Relational Constraint Solving in SMT

Paul Meng, Andrew Reynolds, Cesare Tinelli, Clark Barrett

THE UNIVERSITY OF IOWA

Midwest Verification Day
September 2018

Relational Reasoning

Many problems can be modeled **relationally**

- Ontologies
- Network systems
- High-level system design
- …

Relational logic is well suited for reasoning about structurally rich problems

A Motivating Example

Modeling a Toy File System

There is a **root** directory

$Root \subseteq Dir$

The **contents** defines relations between directories and files or directories

$contents \subseteq Dir \times (File \cup Dir)$

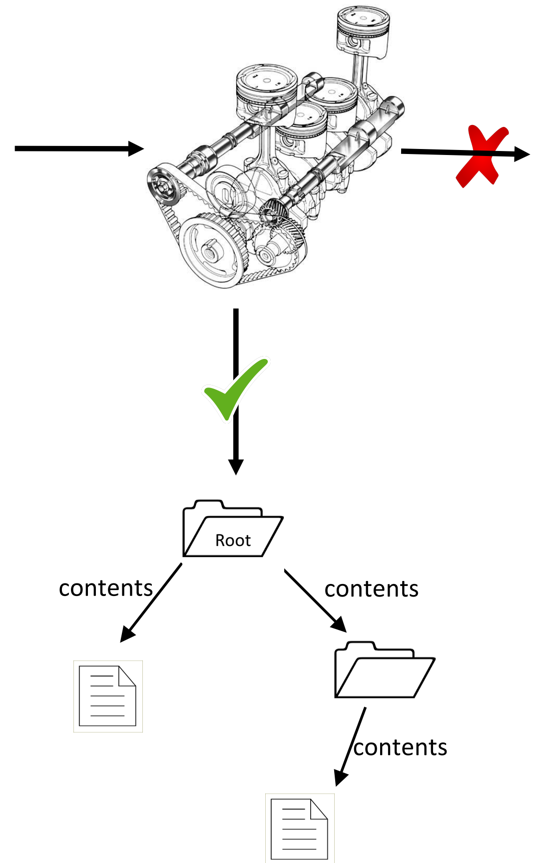
All **directories** and **files** are reachable from the **root** directory by following the **contents**

$(File \cup Dir) \subseteq$ Root.*contents

Contents relation is acyclic

$\forall\, d: Dir \mid \neg(d \subseteq d.$^contents$)$

A Relational Solver



* – reflexive-transitive closure ^ – transitive closure

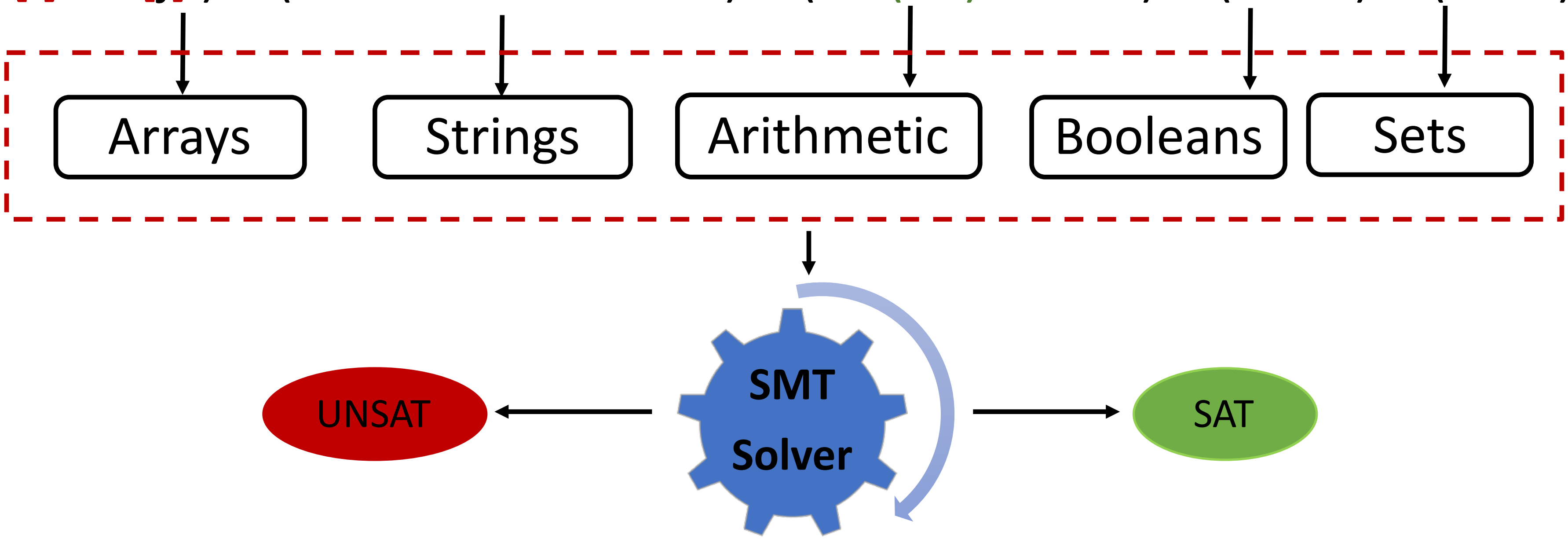
Technical Preliminaries

Satisfiability Modulo Theories (SMT)

Satisfiability Modulo Theories (SMT)

Decide the satisfiability of **many-sorted first-order logic** formulas with respect to combinations of background **theories**

$(a[i] > a[j]) \wedge (str = \text{“Hello World”}) \wedge (len(str) + x = 3) \wedge (A \vee B) \wedge (x \in S)$



Satisfiability Modulo Theories (SMT)

A **theory** $\boldsymbol{\mathcal{T}}$ = (Σ, $\mathbf{I}$) defines

- A **signature** Σ: a set of non-logical symbols
- A class of Σ**-interpretations** $\mathbf{I}$
- **Examples**: integer arithmetic, strings, finite sets, …
  - A simple **theory**: $\Sigma_s = \{0, 1, +, =\}$
  - A **formula** in the theory $\boldsymbol{\mathcal{T}}_s$: $x + 0 = 1$

