l, x_1 \mapsto \alpha_1, G_1, \ldots, G_m, x_2 \mapsto \alpha_2, H_1, \ldots, H_n, \ldots]\!]$$

literals F_i over $[x_1]$, G_i over $[x_1, x_2]$, H_i over $[x_1, x_2, x_3]$.

In addition **theory propagation**:

> **Idea**: From theory (i.e. variable assignments) we know that literal H can't hold, $\neg H$ can be propagated.
> Generate explanation clause E that justifies $\neg H$.

Example: Polynomial Equations

$$C_1: \quad x^2 - 1 = 0 \mod 5$$
$$C_2: \quad xy - y - 1 = 0 \mod 5$$

1. $\Gamma = [\![(x^2 - 1 = 0)]\!]$ decide literal

Example: Polynomial Equations

$$C_1: \quad x^2 - 1 = 0 \mod 5$$
$$C_2: \quad xy - y - 1 = 0 \mod 5$$

1. $\Gamma = [\![(x^2 - 1 = 0)]\!]$ decide literal

   $\rightsquigarrow C_1|_\Gamma : x^2 - 1 = 0 \quad \Rightarrow x = 1 \vee x = 4$

Example: Polynomial Equations

$$C_1: \quad x^2 - 1 = 0 \mod 5$$
$$C_2: \quad xy - y - 1 = 0 \mod 5$$

1. $\Gamma = [\![(x^2 - 1 = 0)]\!]$ decide literal
   $\rightsquigarrow C_1|_\Gamma : x^2 - 1 = 0 \quad \Rightarrow x = 1 \vee x = 4$
2. $\Gamma = [\![(x^2 - 1 = 0)^\delta, (x \mapsto 1)^{C_1}]\!]$ decision on x

Example: Polynomial Equations

$$C_1: \quad x^2 - 1 = 0 \mod 5$$
$$C_2: \quad xy - y - 1 = 0 \mod 5$$

1. $\Gamma = [\![(x^2 - 1 = 0)]\!]$ — decide literal
   $\rightsquigarrow C_1|_\Gamma : x^2 - 1 = 0 \quad \Rightarrow x = 1 \lor x = 4$
2. $\Gamma = [\![(x^2 - 1 = 0)^\delta, (x \mapsto 1)^{C_1}]\!]$ — decision on x
3. $\Gamma = [\![(x^2 - 1 = 0)^\delta, (x \mapsto 1)^{C_1,\delta}, (xy - y - 1 = 0)]\!]$ — add C_2

Example: Polynomial Equations

$$C_1: \quad x^2 - 1 = 0 \mod 5$$
$$C_2: \quad xy - y - 1 = 0 \mod 5$$

1. $\Gamma = [\![(x^2 - 1 = 0)]\!]$ — decide literal
   $\rightsquigarrow C_1|_\Gamma : x^2 - 1 = 0 \quad \Rightarrow x = 1 \vee x = 4$
2. $\Gamma = [\![(x^2 - 1 = 0)^\delta, (x \mapsto 1)^{C_1}]\!]$ — decision on x
3. $\Gamma = [\![(x^2 - 1 = 0)^\delta, (x \mapsto 1)^{C_1,\delta}, (xy - y - 1 = 0)]\!]$ — add C_2
   $\rightsquigarrow C_2|_\Gamma : \ -1 = 0$
   Conflict: $\mathcal{C} = \{C_2, x = 1, C_1\}$
   Generate explanation clause $E = \{x + 1 = 0, \neg C_2\}$ using theory propagation.
   To satisfy C_2 we resolve using E and backtrack to assign a different value to x.

Explain Function

Informal: Bring theory knowledge into the search procedure on demand.

Key ingredient for every MCSat procedure is the explain function!

MCSAT based approaches for non-linear modular arithmetic

1. Constraints in $\mathbb{F}_q[X]$

- Finite field
- Not algebraically closed
- Constraints: $=, \neq$

Modulo 5

$$x^2 - 1 = 0$$
$$xy - y - 1 = 0$$
$$xy - 2 \neq 0$$

$\Rightarrow$ FFSat

2. Constraints in $\mathbb{Z}/2^k\mathbb{Z}[X]$

- Finite commutative ring
- Not algebraically closed
- Constraints: $=, \neq, <, >, \Omega^*(x, y)$

Modulo 2^4

$$xy + y \leq y + 3$$
$$2y + z = 10$$
$$3x + 6yz + 3z^2 = 1$$

$\Rightarrow$ PolySat

Finite Fields

A field is a set of elements closed under operations sum, difference, product and inverse finding.

Finite Fields

A field is a set of elements closed under operations sum, difference, product and inverse finding.

A finite field is a field with a finite amount of elements.

Given a number $q = p^n$ with p prime and $n \geq 1$:

$\mathbb{F}_q$ denotes a finite field of size q.

Finite Fields

A field is a set of elements closed under operations sum, difference, product and inverse finding.

A finite field is a field with a finite amount of elements.

Given a number $q = p^n$ with p prime and $n \geq 1$:

$\mathbb{F}_q$ denotes a finite field of size q.

Example

For $q = 5$ the field $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$.

- $(2 \cdot 3) + 4 = 0$
- inverse of 2 is 3, as $2 \cdot 3 = 1$

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \ldots, x_n]$!

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \ldots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \ldots, F_l]\!]$$

for $1 \leq i \leq l$: $x_k \in \mathsf{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \mathsf{true}$

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \ldots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \ldots, F_l]\!]$$

for $1 \leq i \leq l$: $x_k \in \text{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \text{true}$

- New constraint G on trail such that α_k does not exist any more.

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \dots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\dots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \dots, F_l]\!]$$

for $1 \leq i \leq l$: $x_k \in \mathsf{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \mathsf{true}$

- New constraint G on trail such that α_k does not exist any more.
- Eliminate x_k in $\{F_1, \dots, F_l, \neg G\}$ and generate polynomial set $\mathcal{C} \subset \mathbb{F}_q[x_1, \dots, x_{k-1}]$

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \dots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\dots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \dots, F_l]\!]$$

for $1 \le i \le l$: $x_k \in \text{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \text{true}$

- New constraint G on trail such that α_k does not exist any more.
- Eliminate x_k in $\{F_1, \dots, F_l, \neg G\}$ and generate polynomial set $\mathcal{C} \subset \mathbb{F}_q[x_1, \dots, x_{k-1}]$
- $\nu[\Gamma](\mathcal{C}) = \text{false}$

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \ldots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \ldots, F_l]\!]$$

for $1 \leq i \leq l$: $x_k \in \text{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \text{true}$

- New constraint G on trail such that α_k does not exist any more.
- Eliminate x_k in $\{F_1, \ldots, F_l, \neg G\}$ and generate polynomial set $\mathcal{C} \subset \mathbb{F}_q[x_1, \ldots, x_{k-1}]$
- $\nu[\Gamma](\mathcal{C}) = \text{false}$
- Set $\mathcal{E} = \{\neg F_1, \ldots, \neg F_l,\} \cup \{G\} \cup \mathcal{C}$

Explanation Generation

Generate an explanation function for constraints over $\mathbb{F}_q[x_1, \ldots, x_n]$!

