Formal Methods in Software Engineering

Satisfiability Modulo Theories — Spring 2025

Konstantin Chukharev

§1 First-Order Theories

Motivation

Consider the signature $\Sigma = \langle \Sigma^S, \Sigma^F \rangle$ for a fragment of number theory:

- $\Sigma^S = \{\texttt{Nat}\}$, $\Sigma^F = \{0, 1, +, <\}$
- $\text{rank}(0) = \text{rank}(1) = \langle \texttt{Nat} \rangle$
- $\text{rank}(+) = \langle \texttt{Nat}, \texttt{Nat}, \texttt{Nat} \rangle$
- $\text{rank}(<) = \langle \texttt{Nat}, \texttt{Nat}, \texttt{Bool} \rangle$

1. Consider the Σ-sentence: $\forall x : \texttt{Nat}.\, \neg(x < x)$
   - Is it *valid*, that is, true under *all* interpretations?
   - No, e.g., if we interpret $<$ as *equals* or *divides*.

2. Consider the Σ-sentence: $\neg\exists x : \texttt{Nat}.\, (x < 0)$
   - Is it *valid*?
   - No, e.g., if we interpret Nat as the set of *all* integers.

3. Consider the Σ-sentence: $\forall x : \texttt{Nat}. \forall y : \texttt{Nat}. \forall z : \texttt{Nat}.\, (x < y) \land (y < z) \to (x < z)$
   - Is it *valid*?
   - No, e.g., if we interpret $<$ as the *successor* relation.

Motivation [2]

In practice, we often *do not care* about satisfiability or validity in *general*, but rather with respect to a *limited class* of interpretations.

A practical reason:

- When reasoning in a particular application domain, we typically have *specific* data types/structures in mind (e.g., integers, strings, lists, arrays, finite sets, …).
- More generally, we are typically *not* interested in *arbitrary* interpretations, but rather in *specific* ones.

Theories formalize this domain-specific reasoning: we talk about satisfiability and validity *with respect to a theory* or "*modulo a theory*".

A computational reason:

- The validity problem for FOL is *undecidable* in general.
- However, the validity problem for many *restricted* theories, is *decidable*.

First-Order Theories

Hereinafter, we assume that we have an infinite set of variables X.

Definition 1 (Theory): A first-order *theory* $\mathcal{T}$ is a pair $\langle \Sigma, M \rangle$, where

- $\Sigma = \langle \Sigma^S, \Sigma^F \rangle$ is a first-order signature,
- M is a class of Σ-interpretations over X that is *closed under variable re-assignment*.

Definition 2: M is *closed under variable re-assignment* if every Σ-interpretation that differs from one in M in the way it interprets the variables in X is also in M.

A theory limits the interpretations of Σ-formulas to those from M.

Theory Examples

Example: Theory of Real Arithmetic $\mathcal{T}_{\mathrm{RA}} = \langle \Sigma_{\mathrm{RA}}, M_{\mathrm{RA}} \rangle$:

- $\Sigma^S_{\mathrm{RA}} = \{\texttt{Real}\}$
- $\Sigma^F_{\mathrm{RA}} = \{+, -, *, \leq\} \cup \{q \mid q \text{ is a decimal numeral}\}$
- All $\mathcal{I} \in M_{\mathrm{RA}}$ interpret `Real` as the set of *real numbers* $\mathbb{R}$, each q as the *decimal number* that it denotes, and the function symbols in the usual way.

Example: Theory of Ternary Strings $\mathcal{T}_{\mathrm{TS}} = \langle \Sigma_{\mathrm{TS}}, M_{\mathrm{TS}} \rangle$:

- $\Sigma^S_{\mathrm{TS}} = \{\texttt{String}\}$
- $\Sigma^F_{\mathrm{TS}} = \{\,\cdot\,, <\} \cup \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$
- All $\mathcal{I} \in M_{\mathrm{TS}}$ interpret `String` as the set $\{\mathrm{a}, \mathrm{b}, \mathrm{c}\}^*$ of all finite strings over the characters {“a”, “b”, “c”}, symbol $\cdot$ as string concatenation (e.g., $\mathrm{a} \cdot \mathrm{b} = \mathrm{ab}$), and $<$ as lexicographic order.