Related Work

Alloy

A **declarative language** based on first-order **relational logic** created **at MIT**

Model and analyze **structurally-rich** systems

SAT-based analysis by the Alloy Analyzer

- **Checks the consistency** of an Alloy Specification
- Can **disprove but only prove a given property** for an Alloy specification **within a given bounds**

Analysis of Alloy Specifications via SMT

El Ghazi et al. [8, 9, 10] **translates** the Alloy kernel language to SMT-LIB language and **solves using SMT solvers** (**AlloyPE**)

The resulting SMT formulas are difficult to solve due to **heavy usage of quantifiers** in the translation

Description Logics (DLs)

Fragments of relational logic for efficient **knowledge representation and reasoning**

Consider on purpose **only unary and binary relations**

OWL: a standardized semantic **web ontology language** based on **description logics**

- **Efficient solvers**: KONCLUDE, HermiT, FaCT++ and etc.

A Theory of Finite Set $\mathcal{T}_\mathcal{S}$ in SMT

A theory $\mathcal{T}_\mathcal{S}$ of **finite sets** was introduced by Kshitij Bansal et al. [3]

Signature $\Sigma_\mathcal{S}$ of $\mathcal{T}_\mathcal{S}$

- **Singleton set constructor**: $[\_]:\ \alpha \to \mathrm{Set}(\alpha)$
- **Subset**: $\sqsubseteq\ : \mathrm{Set}(\alpha) \times \mathrm{Set}(\alpha) \to \mathrm{Bool}$
- **Membership**: $\in\ : \alpha \times \mathrm{Set}(\alpha) \to \mathrm{Bool}$
- **Union, intersection, set difference**:

$$\sqcap, \sqcup, \setminus\ : \mathrm{Set}(\alpha) \times \mathrm{Set}(\alpha) \to \mathrm{Set}(\alpha)$$

A modular **set solver** was implemented in **CVC4**

My Research

A Theory of Finite Relations $\mathcal{T}_{\mathcal{R}}$

Type Notations

$\mathbf{Tup}_{\boldsymbol{n}}(\boldsymbol{\alpha}_{\mathbf{1}}, \ldots, \boldsymbol{\alpha}_{\boldsymbol{n}})$**:** a parametric tuple sort (n > 0)

$\mathbf{Set}(\mathbf{Tup}_{\boldsymbol{n}}(\boldsymbol{\alpha}_{\mathbf{1}}, \ldots, \boldsymbol{\alpha}_{\boldsymbol{n}}))$**:** a relational sort abbreviated as $\mathbf{Rel}_{\boldsymbol{n}}(\boldsymbol{\alpha}_{\mathbf{1}}, \cdots, \boldsymbol{\alpha}_{\boldsymbol{n}})$

Relational Signature $\Sigma_{\mathcal{R}}$ of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{R}}}$

Tuple constructor:

$$\langle\, \_\, ,\ \ldots , \_\, \rangle : \alpha_1 \times \cdots \times \alpha_n \rightarrow \mathrm{Tup}_n(\alpha_1, \ldots, \alpha_n)$$

- **Example**: $\langle 1, 2 \rangle$ a binary integer tuple constant

Singleton relation constructor:

$$[\_] : \mathrm{Tup}_n(\alpha_1, \ldots, \alpha_n) \rightarrow \mathrm{Rel}_\mathrm{n}(\alpha_1, \ldots, \alpha_n)$$

- **Example**: $[\langle 1, \text{“Hello”} \rangle]$ a singleton set of integer and string binary tuple

Relational Signature $\Sigma_{\mathcal{R}}$ of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{R}}}$

Product: $* : \mathrm{Rel}_m(\alpha) \times \mathrm{Rel}_n(\beta) \to \mathrm{Rel}_{m+n}(\alpha, \beta)$

- **Example**: R1 = $[\langle 1, 2\rangle, \langle 3, 4\rangle]$; R2 = $[\langle 5, 6\rangle]$
  R1 $*$ R2 = $[\langle 1, 2, 5, 6\rangle, \langle 3, 4, 5, 6\rangle]$

Join: $\bowtie$: $\mathrm{Rel}_{p+1}(\alpha, \gamma) \times \mathrm{Rel}_{q+1}(\gamma, \beta) \to \mathrm{Rel}_{p+q}(\alpha, \beta)$
with $p + q > 0$

- **Example**: R1 = $[\langle 1, \text{“Hello”}\rangle, \langle 2, \text{“Hi”}\rangle]$;
  R2 = $[\langle \text{“Hello”}, 3\rangle, \langle \text{“World”}, 4\rangle]$;
  R1 $\bowtie$ R2 = $[\langle 1, 3\rangle]$

Relational Signature $\Sigma_{\mathcal{R}}$ of $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{R}}}$

Transpose: $\_^{-1}\colon \mathrm{Rel}_m(\alpha_1, \cdots, \alpha_m\,) \to \mathrm{Rel}_m(\alpha_m, \cdots, \alpha_1\,)$

- **Example**: R =$[\langle 1, \text{“Hello”}\rangle, \langle 2, \text{“Hi”}\rangle]$;
  R^{-1} = $[\langle \text{“Hello”}, 1\rangle, \langle \text{“Hi”}, 2\rangle]$;

Transitive Closure: $\_^{+}\colon \mathrm{Rel}_2(\alpha, \alpha) \to \mathrm{Rel}_2(\alpha, \alpha)$

- **Example**: R =$[\langle 1, 2\rangle, \langle 2, 3\rangle]$
  R^{+} =$[\langle 1, 2\rangle, \langle 2, 3\rangle, \langle 1, 3\rangle]$