General Idea: Given a trail Γ

$$\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, F_2, \ldots, F_l]\!]$$

for $1 \leq i \leq l$: $x_k \in \mathsf{var}(F_i)$ and $\exists \alpha_k \in \mathbb{F}_q$ s.t. $\nu[\Gamma][x_k \mapsto \alpha_k](F_i) = \mathsf{true}$

- New constraint G on trail such that α_k does not exist any more.
- Eliminate x_k in $\{F_1, \ldots, F_l, \neg G\}$ and generate polynomial set $\mathcal{C} \subset \mathbb{F}_q[x_1, \ldots, x_{k-1}]$
- $\nu[\Gamma](\mathcal{C}) = \mathsf{false}$
- Set $\mathcal{E} = \{\neg F_1, \ldots, \neg F_l, \} \cup \{G\} \cup \mathcal{C}$

Variable Elimination

Given a set of polynomials $\mathcal{P} \subset \mathbb{F}_q[x_1, \ldots, x_k]$.

We eliminate x_k by generating set $\mathcal{P}' \subset \mathbb{F}_q[x_1, \ldots, x_{k-1}]$ s.t.

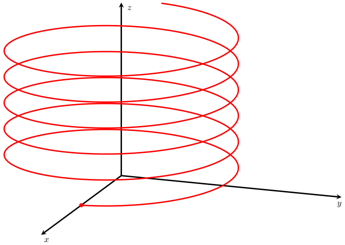
$$(\alpha_1, \ldots, \alpha_{k-1}) \in \mathsf{zero}(\mathcal{P}') \quad \text{iff} \quad \exists \beta \in \mathbb{F}_q. (\alpha_1, \ldots, \alpha_{k-1}, \beta) \in \mathsf{zero}(\mathcal{P})$$

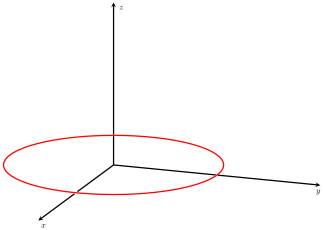
Variable Elimination

Given a set of polynomials $\mathcal{P} \subset \mathbb{F}_q[x_1, \ldots, x_k]$.

We eliminate x_k by generating set $\mathcal{P}' \subset \mathbb{F}_q[x_1, \ldots, x_{k-1}]$ s.t.

$$(\alpha_1, \ldots, \alpha_{k-1}) \in \mathsf{zero}(\mathcal{P}') \quad \text{iff} \quad \exists \beta \in \mathbb{F}_q.\,(\alpha_1, \ldots, \alpha_{k-1}, \beta) \in \mathsf{zero}(\mathcal{P})$$





Single incompatibility

- Let $\Gamma = [\![(x^2 - 1 = 0), x \mapsto 1]\!]$ and $G := (xy - y - 1 = 0)$ is incompatible

Single incompatibility

- Let $\Gamma = [\![(x^2 - 1 = 0), x \mapsto 1]\!]$ and $G := (xy - y - 1 = 0)$ is incompatible
- Assignment $\nu[\Gamma][y \mapsto \alpha_y]$ violates G for all $\alpha_y \in \mathbb{F}_q$

$$(x-1) \cdot y - 1 \qquad\qquad (x-1) \in \mathbb{F}_q[x]$$

Single incompatibility

- Let $\Gamma = [\![(x^2 - 1 = 0), x \mapsto 1]\!]$ and $G := (xy - y - 1 = 0)$ is incompatible
- Assignment $\nu[\Gamma][y \mapsto \alpha_y]$ violates G for all $\alpha_y \in \mathbb{F}_q$

$$(x-1) \cdot y - 1 \qquad\qquad (x-1) \in \mathbb{F}_q[x]$$

- Exclude all assignments with the same coefficient evaluation
  - evaluate coefficients $\nu[\Gamma](x-1) = 0$
  - define clause $\{(x-1) - 0 \neq 0\}$

Single incompatibility

- Let $\Gamma = [\![(x^2 - 1 = 0), x \mapsto 1]\!]$ and $G := (xy - y - 1 = 0)$ is incompatible
- Assignment $\nu[\Gamma][y \mapsto \alpha_y]$ violates G for all $\alpha_y \in \mathbb{F}_q$

$$(x-1) \cdot y - 1 \qquad\qquad (x-1) \in \mathbb{F}_q[x]$$

- Exclude all assignments with the same coefficient evaluation
  - evaluate coefficients $\nu[\Gamma](x-1) = 0$
  - define clause $\{(x-1) - 0 \neq 0\}$
- Excludes (at least) the **current assignment** that violates a **single** constraint

Coefficient based explanation generation

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}]\!]$ and $G := (p = 0)$ is incompatible

Coefficient based explanation generation

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}]\!]$ and $G := (p = 0)$ is incompatible
- Assignment $\nu[\Gamma][x_k \mapsto \alpha_k]$ violates G for all $\alpha_k \in \mathbb{F}_q$

$$p = c_1 \cdot x_k^{d_1} + \cdots + c_m \cdot x_k^{d_m} \qquad c_i \in \mathbb{F}_q[x_1, \ldots, x_{k-1}]$$

Coefficient based explanation generation

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}]\!]$ and $G := (p = 0)$ is incompatible
- Assignment $\nu[\Gamma][x_k \mapsto \alpha_k]$ violates G for all $\alpha_k \in \mathbb{F}_q$

$$p = c_1 \cdot x_k^{d_1} + \cdots + c_m \cdot x_k^{d_m} \qquad c_i \in \mathbb{F}_q[x_1, \ldots, x_{k-1}]$$

- Exclude all assignments with the same coefficient evaluation
  - evaluate coefficients $\gamma_i = \nu[\Gamma](c_i)$
  - define clause $\{c_i - \gamma_i \neq 0 \mid 1 \leq i \leq m\}$