$\mathcal{T}$-interpretations

Definition 3 (Reduct): Let Σ and Ω be two signatures over variables X, where $\Omega \supseteq \Sigma$, that is, $\Omega^S \supseteq \Sigma^S$ and $\Omega^F \supseteq \Sigma^F$.

Let $\mathcal{I}$ be an Ω-interpretation over X.

The *reduct* $\mathcal{I}^\Sigma$ of $\mathcal{I}$ to Σ is a Σ-interpretation obtained from $\mathcal{I}$ by resticting it to the symbols in Σ.

Definition 4 ($\mathcal{T}$-interpretation): Given a theory $\mathcal{T} = \langle \Sigma, M \rangle$, a *$\mathcal{T}$-interpretation* is any Ω-interpretation $\mathcal{I}$ for some signature $\Omega \supseteq \Sigma$ such that $\mathcal{I}^\Sigma \in M$.

Note: This definition allows us to consider the satisfiability in a theory $\mathcal{T} = \langle \Sigma, M \rangle$ of formulas that contain sorts or function symbols not in Σ. These symbols are usually called *uninterpreted* (in $\mathcal{T}$).

$\mathcal{T}$-interpretations [2]

Example: Consider again the theory of real arithmetic $\mathcal{T}_{\text{RA}} = \langle \Sigma_{\text{RA}}, M_{\text{RA}} \rangle$.

All $\mathcal{I} \in M_{\text{RA}}$ interpret `Real` as $\mathbb{R}$ and function symbols as usual.

Which of the following interpretations are $\mathcal{T}_{\text{RA}}$-interpretations?

1. $\texttt{Real}^{\mathcal{I}_1} = \mathbb{Q}$, symbols in Σ^F_{RA} interpreted as usual. ✗
2. $\texttt{Real}^{\mathcal{I}_2} = \mathbb{R}$, symbols in Σ^F_{RA} interpreted as usual, and $\texttt{String}^{\mathcal{I}_2} = \{0.5, 1.3\}$. ✓
3. $\texttt{Real}^{\mathcal{I}_3} = \mathbb{R}$, symbols in Σ^F_{RA} interpreted as usual, and $\log^{\mathcal{I}_3}$ is the successor function. ✓

$\mathcal{T}$-satisfiability, $\mathcal{T}$-entailment, $\mathcal{T}$-validity

Definition 5 ($\mathcal{T}$-satisfiability): A Σ-formula α is *satisfiable in $\mathcal{T}$*, or *$\mathcal{T}$-satisfiable*, if it is satisfied by *some* $\mathcal{T}$-interpretation $\mathcal{I}$.

Definition 6 ($\mathcal{T}$-entailment): A set Γ of formulas *$\mathcal{T}$-entails* a formula α, if every $\mathcal{T}$-interpretation that satisfies all formulas in Γ also satisfies α.

Definition 7 ($\mathcal{T}$-validity): A formula α is *$\mathcal{T}$-valid*, if it is satisfied by *all* $\mathcal{T}$-interpretations.

Note: A formula α is *$\mathcal{T}$-valid* iff $\emptyset \vDash \alpha$.

Example: Which of the following Σ_{RA}-formulas is satisfiable or valid in $\mathcal{T}_{\mathrm{RA}}$?

1. $(x_0 + x_1 \leq 0.5) \land (x_0 - x_1 \leq 2)$ — *satisfiable*, *falsifiable*
2. $\forall x_0 . (x_0 + x_1 \leq 1.7) \to (x_1 \leq 1.7 - x_0)$ — *satisfiable*, *valid*
3. $\forall x_0 . \forall x_1 . (x_0 + x_1 \leq 1)$ — *unsatisfiable*, *falsifiable*

FOL vs Theory

For every signature Σ, entailment and validity in "pure" FOL can be seen as entailment and validity in the theory $\mathcal{T}_{\mathrm{FOL}} = \langle \Sigma, M_{\mathrm{FOL}} \rangle$ where M_{FOL} is the class of *all possible* Σ-interpretations.

- Pure first-order logic = reasoning over *all* possible interpretations.
- Reasoning modulo a theory = *restricting* interpretations with some domain constraints.
- Theories make automated reasoning *feasible* in many domains.

Axiomatization

Definition 8 (Axiomatic theory): A first-order *axiomatic theory* $\mathcal{T}$ is defined by a signature Σ and a set $\mathcal{A}$ of Σ-sentences, or *axioms*.