A Calculus $\mathcal{C}_{\mathcal{R}}$ for $\mathcal{T}_{\mathcal{R}}$

A Compact Calculus for $\mathcal{T}_{\mathcal{S}}$

$$\textsc{Inter Up}\quad \frac{x \sqsubseteq s \in \mathcal{S}^* \qquad x \sqsubseteq t \in \mathcal{S}^* \qquad s \sqcap t \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ x \sqsubseteq s \sqcap t}$$

$$\textsc{Inter Down}\quad \frac{x \sqsubseteq s \sqcap t \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\ x \sqsubseteq s,\ x \sqsubseteq t}$$

$$\textsc{Union Up}\quad \frac{x \sqsubseteq u \in \mathcal{S}^* \qquad u \in \{s, t\} \qquad s \sqcup t \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ x \sqsubseteq s \sqcup t}$$

$$\textsc{Union Down}\quad \frac{x \sqsubseteq s \sqcup t \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\ x \sqsubseteq s \ \| \ \mathcal{S} := \mathcal{S},\ x \sqsubseteq t}$$

Derivation rules for intersection and union

A Compact Calculus for $\mathcal{T}_{\mathcal{S}}$

$$\textsc{Diff Up}\quad \frac{x \sqsubseteq s \in \mathcal{S}^* \qquad s \setminus t \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\, x \sqsubseteq t \;\;\|\;\; \mathcal{S} := \mathcal{S},\, x \sqsubseteq s \setminus t}$$

$$\textsc{Diff Down}\quad \frac{x \sqsubseteq s \setminus t \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\, x \sqsubseteq s,\, x \not\sqsubseteq t}$$

$$\textsc{Single Up}\quad \frac{[\,x\,] \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\, x \sqsubseteq [\,x\,]}$$

$$\textsc{Single Down}\quad \frac{x \sqsubseteq [\,y\,] \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\, x \approx y}$$

$$\textsc{Empty Unsat}\quad \frac{x \sqsubseteq [\,] \in \mathcal{S}^*}{\mathsf{unsat}}$$

$$\textsc{Set Diseq}\quad \frac{s \not\approx t \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\, z \sqsubseteq s,\, z \not\sqsubseteq t \quad \| \quad \mathcal{S} := \mathcal{S},\, z \not\sqsubseteq s,\, z \sqsubseteq t}$$

$$\textsc{Eq Unsat}\quad \frac{(t \not\approx t) \in \mathcal{S}^*}{\mathsf{unsat}}$$

Derivation rules for set difference, singleton, disequality and contradiction

TRANSPOSE Derivation Rule ($\_^{-1}$)

$$\textsc{Transp Up}\quad \frac{\langle x_1, \ldots, x_n \rangle \sqsubseteq R \in \mathcal{S}^* \qquad R^{-1} \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ \langle x_n, \ldots, x_1 \rangle \sqsubseteq R^{-1}}$$

$$\textsc{Transp Down}\quad \frac{\langle x_1, \ldots, x_n \rangle \sqsubseteq R^{-1} \in \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\ \langle x_n, \ldots, x_1 \rangle \sqsubseteq R}$$

JOIN Derivation Rule ($\bowtie$)

$$\textsc{Join Up}\quad \frac{\langle x_1, \ldots, x_m, z \rangle \sqsubseteq R_1, \langle z, y_1, \ldots, y_n \rangle \sqsubseteq R_2 \in \mathcal{S}^* \quad m + n > 0 \quad R_1 \bowtie R_2 \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ \langle x_1, \ldots, x_m, y_1, \ldots, y_n \rangle \sqsubseteq R_1 \bowtie R_2}$$

$$\textsc{Join Down}\quad \frac{\langle x_1, \ldots, x_m, y_1, \ldots, y_n \rangle \sqsubseteq R_1 \bowtie R_2 \in \mathcal{S}^* \quad \mathrm{ar}(R_1) = m + 1}{\mathcal{S} := \mathcal{S},\ \langle x_1, \ldots, x_m, z \rangle \sqsubseteq R_1,\ \langle z, y_1, \ldots, y_n \rangle \sqsubseteq R_2}$$

z is a fresh variable

PRODUCT Derivation Rule ($*$)

$$\textsc{Prod Up}\;\frac{\langle x_1, \ldots, x_m \rangle \sqsubseteq R_1 \in \mathcal{S}^* \quad \langle y_1, \ldots, y_n \rangle \sqsubseteq R_2 \in \mathcal{S}^* \quad R_1 * R_2 \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ \langle x_1, \ldots, x_m, y_1, \ldots, y_n \rangle \sqsubseteq R_1 * R_2}$$

$$\textsc{Prod Down}\;\frac{\langle x_1, \ldots, x_m, y_1, \ldots, y_n \rangle \sqsubseteq R_1 * R_2 \in \mathcal{S}^* \quad \mathrm{ar}(R_1) = m}{\mathcal{S} := \mathcal{S},\ \langle x_1, \ldots, x_m \rangle \sqsubseteq R_1,\ \langle y_1, \ldots, y_n \rangle \sqsubseteq R_2}$$

TRANSITIVE CLOSURE Derivation Rule ($\_^+$)

TClos Up I

$$\frac{\langle x_1, x_2 \rangle \sqsubseteq R \in \mathcal{S}^* \quad R^+ \in \mathcal{T}(\mathcal{S})}{\mathcal{S} := \mathcal{S},\ \langle x_1, x_2 \rangle \sqsubseteq R^+}$$

TClos Up II

$$\frac{\langle x_1, x_2 \rangle \sqsubseteq R^+,\ \langle x_2, x_3 \rangle \sqsubseteq R^+ \in\ \mathcal{S}^*}{\mathcal{S} := \mathcal{S},\ \langle x_1, x_3 \rangle \sqsubseteq R^+}$$