Coefficient based explanation generation

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}]\!]$ and $G := (p = 0)$ is incompatible
- Assignment $\nu[\Gamma][x_k \mapsto \alpha_k]$ violates G for all $\alpha_k \in \mathbb{F}_q$

$$p = c_1 \cdot x_k^{d_1} + \cdots + c_m \cdot x_k^{d_m} \qquad\qquad c_i \in \mathbb{F}_q[x_1, \ldots, x_{k-1}]$$

- Exclude all assignments with the same coefficient evaluation
  - evaluate coefficients $\gamma_i = \nu[\Gamma](c_i)$
  - define clause $\{c_i - \gamma_i \neq 0 \mid 1 \leq i \leq m\}$
- Similar for $G := (p \neq 0)$
- Excludes (at least) the **current assignment** that violates a **single** constraint

Multiple incompatibilities - Gröbner Basis

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, \ldots F_m]\!]$ and $G_1, \ldots, G_n$ are incompatible
- Gröbner basis with a lexicographical term ordering has the projection property.

Multiple incompatibilities - Gröbner Basis

- Let $\Gamma = [\![\ldots, x_{k-1} \mapsto \alpha_{k-1}, F_1, \ldots F_m]\!]$ and $G_1, \ldots, G_n$ are incompatible
- Gröbner basis with a lexicographical term ordering has the projection property.
- Introduce fresh variable z for negations

$$f'(x_1, \ldots, x_k, z) = z \cdot f(x_1, \ldots, x_k) - 1$$

- Field polynomials $\mathcal{FP} = \{x_i^q - x_i | x_i \in X\}$ are required.
- Generate the $k-1$ elimination ideal of

$$\langle F_1, \cdots, F_m, G_1, \ldots, G_n \rangle + \langle \mathcal{FP} \rangle$$

Multiple incompatibilities - Exclude factors

- Let $f, g \in \mathbb{F}_q[x_1, \ldots, x_k]$ and α an assignment
- Factor univariate polynomials $\nu[\Gamma](f) \in \mathbb{F}_q[x_k]$ and $\nu[\Gamma](g) \in \mathbb{F}_q[x_k]$
- Exclude common irreducible factors

Subresultant Regular Subchain

- **GCD w.r.t. assignment**
- Let $f, g \in \mathbb{F}_q[x_1, \ldots, x_k]$
- $\mathsf{srs}(f, g, x_k) = h_2, \ldots, h_r$
- $i = \mathsf{lc}(g, x_k)$ and $i_\ell = \mathsf{lc}(h_\ell, x_k)$

$$\gcd(f(\boldsymbol{\alpha}, x_k), g(\boldsymbol{\alpha}, x_k)) = h_\ell(\boldsymbol{\alpha}, x_k)$$

if $\boldsymbol{\alpha} \in \mathsf{zero}(\{i_{\ell+1}, \ldots, i_r\}/\{i, i_\ell\})$

Subresultant Regular Subchain

$\mathbb{F}_q[X]$

- **GCD w.r.t. assignment**
- Let $f, g \in \mathbb{F}_q[x_1, \ldots, x_k]$
- $\mathsf{srs}(f, g, x_k) = h_2, \ldots, h_r$
- $i = \mathsf{lc}(g, x_k)$ and $i_\ell = \mathsf{lc}(h_\ell, x_k)$

$$\gcd(f(\boldsymbol{\alpha}, x_k), g(\boldsymbol{\alpha}, x_k)) = h_\ell(\boldsymbol{\alpha}, x_k)$$

if $\boldsymbol{\alpha} \in \mathsf{zero}(\{i_{\ell+1}, \ldots, i_r\}/\{i, i_\ell\})$

Think of **"Euclidean Division algorithm"** w.r.t. current assignment

Example: SRS

$f = z^2 + yz + 4 = 0$ and $g = x + yz \neq 0 \in \mathbb{F}_5[x, y, z]$

Let $\alpha = \{x \mapsto 3, y \mapsto 1\}$ be the current assignment on Γ

- f and g are incompatible with Γ

Example: SRS

$$f = z^2 + yz + 4 = 0 \quad \text{and} \quad g = x + yz \neq 0 \in \mathbb{F}_5[x, y, z]$$

Let $\alpha = \{x \mapsto 3, y \mapsto 1\}$ be the current assignment on Γ

- f and g are incompatible with Γ
- $\mathsf{srs}(f, g, z) = [x + yz, x^2 - xy^2 - y^2]$
- Learn $x^2 - xy^2 - y^2 \neq 0$

In addition to $\{x \mapsto 3, y \mapsto 1\}$ we also exclude $\{x \mapsto 0, y \mapsto 0\}$ and $\{x \mapsto 3, y \mapsto 4\}$

Results

$\mathbb{F}_q[X]$

Type	q	n	c	FFSAT	GB	GBLEX
Craft	3	32	32	**25**	**25**	0
Craft	3	64	64	**25**	24	0
Craft	13	32	16	**19**	18	1
Craft	211	16	8	24	**25**	**25**
Rand	3	8	8	**25**	**25**	**25**
Rand	3	16	16	**12**	11	0
Rand	13	8	4	**25**	0	0
Rand	13	8	8	**1**	0	0
Rand	211	8	4	**17**	0	0
Rand	211	8	16	0	0	0

Instances solved by FFSAT, GB, and GBLEX, out of 25 polynomial systems per test set within 300 seconds.

MCSAT based approaches for non-linear modular arithmetic

1. Constraints in $\mathbb{F}_q[X]$

- Finite field
- Not algebraically closed
- Constraints: $=, \neq$

Modulo 5

$$
\begin{aligned}
x^2 - 1 &= 0 \\
xy - y - 1 &= 0 \\
xy - 2 &\neq 0
\end{aligned}
$$

$\Rightarrow$ FFSAT

2. Constraints in $\mathbb{Z}/2^k\mathbb{Z}[X]$

- Finite commutative ring
- Not algebraically closed
- Constraints: $=, \neq, <, >, \Omega^*(x, y)$

Modulo 2^4

$$
\begin{aligned}
xy + y &\leq y + 3 \\
2y + z &= 10 \\
3x + 6yz + 3z^2 &= 1
\end{aligned}
$$

$\Rightarrow$ POLYSAT

PolySAT: a Word-level Solver for Large Bitvectors

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Bitvectors?

1. Sequence of bits, e.g., `01011`
2. Fixed-width machine integers, e.g., `uint32_t`, `int64_t`
3. Modular arithmetic: $\mathbb{Z}/2^k\mathbb{Z}$

PolySAT: a Word-level Solver for Large Bitvectors

Bitvectors?