TClos Down

$$\frac{\langle x_1, x_2 \rangle \sqsubseteq R^+ \in \mathcal{S}^*}{\begin{array}{l} \mathcal{S} := \mathcal{S},\ \langle x_1, x_2 \rangle \sqsubseteq R \quad \| \quad \mathcal{S} := \mathcal{S},\ \langle x_1, z \rangle \sqsubseteq R,\ \langle z, x_2 \rangle \sqsubseteq R \\ \| \quad \mathcal{S} := \mathcal{S},\ \langle x_1, z_1 \rangle \sqsubseteq R,\ \langle z_1, z_2 \rangle \sqsubseteq R^+,\ \langle z_2, x_2 \rangle \sqsubseteq R,\ z_1 \not\approx z_2 \end{array}}$$

z, z_1, z_2 are fresh variables

An Example

$$\mathcal{S} = \{\boxed{\langle a,b\rangle \in R^+}, \langle a,b\rangle \notin \mathrm{R}, \langle a,b\rangle \notin \mathrm{R} \bowtie \mathrm{R}\}$$

TCLOS DOWN

$$\mathcal{S} := \mathcal{S} \cup \{\langle a,b\rangle \in \mathrm{R}\}$$

EQ UNSAT $\langle a,b\rangle \notin \mathrm{R}$

UNSAT

$$\mathcal{S} := \mathcal{S} \cup \{\langle a,k\rangle \in \mathrm{R}, \langle k\ b\rangle \in \mathrm{R}\}$$

JOIN UP

$$\mathcal{S} := \mathcal{S} \cup \{\langle a,b\rangle \in \mathrm{R} \bowtie \mathrm{R}\}$$

$\langle a,b\rangle \notin \mathrm{R} \bowtie \mathrm{R}$ **EQ UNSAT**

UNSAT

$$\mathcal{S} := \mathcal{S} \cup \{\langle a,k_1\rangle \in \mathrm{R}, \langle k_1,k_2\rangle \in \mathrm{R}, \langle k_2\ b\rangle \in \mathrm{R}, k_1 \not\approx k_2\}$$

NO RULES APPLY

(After exhaustively applying JOIN-UP)

SAT

Calculus $\mathcal{C}_{\mathcal{R}}$ Correctness

Calculus $\mathcal{C}_{\mathcal{R}}$ Correctness

Refutation Sound – a closed derivation tree proves that input constraints are **UNSAT**

Model Sound – from a saturated branch of a derivation tree one can extract a model for input constraints

Detailed proof can be found in Meng et al. [21]

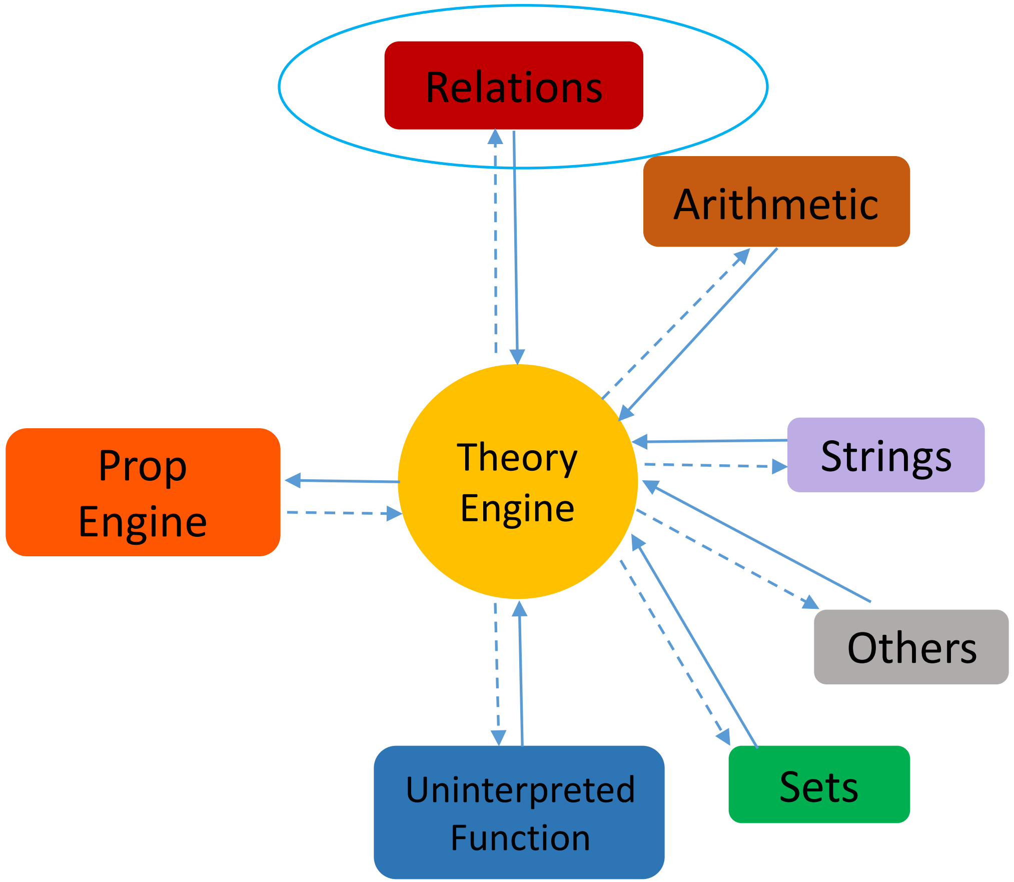
Termination for a Fragment of $\mathcal{T}_{\mathcal{R}}$

(element)	$e := x$
(unary relation)	$u := x \mid [\,] \mid u_1 \sqcup u_2 \mid u_1 \sqcap u_2 \mid [\,\langle e \rangle\,] \mid b \bowtie u$
(binary relation)	$b := x \mid [\,] \mid b_1 \sqcup b_2 \mid b_1 \sqcap b_2 \mid [\,\langle e_1, e_2 \rangle\,] \mid b^{-1}$
(constraint)	$\varphi := e_1 \approx e_2 \mid \langle e \rangle \sqsubseteq u \mid \langle e_1, e_2 \rangle \sqsubseteq b \mid \neg \varphi_1$

Termination: If S is a finite set of constraints generated by the grammar above, then all derivation trees are finite.

Detailed proof can be found in Meng et al. [21]

A Relational Solver in CVC4



A Relational Solver in CVC4

- Allows us to solve constraints from a combination of relations and other domains
- Extend SMT-LIB/CVC4 native language with support for relations
- Enables natural mappings from several relational modeling languages to SMT
- Brings to those languages the power of SMT solvers and their ability to reason efficiently about built-in types

Applications of $\mathcal{T}_{\mathcal{R}}$

Application 1: Alloy to CVC4

Support Alloy kernel language in **SMT natively**

Finite model finding of CVC4 can efficiently reason about problems with presence of quantifiers

Built a **translator** from Alloy kernel language to SMT

Can **disprove and prove properties** with respect to Alloy specifications

ALLOY KERNEL LANGUAGE	CVC4
Signature **sig** S	$S : Rel_1(Atom)$
Field $f : S_1 \to \cdots \to S_n$ of a sig S	$f : Rel_{n+1}(Atom, \ldots, Atom)$ $f \sqsubseteq S * S_1 * \cdots * S_n$
sig $S_1, \ldots, S_n$ **extends** S	$S_1 \sqsubseteq S, \ldots, S_n \sqsubseteq S$ $S_i \sqcap S_j = [\,]$ for $1 \le i < j \le n$ $S_1 \sqcup \cdots \sqcup S_n = S$ if S is abstract
sig $S_1, \ldots, S_n$ **in** S,	$S_1 \sqsubseteq S, \ldots, S_n \sqsubseteq S$

ALLOY KERNEL LANGUAGE	CVC4
Sets Operators: $+$, $\&$, $-$, $=$, in	$\sqcup$,$\sqcap$ $-$, $\approx$,$\sqsubseteq$
Relational Operators: ~, $\cdot$, $\rightarrow$, ^	$\_^{-1}$, $\bowtie$, $*$, $\_^{+}$
Logical operators: and, or, not	AND, OR, NOT
Quantifiers: all, some	FORALL, EXISTS

Evaluation on Alloy Benchmarks

Evaluated CVC4 with two configurations

- **CVC4**: enables full native support for relational operators
- **CVC4+AX:** encodes all relational operators as uninterpreted functions with axioms

Compared with **Alloy Analyzer** and **AlloyPE** on two sets of benchmarks:

1. From AlloyPE and
2. From an academic course

Evaluation on Alloy Benchmarks

Compared to the **Alloy Analyzer**

- **CVC4** is **overall slower for SAT benchmarks**
- **CVC4** **solves UNSAT benchmarks**, whereas the Alloy Analyzer can only answer bounded UNSAT

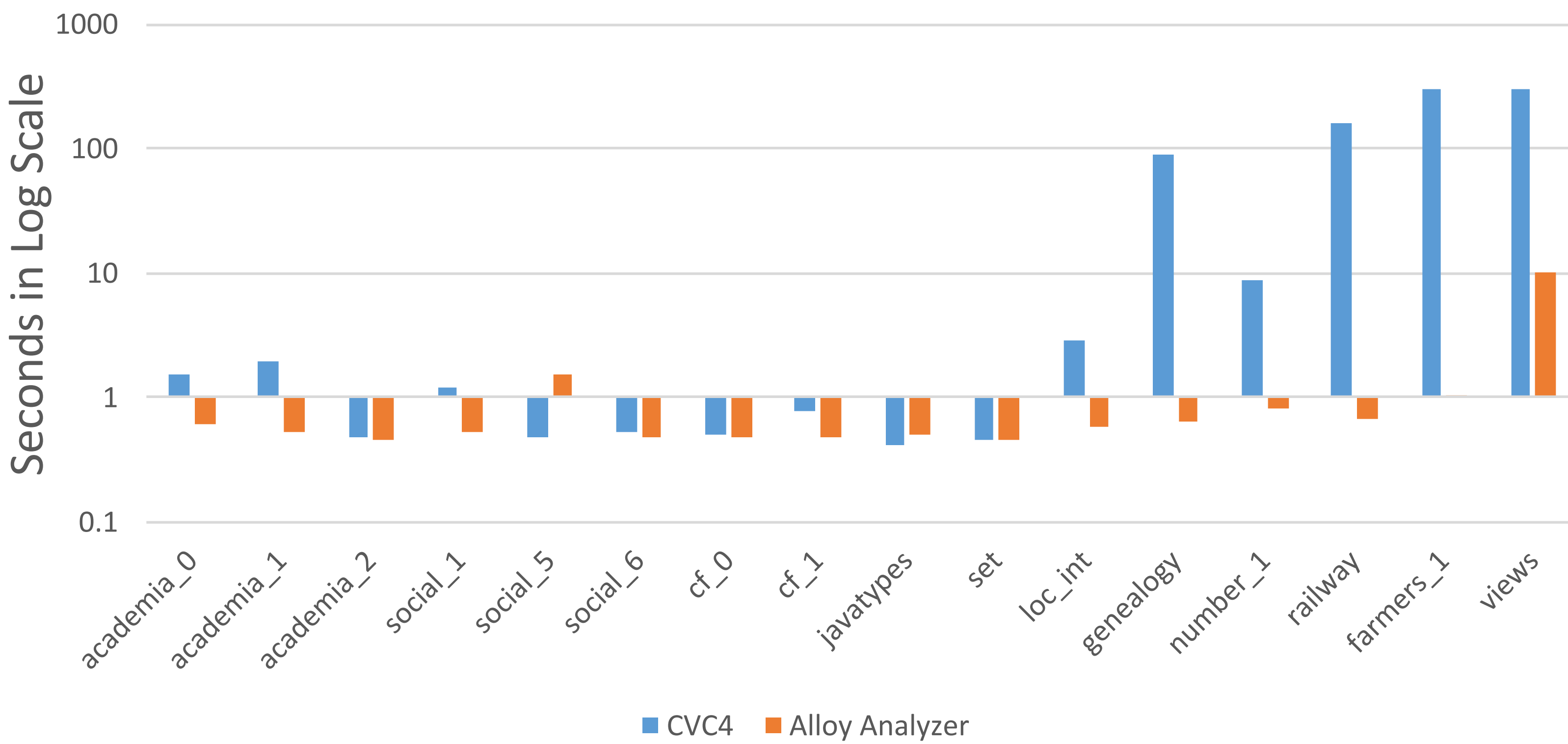
Compared to **AlloyPE**

- **CVC4** **solves SAT benchmarks**, whereas AlloyPE solves none
- **CVC4** **solves most of AlloyPE's benchmarks**