1. Sequence of bits, e.g., `01011`
2. Fixed-width machine integers, e.g., `uint32_t`, `int64_t`
3. Modular arithmetic: $\mathbb{Z}/2^k\mathbb{Z}$

Examples:

- $2x^2y + z = 3$
- $x + 3 \leq x + y$
- $\neg\Omega^*(x, y), \quad z = x \,\&\, y, \quad \ldots$

PolySAT: a Word-level Solver for Large Bitvectors

Bitvectors?

1. Sequence of bits, e.g., `01011`
2. Fixed-width machine integers, e.g., `uint32_t`, `int64_t`
3. Modular arithmetic: $\mathbb{Z}/2^k\mathbb{Z}$

Examples:

- $2x^2y + z = 3$
- $x + 3 \leq x + y$
- $\neg\Omega^*(x, y), \quad z = x \,\&\, y, \quad \ldots$

Natural target for many program verification tasks!

Certora and smart contract verification: 256-bit unsigned integers

Bitvector Pitfalls

$\mathbb{Z}/2^k\mathbb{Z}$ is a finite commutative ring, but not a field.

$x, y \geq 0 \not\Longrightarrow xy \geq x$	Overflow/wraparound: $3 \cdot 6 = 2 \mod 2^4$
$x, y \neq 0 \not\Longrightarrow xy \neq 0$	Zero divisors: $6 \cdot 8 = 0 \mod 2^4$
$x \leq y \not\Longrightarrow x - y \leq 0$	Usual inequality normalization fails

Bitvector Pitfalls

$\mathbb{Z}/2^k\mathbb{Z}$ is a finite commutative ring, but not a field.

$x, y \geq 0 \;\not\Rightarrow\; xy \geq x$ — Overflow/wraparound: $3 \cdot 6 = 2 \mod 2^4$

$x, y \neq 0 \;\not\Rightarrow\; xy \neq 0$ — Zero divisors: $6 \cdot 8 = 0 \mod 2^4$

$x \leq y \;\not\Rightarrow\; x - y \leq 0$ — Usual inequality normalization fails

Example

$x + 3 \leq x + y \mod 2^3$

- For $x = 0$: $3 \leq y \iff y \in \{3, 4, 5, 6, 7\}$
- For $x = 2$: $5 \leq 2 + y \iff y \in \{3, 4, 5\}$

Solving Approaches

- Bit-blasting

  Translate into boolean formula and use SAT solver

Solving Approaches

- Bit-blasting

  Translate into boolean formula and use SAT solver

- Int-blasting **[Zohar et al., VMCAI'22]**

  Translate into integer arithmetic with bound constraints and modulo operations

Solving Approaches

$\mathbb{Z}/2^k\mathbb{Z}[X]$

- Bit-blasting

  Translate into boolean formula and use SAT solver

- Int-blasting **[Zohar et al., VMCAI'22]**

  Translate into integer arithmetic with bound constraints and modulo operations

- MCSAT-based approaches **[Zeljić et al., SAT'16]** **[Graham-Lengrand et al., IJCAR'20]**

  Search for assignment to bitvector variables $\rightsquigarrow$ PolySAT

PolySAT Overview

$\mathbb{Z}/2^k\mathbb{Z}[X]$

- Theory solver for bitvector arithmetic
  - Input: conjunction of bitvector constraints
  - Output: SAT or UNSAT
- Based on modular integer arithmetic ($\mathbb{Z}/2^k\mathbb{Z}$)

PolySAT Overview

- Theory solver for bitvector arithmetic
  - Input: conjunction of bitvector constraints
  - Output: SAT or UNSAT
- Based on modular integer arithmetic ($\mathbb{Z}/2^k\mathbb{Z}$)
- Search for a model of the input constraints
  - Incrementally assign bitvector variables
  - Keep track of viable values for variables
  - Add lemmas on demand to generate explanation clauses

Bitvector Constraints in PolySAT

Inequalities	$p \leq q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \mathbin{\&} q$	
Structural	$r = p \ll q$, $r = p \gg q$	
Clauses	Disjunction of constraint literals	

Bitvector Constraints in PolySAT

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Inequalities	$p \leq q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \,\&\, q$	
Structural	$r = p \ll q$, $r = p \gg q$	
Clauses	Disjunction of constraint literals	
By Reduction:		
Equations	$p = q \iff p - q \leq 0$	

Bitvector Constraints in PolySAT

Inequalities	$p \leq q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \,\&\, q$	
Structural	$r = p \ll q$, $r = p \gg q$	
Clauses	Disjunction of constraint literals	
By Reduction:		
Equations	$p = q \iff p - q \leq 0$	
Inequalities (signed)	$p \leq_s q \iff p + 2^{k-1} \leq q + 2^{k-1}$	

Bitvector Constraints in PolySAT

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Inequalities	$p \leq q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \,\&\, q$	
Structural	$r = p \ll q$, $r = p \gg q$	
Clauses	Disjunction of constraint literals	
By Reduction:		
Equations	$p = q \iff p - q \leq 0$	
Inequalities (signed)	$p \leq_s q \iff p + 2^{k-1} \leq q + 2^{k-1}$	
Bit-wise negation	$\sim p = -p - 1$	

Bitvector Constraints in PolySAT

Inequalities	$p \le q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \,\&\, q$	
Structural	$r = p \ll q$, $r = p \gg q$	
Clauses	Disjunction of constraint literals	
By Reduction:		
Equations	$p = q \iff p - q \le 0$	
Inequalities (signed)	$p \le_s q \iff p + 2^{k-1} \le q + 2^{k-1}$	
Bit-wise negation	$\sim p = -p - 1$	
Bit-wise or	$p \mid q = p + q - (p \,\&\, q)$	

Bitvector Constraints in PolySAT

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Inequalities	$p \leq q$	(polynomials p, q)
Overflow	$\Omega^*(p, q)$	
Bit-wise	$r = p \,\&\, q$	
Structural	$r = p \ll q,\ r = p \gg q$	
Clauses	Disjunction of constraint literals	
By Reduction:		
Equations	$p = q \iff p - q \leq 0$	
Inequalities (signed)	$p \leq_s q \iff p + 2^{k-1} \leq q + 2^{k-1}$	
Bit-wise negation	$\sim p = -p - 1$	
Bit-wise or	$p \mid q = p + q - (p \,\&\, q)$	
Quotient/remainder	$q \coloneqq \mathtt{bvudiv}(a, b),\ r \coloneqq \mathtt{bvurem}(a, b)$	
	▶ $a = bq + r$	
	▶ $\neg\Omega^*(b, q)$	
	▶ $\neg\Omega^+(bq, r)$	(e.g., $bq \leq -r - 1$)
	▶ $b \neq 0 \rightarrow r < b$	