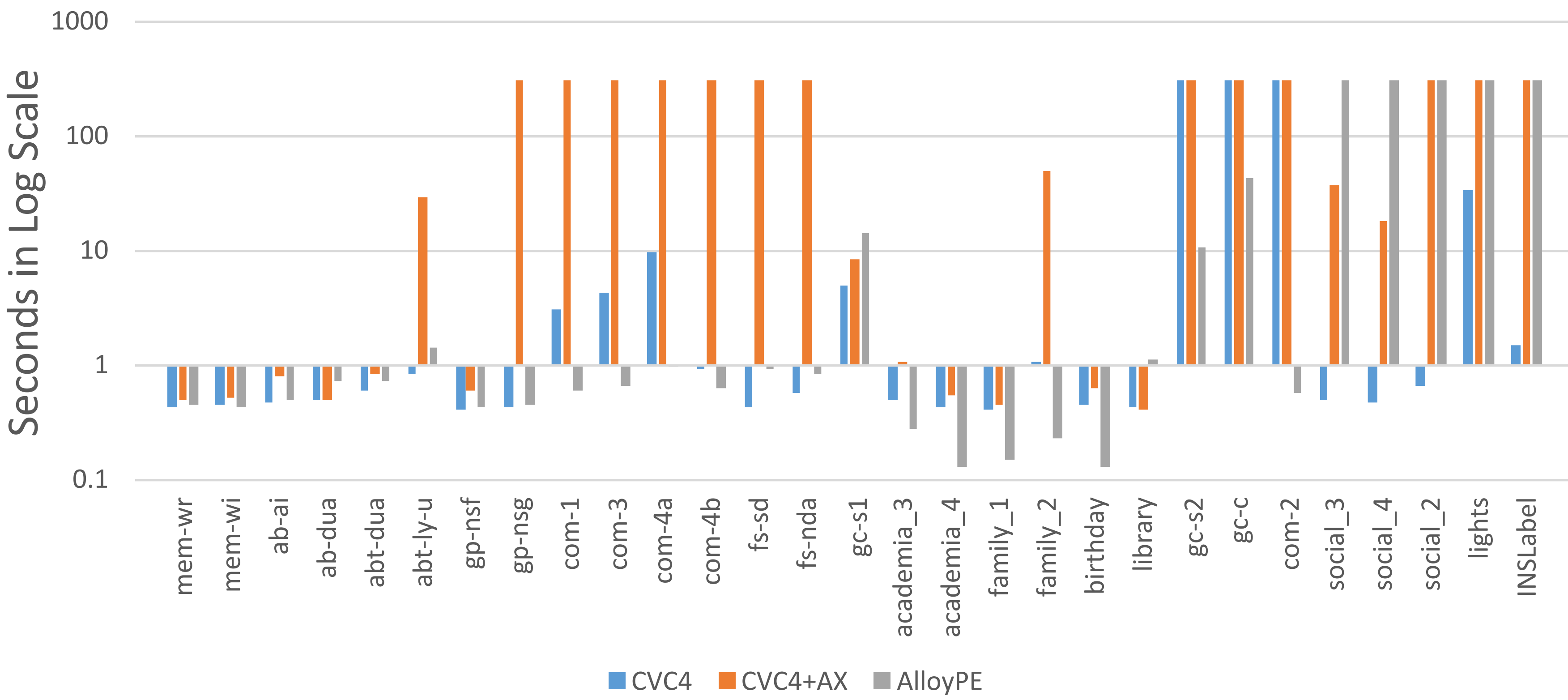
Compared to **CVC4+AX**

- **CVC4** **solves SAT benchmarks**, whereas CVC4+AX solves none
- **CVC4** **solves significantly more UNSAT benchmarks**

Evaluation on SAT Benchmarks



Evaluation on UNSAT Benchmarks



Application 2: OWL DL to SMT

OWL DL based on an **expressive** description logic fragment

Built a **translator** from **OWL DL** to **SMT** in $\mathcal{T}_{\mathcal{R}}$

Check **logical consistency** of OWL models using CVC4

OWL DL	CVC4
Individual name a	a : Atom
Nominal {a}	{<a>}
Top concept T Bottom concept ⊥	Univ, {∀ a : Atom | <a> ∈ Univ} [ ]
Atomic concept C Role R	C : Rel_1(Atom) R : Rel_2(Atom, Atom)
Union C ⊔ D Intersection C ⊓ D	C ⊔ D C ⊓ D
Inverse role R^{-} Complement ¬C	R^{-1} Univ \ C

OWL DL	CVC4
Concept, role assertion $C(a)$, $R(a; b)$	$a \in C$, $<a, b> \in R$
Individual (dis)equality $a \approx b$, $a \not\approx b$	$a \approx b$, $a \not\approx b$
Concept, role inclusion $C \sqsubseteq D$, $R \sqsubseteq S$	$C \sqsubseteq D$, $R \sqsubseteq S$
Concept, role equiv. $C \equiv D$, $R \equiv S$	$C \approx D$, $R \approx S$
Complex role inclusion $R_1 \circ R_2 \sqsubseteq S$	$R_1 \bowtie R_2 \sqsubseteq S$
Role disjointness Disjoint(R, S)	$R \sqcap S \approx [\ ]$

OWL DL	CVC4
Existential restriction $\exists R.C$	$R \bowtie C$
Universal restriction $\forall R.C$	$[\, x \mid x \in Univ \wedge [x] \bowtie R \sqsubseteq C \,]$
At-least restriction $\geq_{n} R.C$	$[\, x \mid x \in Univ \wedge (\exists\, a_1, \ldots, a_n : Atom$ $[\langle a_1 \rangle, \ldots, \langle a_n \rangle] \sqsubseteq (([x] \bowtie R) \sqcap C)$ $\wedge\, Dist(a_1, \ldots, a_n))]$
At-most restriction $\leq_{n} R.C$	$[\, x \mid x \in Univ \wedge (\exists\, a_1, \ldots, a_n : Atom$ $(([x] \bowtie R) \sqcap C) \sqsubseteq [\langle a_1 \rangle, \ldots, \langle a_n \rangle]$ $\wedge\, [\langle a_1 \rangle, \ldots, \langle a_n \rangle] \sqsubseteq C)]$
Local reflexivity $\exists R.Self$	$[\langle x, y \rangle \mid \langle x, y \rangle \in R\ x \approx y]$

Evaluation on OWL Benchmarks

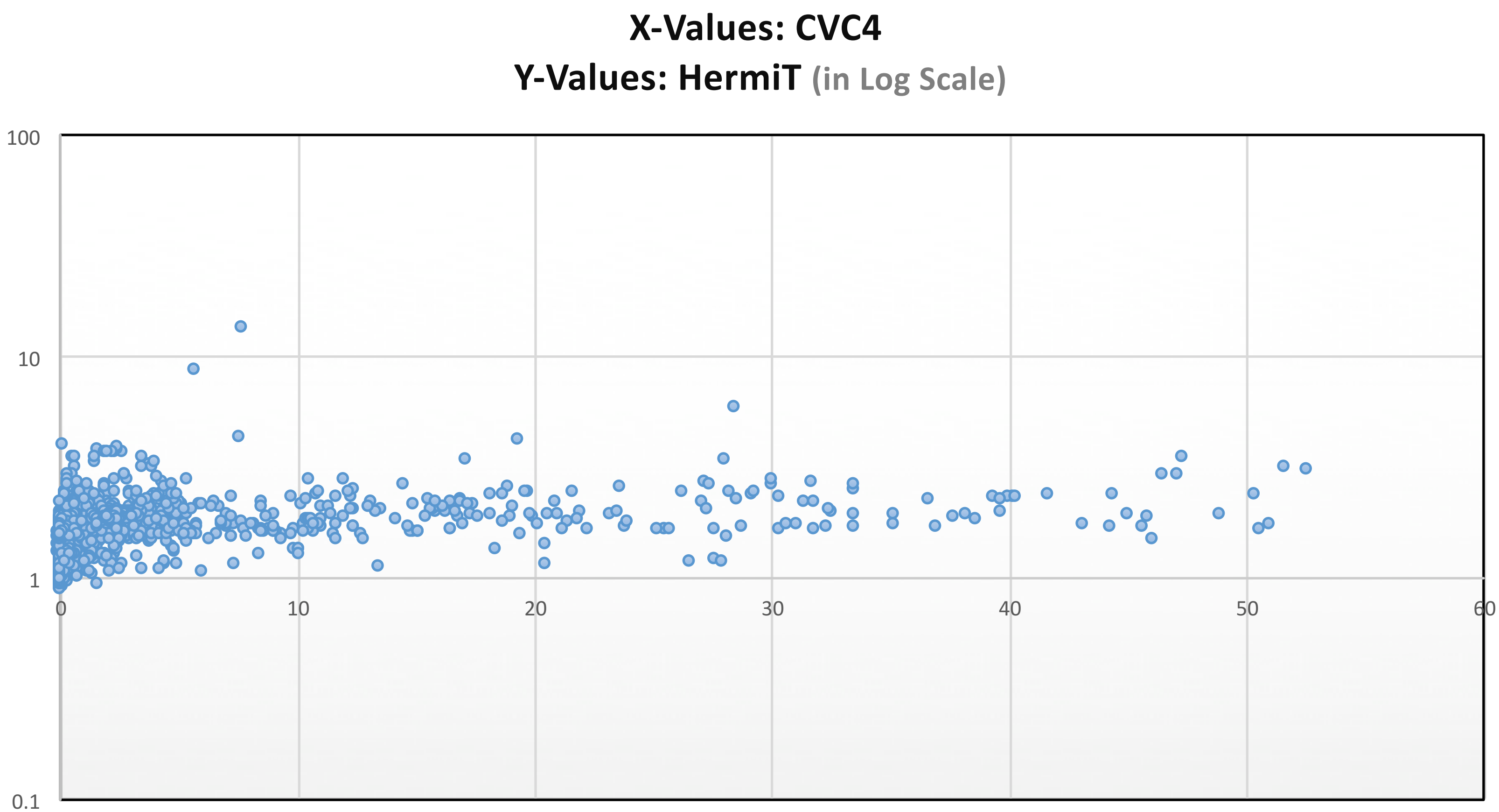
Evaluated on **OWL models** from 4th OWL Reasoner Evaluation competition

Compared with a state of the art **DL reasoner HermiT**

For the ones (4269) we both solved:

- **CVC4** takes **2.62s** per benchmark and solves faster on **1617** benchmarks
- **HermiT** takes **1.76s** per benchmark and solves faster on **2652** benchmarks

Comparison with HermiT



Summary

- Introduced **a theory of finite relations in SMT**
- Developed **a refutation-sound and model-sound calculus** for the theory of relations
- Demonstrated useful **applications** in Alloy and OWL
- Shown **promising experimental results** on Alloy and OWL benchmarks

Thank you!

References

1. F. Baader. The description logic handbook: Theory, implementation and applications. Cambridge university press, 2003.

2. F. Baader, I. Horrocks, and U. Sattler. Description logics. In V. L. Frank van Harmelen and B. Porter, editors, Handbook of Knowledge Representation, volume 3 of Foundations of Artificial Intelligence, pages 135 – 179. Elsevier, 2008.

3. K. Bansal, A. Reynolds, C. W. Barrett, and C. Tinelli. A new decision procedure for finite sets and cardinality constraints in SMT. In Proceedings of IJCAR'16, volume 9706 of LNCS, pages 82–98. Springer, 2016.