PolySAT Solving Loop

Modified CDCL loop with theory assignments

- Assign boolean values to constraint literals ($p \leq q$ vs. $p > q$)
- Assign integer values to bitvector variables ($x \mapsto 3$)

PolySAT Solving Loop

Modified CDCL loop with theory assignments

- Assign boolean values to constraint literals ($p \leq q$ vs. $p > q$)
- Assign integer values to bitvector variables ($x \mapsto 3$)

Main components:

- Trail Γ records assignments and reasons
- For each variable x, keep track of viable values V_x
- Conflict $\mathcal{C}$: set of constraints that contradicts Γ
- Conflict analysis learn a new constraint to avoid the conflict in the future

Example: Polynomial Equations

$\mathbb{Z}/2^k\mathbb{Z}[X]$

$$
\begin{aligned}
C_1: \quad & x^2y + 3y + 7 = 0 \mod 2^4 \\
C_2: \quad & 2y + z + 8 = 0 \mod 2^4 \\
C_3: \quad & 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4
\end{aligned}
$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ decide x

Example: Polynomial Equations

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

$$C_1: \quad x^2y + 3y + 7 = 0 \mod 2^4$$
$$C_2: \quad 2y + z + 8 = 0 \mod 2^4$$
$$C_3: \quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Example: Polynomial Equations

$$C_1: \quad x^2y + 3y + 7 = 0 \mod 2^4$$
$$C_2: \quad 2y + z + 8 = 0 \mod 2^4$$
$$C_3: \quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1

   $\rightsquigarrow C_1|_\Gamma : 3y + 7 = 0 \quad \Rightarrow y = 3$

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Example: Polynomial Equations

$$C_1: \quad x^2y + 3y + 7 = 0 \mod 2^4$$
$$C_2: \quad 2y + z + 8 = 0 \mod 2^4$$
$$C_3: \quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1

   $\rightsquigarrow C_1|_\Gamma : 3y + 7 = 0 \quad \Rightarrow y = 3$
3. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1, x}, C_2]\!]$ — propagate y, add C_2

Example: Polynomial Equations

$$C_1: \quad x^2y + 3y + 7 = 0 \mod 2^4$$
$$C_2: \quad 2y + z + 8 = 0 \mod 2^4$$
$$C_3: \quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1
   $\rightsquigarrow C_1|_\Gamma : 3y + 7 = 0 \quad \Rightarrow y = 3$
3. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1,x}, C_2]\!]$ — propagate y, add C_2
   $\rightsquigarrow C_2|_\Gamma : z + 14 = 0 \quad \Rightarrow z = 2$

Example: Polynomial Equations

$\mathbb{Z}/2^k\mathbb{Z}[X]$

$$\begin{aligned} C_1: &\quad x^2y + 3y + 7 = 0 \mod 2^4 \\ C_2: &\quad 2y + z + 8 = 0 \mod 2^4 \\ C_3: &\quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4 \end{aligned}$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1

   $\rightsquigarrow C_1|_\Gamma : 3y + 7 = 0 \quad \Rightarrow y = 3$
3. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1,x}, C_2]\!]$ — propagate y, add C_2

   $\rightsquigarrow C_2|_\Gamma : z + 14 = 0 \quad \Rightarrow z = 2$
4. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1,x}, C_2, (z \mapsto 2)^{C_2,y}, C_3]\!]$ — propagate z, add C_3

Example: Polynomial Equations

$\mathbb{Z}/2^k\mathbb{Z}[X]$

$$
\begin{aligned}
C_1: &\quad x^2y + 3y + 7 = 0 \mod 2^4\\
C_2: &\quad 2y + z + 8 = 0 \mod 2^4\\
C_3: &\quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4
\end{aligned}
$$

1. $\Gamma = [\![(x \mapsto 0)^\delta]\!]$ — decide x
2. $\Gamma = [\![(x \mapsto 0)^\delta, C_1]\!]$ — add C_1

   $\rightsquigarrow C_1|_\Gamma : 3y + 7 = 0 \quad \Rightarrow y = 3$
3. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1,x}, C_2]\!]$ — propagate y, add C_2

   $\rightsquigarrow C_2|_\Gamma : z + 14 = 0 \quad \Rightarrow z = 2$
4. $\Gamma = [\![(x \mapsto 0)^\delta, C_1, (y \mapsto 3)^{C_1,x}, C_2, (z \mapsto 2)^{C_2,y}, C_3]\!]$ — propagate z, add C_3

   $\rightsquigarrow C_3|_\Gamma : 1 = 0$

   Conflict: $\mathcal{C} = \{C_3, x = 0, y = 3, z = 2\}$

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Example: Polynomial Equations (conflict)

$$\Gamma = [\![(x \mapsto 0)^{\delta}, C_1, (y \mapsto 3)^{C_1,x}, C_2, (z \mapsto 2)^{C_2,y}, C_3]\!]$$
$$\mathcal{C} = \{C_3, x = 0, y = 3, z = 2\}$$

Follow dependencies of $\mathcal{C}$ according to Γ:

$$\mathcal{C}' = \{C_3, x = 0, y = 3, C_2\}$$

Example: Polynomial Equations (conflict)

$$\Gamma = [\![(x \mapsto 0)^{\delta}, C_1, (y \mapsto 3)^{C_1,x}, C_2, (z \mapsto 2)^{C_2,y}, C_3]\!]$$
$$\mathcal{C} = \{C_3, x = 0, y = 3, z = 2\}$$

Follow dependencies of $\mathcal{C}$ according to Γ:

$$\mathcal{C}' = \{C_3, x = 0, y = 3, C_2\}$$

$$\begin{aligned} C_3 &: & 3x + 4yz + 2z^2 + 1 &= 0 & \\ C_2 &: & 2y + z + 8 &= 0 & \mid \cdot 2z \\ C_3 - 2z \cdot C_2 &: & 3x + 1 &= 0 & \end{aligned}$$

Example: Polynomial Equations (conflict)