4. C. Barrett, C. L. Conway, M. Deters, L. Hadarean, D. Jovanovi´c, T. King, A. Reynolds, and C. Tinelli. CVC4. In Proceedings of CAV'11, volume 6806 of LNCS, pages 171–177. Springer, 2011.

5. C. Barrett, P. Fontaine, and C. Tinelli. The SMT-LIB standard—Version 2.6. In A. Gupta and D. Kroening, editors, SMT 2010, 2010.

References

6. C. Barrett, R. Sebastiani, S. Seshia, and C. Tinelli. Satisfiability modulo theories. In A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, editors, Handbook of Satisfiability, volume 185, chapter 26, pages 825–885. IOS Press, February 2009.

7. B. Dutertre and L. D. Moura. The YICES SMT solver. Technical report, SRI International, 2006.

8. A. A. E. Ghazi and M. Taghdiri. Analyzing alloy constraints using an SMT solver: a case study. In 5th International Workshop on Automated Formal Methods (AFM), 2010.

9. A. A. E. Ghazi and M. Taghdiri. Relational reasoning via SMT solving. In Proceedings of FM'11, volume 6664 of LNCS, pages 133–148. Springer, 2011.

10. A. A. E. Ghazi, M. Taghdiri, and M. Herda. First-order transitive closure axiomatization via iterative invariant injections. In Proceedings of NFM'15, volume 9058 of LNCS. Springer, 2015.

References

11. I. Horrocks and U. Sattler. Decidability of shiq with complex role inclusion axioms. Artificial Intelligence, 160(1-2):79–104, 2004.

12. D. Jackson. Alloy: a lightweight object modelling notation. ACM Trans. Softw. Eng. Methodol., 11(2):256–290, 2002.

13. D. Jackson. Software Abstractions - Logic, Language, and Analysis. MIT Press, 2006.

14. R. Nieuwenhuis, A. Oliveras, and C. Tinelli. Solving SAT and SAT Modulo Theories: from an abstract Davis-Putnam-Logemann-Loveland Procedure to DPLL(T). Journal of the ACM, 53(6):937–977, Nov. 2006.

15. A. Reynolds, C. Tinelli, A. Goel, and S. Krstic. Finite model finding in SMT. In Proceedings of CAV’13, volume 8044 of LNCS, pages 640–655. Springer, 2013.

References

16. A. Steigmiller, T. Liebig, and B. Glimm. Konclude: System description. Web Semantics: Science, Services and Agents on the World Wide Web, 27(1), 2014.

17. E. Torlak and D. Jackson. Kodkod: a relational model finder. In Proceedings of TACAS’07, volume 4424 of LNCS, pages 632–647. Springer, 2007.

18. D. Tsarkov and I. Horrocks. Fact++ description logic reasoner: system description. In Proceedings of IJCAR’06, volume 4130 of LNCS. Springer, 2006.

19. D. Tsarkov and I. Palmisano. Chainsaw: a metareasoner for large ontologies. In I. Horrocks, M. Yatskevich, and E. Jim´enez-Ruiz, editors, ORE, 2012.

20. W3C. OWL 2 web ontology language, https://www.w3.org/2007/OWL/wiki/Syntax.

21. Baoluo Meng, Andrew Reynolds, Cesare Tinelli, and Clark Barrett. Relational Constraint Solving in SMT. In Proceedings of the 26th International Conference on Automated Deduction, Gothenburg, Sweden

A Toy File System Specification in Alloy

```
abstract sig FSO {}
sig File extends FSO {}
sig Dir extends FSO {
    contents: Set FSO
}
-- contents relation is acyclic
fact {all d: Dir | not (d in d.^contents)}
-- Every file system object only has one location
assert oneLocation {
  all o : FSO | lone d : FSO | o in d.contents
}
```

check oneLocation for **7**

An Example

$$\mathcal{S} = \{\langle a,b\rangle \notin \mathrm{R}^{-1}, \mathrm{R} \approx \mathrm{Q}, \boxed{\langle a\rangle \in \mathrm{P}, \langle b\rangle \in \mathrm{P}, \mathrm{P} * \mathrm{P}} \approx \mathrm{Q} \sqcap \mathrm{T}\}$$

PROD UP

$$\mathcal{S} := \mathcal{S} \cup \{\boxed{\langle a,b\rangle \in \mathrm{P} * \mathrm{P}, \langle b,a\rangle \in \mathrm{P} * \mathrm{P}, \langle a,a\rangle \in \mathrm{P} * \mathrm{P}, \ldots}\}$$

$\mathrm{P} * \mathrm{P} \approx \mathrm{Q} \sqcap \mathrm{T}$ **INTER DOWN**

$$\mathcal{S} := \mathcal{S} \cup \{\langle a,b\rangle \in \mathrm{Q}, \boxed{\langle b,a\rangle \in \mathrm{Q}}, \langle a,a\rangle \in \mathrm{Q}, \langle b,b\rangle \in \mathrm{Q}, \ldots\}$$

$\langle a,b\rangle \notin \mathrm{R}^{-1}, \mathrm{R} \approx \mathrm{Q}$ **TRANS UP**

$$\mathcal{S} := \mathcal{S} \cup \{\boxed{\langle a,b\rangle \in \mathrm{R}^{-1}}, \ldots\} \xrightarrow[\langle a,b\rangle \notin \mathrm{R}^{-1}]{\textbf{EQ UNSAT}} \textbf{UNSAT}$$

Satisfiability Modulo Theories (SMT)

A **theory** $\boldsymbol{\mathcal{T}}$ = (Σ, $\mathbf{I}$) defines

- A **signature** Σ: a set of non-logical symbols
- A class of Σ**-interpretations** $\mathbf{I}$
- **Examples**: integer arithmetic, strings, finite sets, …
  - ➢ A simple **theory**: $\Sigma_s = \{0, 1, +, =\}$
  - ➢ A **formula** in the theory $\boldsymbol{\mathcal{T}}_s$: $x + 0 = 1$