$\mathbb{Z}/2^k\mathbb{Z}[X]$

$$\Gamma = [\![(x \mapsto 0)^{\delta}, C_1, (y \mapsto 3)^{C_1,x}, C_2, (z \mapsto 2)^{C_2,y}, C_3]\!]$$

$$\mathcal{C} = \{C_3, x = 0, y = 3, z = 2\}$$

Follow dependencies of $\mathcal{C}$ according to Γ:

$$\mathcal{C}' = \{C_3, x = 0, y = 3, C_2\}$$

$$\begin{aligned} C_3 &: \quad 3x + 4yz + 2z^2 + 1 = 0 \\ C_2 &: \quad 2y + z + 8 = 0 \qquad | \cdot 2z \\ C_3 - 2z \cdot C_2 &: \quad 3x + 1 = 0 \end{aligned}$$

Lemma:

$$C_3 \wedge C_2 \rightarrow 3x + 1 = 0$$

Example: Polynomial Equations

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Constraints:

$$
\begin{aligned}
C_1 &: & x^2y + 3y + 7 &= 0 \mod 2^4 \\
C_2 &: & 2y + z + 8 &= 0 \mod 2^4 \\
C_3 &: & 3x + 4yz + 2z^2 + 1 &= 0 \mod 2^4 \\
C_4 &: & 3x + 1 &= 0 \mod 2^4
\end{aligned}
$$

Example: Polynomial Equations

Constraints:

$$
\begin{aligned}
C_1: && x^2y + 3y + 7 = 0 &\mod 2^4\\
C_2: && 2y + z + 8 = 0 &\mod 2^4\\
C_3: && 3x + 4yz + 2z^2 + 1 = 0 &\mod 2^4\\
C_4: && 3x + 1 = 0 &\mod 2^4
\end{aligned}
$$

Continued:

5. $\Gamma = [\![C_4^{C_2,C_3}]\!]$ backjump, propagate lemma

Example: Polynomial Equations

$\mathbb{Z}/2^k\mathbb{Z}[X]$

Constraints:

$$
\begin{aligned}
C_1 &: & x^2y + 3y + 7 = 0 &\mod 2^4\\
C_2 &: & 2y + z + 8 = 0 &\mod 2^4\\
C_3 &: & 3x + 4yz + 2z^2 + 1 = 0 &\mod 2^4\\
C_4 &: & 3x + 1 = 0 &\mod 2^4
\end{aligned}
$$

Continued:

5. $\Gamma = [\![C_4^{C_2,C_3}]\!]$ — backjump, propagate lemma
6. $\Gamma = [\![C_4^{C_2,C_3}, (x \mapsto 5)^{C_4}, C_1]\!]$ — propagate x

Example: Polynomial Equations

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Constraints:

$$
\begin{aligned}
C_1: &\quad x^2y + 3y + 7 = 0 \mod 2^4\\
C_2: &\quad 2y + z + 8 = 0 \mod 2^4\\
C_3: &\quad 3x + 4yz + 2z^2 + 1 = 0 \mod 2^4\\
C_4: &\quad 3x + 1 = 0 \mod 2^4
\end{aligned}
$$

Continued:

5. $\Gamma = [\![C_4^{C_2,C_3}]\!]$ — backjump, propagate lemma
6. $\Gamma = [\![C_4^{C_2,C_3}, (x \mapsto 5)^{C_4}, C_1]\!]$ — propagate x

   $\rightsquigarrow C_1|_\Gamma : 12y + 7 = 0$

   Conflict due to parity!

Example: Polynomial Equations

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Constraints:

$$
\begin{aligned}
C_1 &: & x^2y + 3y + 7 = 0 &\mod 2^4 \\
C_2 &: & 2y + z + 8 = 0 &\mod 2^4 \\
C_3 &: & 3x + 4yz + 2z^2 + 1 = 0 &\mod 2^4 \\
C_4 &: & 3x + 1 = 0 &\mod 2^4
\end{aligned}
$$

Continued:

5. $\Gamma = [\![C_4^{C_2,C_3}]\!]$ backjump, propagate lemma
6. $\Gamma = [\![C_4^{C_2,C_3}, (x \mapsto 5)^{C_4}, C_1]\!]$ propagate x
   $\rightsquigarrow C_1|_\Gamma : 12y + 7 = 0$
   Conflict due to parity!
7. Unsatisfiable.

How to choose values?

For each variable x, keep track of viable values V_x:

- choose a value from V_x for decisions
- propagate $x \mapsto v$ when $V_x = \{v\}$ is a singleton set
- conflict if $V_x = \emptyset$

How to choose values?

For each variable x, keep track of viable values V_x:

- choose a value from V_x for decisions
- propagate $x \mapsto v$ when $V_x = \{v\}$ is a singleton set
- conflict if $V_x = \emptyset$
- whenever a constraint becomes "simple enough", use it to restrict V_x

How to choose values?

For each variable x, keep track of viable values V_x:

- choose a value from V_x for decisions
- propagate $x \mapsto v$ when $V_x = \{v\}$ is a singleton set
- conflict if $V_x = \emptyset$
- whenever a constraint becomes "simple enough",
  use it to restrict V_x

Currently:

- V_x represented as set of intervals
- when x appears only linearly, extract a forbidden interval **[Graham-Lengrand et al., IJCAR'20]**
- additionally, keep track of fixed bits of x **[Zeljić et al., SAT'16]**
- bit-blasting as fallback
  (only a single bitvector variable)

Intervals

We use half-open intervals:

- Usual notation $[\ell; u[$
- but wrap around if $\ell > u$

Intervals

We use half-open intervals:

- Usual notation $[\ell; u[$
- but wrap around if $\ell > u$

Examples mod 2^4:

$$[2;5[= \{2,3,4\}$$
$$[13;2[= \{13,14,15,0,1\}$$
$$[0;0[= \emptyset$$

Note:

$$p \in [\ell; u[\quad \iff \quad p - \ell < u - \ell$$

Forbidden Intervals

Forbidden interval of a constraint (example in $\mathbb{Z}/2^4\mathbb{Z}$):

- Current trail Γ contains $x_1 \mapsto 11$, $x_2 \mapsto 13$, and $x_3 \mapsto 9$.

Forbidden Intervals

Forbidden interval of a constraint (example in $\mathbb{Z}/2^4\mathbb{Z}$):

- Current trail Γ contains $x_1 \mapsto 11$, $x_2 \mapsto 13$, and $x_3 \mapsto 9$.
- Constraint C: $x_1 \leq x_1^2 x_3 + y$
  Note: only y is unassigned

Forbidden Intervals

Forbidden interval of a constraint (example in $\mathbb{Z}/2^4\mathbb{Z}$):

- Current trail Γ contains $x_1 \mapsto 11$, $x_2 \mapsto 13$, and $x_3 \mapsto 9$.
- Constraint $C\colon x_1 \leq x_1^2 x_3 + y$
  Note: only y is unassigned
- Substituting the assignment: $C|_\Gamma\colon 11 \leq 1 + y$

Forbidden Intervals

Forbidden interval of a constraint (example in $\mathbb{Z}/2^4\mathbb{Z}$):

- Current trail Γ contains $x_1 \mapsto 11$, $x_2 \mapsto 13$, and $x_3 \mapsto 9$.
- Constraint $C\colon x_1 \leq x_1^2 x_3 + y$
  Note: only y is unassigned
- Substituting the assignment: $C|_\Gamma\colon 11 \leq 1 + y$
- Thus $y \notin [15; 10[$
  $\leadsto$ use to restrict V_y

Forbidden Intervals

Forbidden interval of a constraint (example in $\mathbb{Z}/2^4\mathbb{Z}$):

- Current trail Γ contains $x_1 \mapsto 11$, $x_2 \mapsto 13$, and $x_3 \mapsto 9$.
- Constraint $C\colon x_1 \leq x_1^2 x_3 + y$
  Note: only y is unassigned
- Substituting the assignment: $C|_\Gamma\colon 11 \leq 1 + y$
- Thus $y \notin [15; 10[$
  $\rightsquigarrow$ use to restrict V_y
- Symbolic interval: $y \notin [-x_1^2 x_3; x_1 - x_1^2 x_3[$

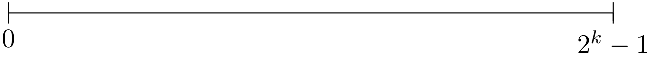
Forbidden Interval Lemma

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$

Forbidden Interval Lemma

$\mathbb{Z}/2^k\mathbb{Z}[X]$

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$
- Concrete intervals cover the domain: $\bigcup_i [\ell_i; u_i[= [0; 2^k[$



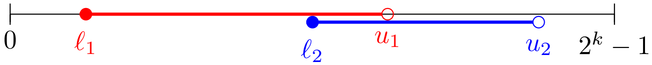
Forbidden Interval Lemma

$\mathbb{Z}/2^k\mathbb{Z}[X]$

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$
- Concrete intervals cover the domain: $\bigcup_i [\ell_i; u_i[= [0; 2^k[$

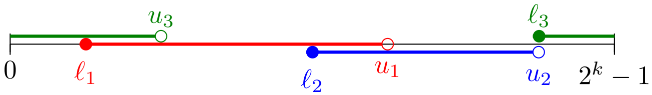
Forbidden Interval Lemma

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$
- Concrete intervals cover the domain: $\bigcup_i [\ell_i; u_i[= [0; 2^k[$



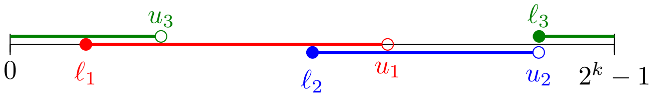
Forbidden Interval Lemma

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$
- Concrete intervals cover the domain: $\bigcup_i [\ell_i; u_i[= [0; 2^k[$



Forbidden Interval Lemma

- Forbidden intervals:
  $C_i \implies x \notin [\ell_i; u_i[$
- Concrete intervals cover the domain: $\bigcup_i [\ell_i; u_i[= [0; 2^k[$



- Use symbolic intervals to express the overlap condition:

$$u_1 \in [\ell_2; u_2[\wedge u_2 \in [\ell_3; u_3[\wedge u_3 \in [\ell_1; u_1[$$

Forbidden Intervals

p, q, r, s: polynomials, evaluable in current trail Γ

x: variable, unassigned in Γ

$$px + r \leq qx + s$$

Forbidden Intervals

p, q, r, s: polynomials, evaluable in current trail Γ
x: variable, unassigned in Γ

$$px + r \leq qx + s$$

[Graham-Lengrand et al., IJCAR'20]

p	q	Interval	
0	1	$x \notin [-s; r - s[$	if $r \neq 0$
1	0	$x \notin [s - r + 1; -r[$	if $s \neq -1$
1	1	$x \notin [-s; -r[$	if $r \neq s$

Forbidden Intervals

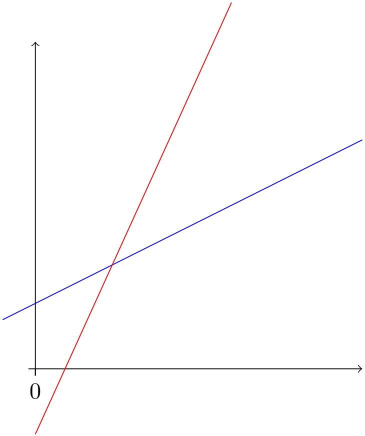
p, q, r, s: polynomials, evaluable in current trail Γ
x: variable, unassigned in Γ

$$px + r \leq qx + s$$

p	q	Lemmas from intervals
$\{0, n\}$	$\{0, n\}$	Set of intervals (“equal coeff.”)
n	m	Set of intervals (“disequal coeff.”)
		Intervals from fixed bits
		Fallback to bit-blasting

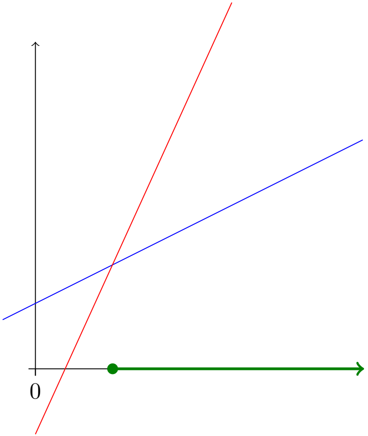
Forbidden Intervals (disequal coefficients)

$px + r \leq qx + s$ with $p \neq q$



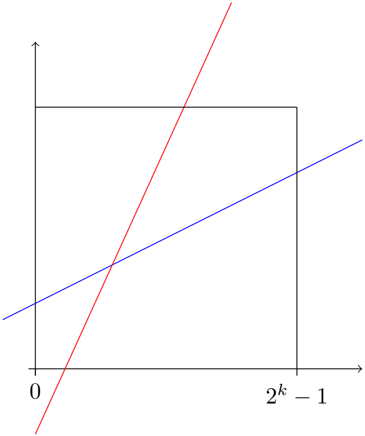
Forbidden Intervals (disequal coefficients)

$px + r \leq qx + s$ with $p \neq q$



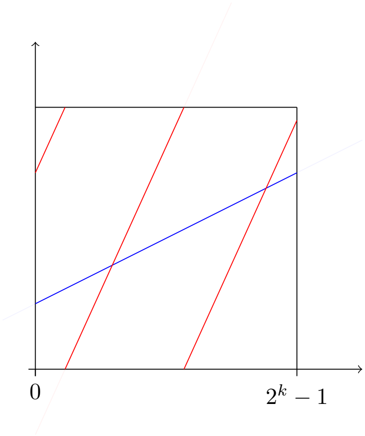
Forbidden Intervals (disequal coefficients)

$px + r \leq qx + s$ with $p \neq q$



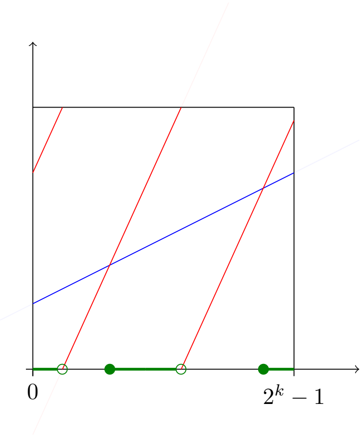
Forbidden Intervals (disequal coefficients)

$$px + r \leq qx + s \quad \text{with } p \neq q$$



Forbidden Intervals (disequal coefficients)

$$px + r \leq qx + s \quad \text{with } p \neq q$$



Conflict Resolution Strategy

1. Track the conflict's cone of influence while backtracking over the trail Γ
2. Conflict resolution plugins derive lemmas from constraints in the conflict
3. Accumulate lemmas from conflict plugins
   - New (often simpler) constraints improve propagation
   - Easy to experiment with new types of lemmas
4. When reaching the first relevant decision, learn lemmas and resume search

Conflict Resolution Plugins

Forbidden Intervals Lemma

Conflict Resolution Plugins

Forbidden Intervals Lemma

Superposition	$p(x) = 0 \wedge q(x) = 0$	$\implies rp(x) + sq(x) = 0$
	choose r, s to eliminate highest power of x	

Conflict Resolution Plugins

Forbidden Intervals Lemma		
Superposition	$p(x) = 0 \wedge q(x) = 0$	$\implies rp(x) + sq(x) = 0$
	choose r, s to eliminate highest power of x	
Var. Elim.	$px = q \wedge C[rx + s] \wedge \ \ldots$	$\implies C[p^{-1}q \cdot (r \gg n) + s]$
	pseudo-inverse: $p^{-1}p = 2^n$ for minimal n	

Conflict Resolution Plugins

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Forbidden Intervals Lemma

Superposition	$p(x) = 0 \wedge q(x) = 0$	$\Longrightarrow$	$rp(x) + sq(x) = 0$
	choose r, s to eliminate highest power of x		
Var. Elim.	$px = q \wedge C[rx + s] \wedge \dots$	$\Longrightarrow$	$C[p^{-1}q \cdot (r \gg n) + s]$
	pseudo-inverse: $p^{-1}p = 2^n$ for minimal n		
Bounds	$C(x, y) \wedge x \in [x_l; x_h]$	$\Longrightarrow$	$y \in [y_l; y_h]$
	$\Omega^*(p, q) \wedge p \leq b_1$	$\Longrightarrow$	$q \geq b_2$
	$axy + bx + cy + d \leq \dots$	$\Longrightarrow$	$\dots$

Conflict Resolution Plugins

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Forbidden Intervals Lemma			
Superposition	$p(x) = 0 \land q(x) = 0$	$\implies$	$rp(x) + sq(x) = 0$
	choose r, s to eliminate highest power of x		
Var. Elim.	$px = q \land C[rx + s] \land \ \ldots$	$\implies$	$C[p^{-1}q \cdot (r \gg n) + s]$
	pseudo-inverse: $p^{-1}p = 2^n$ for minimal n		
Bounds	$C(x, y) \land x \in [x_l; x_h]$	$\implies$	$y \in [y_l; y_h]$
	$\Omega^*(p, q) \land p \leq b_1$	$\implies$	$q \geq b_2$
	$axy + bx + cy + d \leq \ldots$	$\implies$	$\ldots$
Overflow	$\Omega^*(p, q) \land \neg\Omega^*(p, r)$	$\implies$	$q > r$

Conflict Resolution Plugins

$\boxed{\mathbb{Z}/2^k\mathbb{Z}[X]}$

Forbidden Intervals Lemma			
Superposition	$p(x) = 0 \wedge q(x) = 0$	$\Longrightarrow$	$rp(x) + sq(x) = 0$
	choose r, s to eliminate highest power of x		
Var. Elim.	$px = q \wedge C[rx + s] \wedge \ \dots$	$\Longrightarrow$	$C[p^{-1}q \cdot (r \gg n) + s]$
	pseudo-inverse: $p^{-1}p = 2^n$ for minimal n		
Bounds	$C(x, y) \wedge x \in [x_l; x_h]$	$\Longrightarrow$	$y \in [y_l; y_h]$
	$\Omega^*(p, q) \wedge p \leq b_1$	$\Longrightarrow$	$q \geq b_2$
	$axy + bx + cy + d \leq \dots$	$\Longrightarrow$	$\dots$
Overflow	$\Omega^*(p, q) \wedge \neg\Omega^*(p, r)$	$\Longrightarrow$	$q > r$
Bit-wise and	$x = p \,\&\, q$	$\Longrightarrow$	$x \leq p$
	$x = p \,\&\, q \wedge p = q$	$\Longrightarrow$	$x = p$
	$x = p \,\&\, q \wedge p = 2^n - 1$	$\Longrightarrow$	$2^{n-k}x = 2^{n-k}q$
...	...	...	

MCSAT based approaches for non-linear modular arithmetic

1. Constraints in $\mathbb{F}_q[X]$

- Finite field
- Not algebraically closed
- Constraints: $=, \neq$

Modulo 5

$$
\begin{aligned}
x^2 - 1 &= 0 \\
xy - y - 1 &= 0 \\
xy - 2 &\neq 0
\end{aligned}
$$

$\Rightarrow$ FFSat

2. Constraints in $\mathbb{Z}/2^k\mathbb{Z}[X]$

- Finite commutative ring
- Not algebraically closed
- Constraints: $=, \neq, <, >, \Omega^*(x, y)$

Modulo 2^4

$$
\begin{aligned}
xy + y &\leq y + 3 \\
2y + z &= 10 \\
3x + 6yz + 3z^2 &= 1
\end{aligned}
$$

$\Rightarrow$ PolySat