In particular, an Ω-formula α is *valid* in an axiomatic theory $\mathcal{T}$ if it is entailed by the axioms of $\mathcal{T}$, that is, every Ω-interpretation $\mathcal{J}$ that satisfies all axioms of $\mathcal{T}$ also satisfies α.

Given an axiomatic theory $\mathcal{T}$ defined by Σ and $\mathcal{A}$, we can define a theory $\mathcal{T}' = \langle \Sigma, M \rangle$ where M is the class of all Σ-interpretations that satisfy all axioms in $\mathcal{A}$.

It is not hard to show that a formula α is valid in $\mathcal{T}$ *iff* it is valid in $\mathcal{T}'$.

Axiomatization [2]

Note: Not all theories are first-order axiomatizable.

Example: Consider the theory $\mathcal{T}_{\mathtt{Nat}}$ of the natural numbers, with signature Σ with $\Sigma^S = \{\mathtt{Nat}\}$, $\Sigma^F = \{0, S, +, <\}$, and $M = \{\mathcal{I}\}$ where $\mathtt{Nat}^{\mathcal{I}} = \mathbb{N}$ and Σ^F is interpreted as usual.

Any set of axioms for this theory is satisfied by *non-standard models*, e.g., interpretations $\mathcal{I}$ where $\mathtt{Nat}^{\mathcal{I}}$ includes other chains of elements besides the natural numbers, e.g., $\mathbb{N}^{\mathcal{I}} = \{0, 1, 2, \ldots\} \cup \{\omega, \omega + 1, \ldots\}$.

These models *falsify* formulas that are *valid* in $\mathcal{T}_{\mathtt{Nat}}$, e.g., $\neg \exists x.\, (x < 0)$ or $\forall x.\, ((x \doteq 0) \vee \exists y.\, (x \doteq S(y)))$.

Completeness of Theories

Definition 9: A Σ-theory $\mathcal{T}$ is *complete* if for every Σ-sentence α, either α or $\neg\alpha$ is valid in $\mathcal{T}$.

Note: In a complete Σ-theory, every Σ-sentence is either valid or unsatisfiable.

Example: Any theory $\mathcal{T} = \langle \Sigma, M \rangle$ where all interpretations in M only differ in how they interpret the variables (e.g., $\mathcal{T}_{\text{RA}}$) is *complete*.

Completeness of Theories [2]

Example: The axiomatic (mono-sorted) theory of *monoids* with $\Sigma^F = \{\cdot, \varepsilon\}$ and axioms

$$\forall x. \forall y. \forall z. \, (x \cdot y) \cdot z \doteq x \cdot (y \cdot z) \qquad \forall x. \, (x \cdot \varepsilon \doteq x) \qquad \forall x. \, (\varepsilon \cdot x \doteq x)$$

is *incomplete*.

For example, the sentence $\forall x. \forall y. \, (x \cdot y \doteq y \cdot x)$ is `true` in some monoids (e.g. the integers with addition) but `false` in others (e.g. the strings with concatenation).

Completeness of Theories [3]

Example: The axiomatic (mono-sorted) theory of *dense linear orders without endpoints* with $\Sigma^F = \{\prec\}$ and axioms

$$\begin{array}{rl} \forall x.\forall y.(x \prec y) \to \exists z.\,((x \prec z) \wedge (z \prec y)) & \text{(dense)} \\ \forall x.\forall y.\,((x \prec y) \vee (y \prec x) \vee (x \doteq y)) & \text{(linear)} \\ \forall x.\,\neg(x \prec x) \quad \forall x.\forall y.\forall z.\,((x \prec y) \wedge (y \prec z) \to (x \prec z)) & \text{(orders)} \\ \forall x.\exists y.\,(y \prec x) \quad \forall x.\exists y.\,(x \prec y) & \text{(without endpoints)} \end{array}$$

is *complete*.

Decidability

Recall that a set A is *decidable* if there exists a *terminating* procedure that, given an input element a, returns (after *finite* time) either "yes" if $a \in A$ or "no" if $a \notin A$.

Definition 10: A theory $\mathcal{T} = \langle \Sigma, M \rangle$ is *decidable* if the set of all *$\mathcal{T}$-valid* Σ-formulas is decidable.

Definition 11: A *fragment* of $\mathcal{T}$ is a *syntactically-restricted subset* of $\mathcal{T}$-valid Σ-formulas.

Example: The *quantifier-free* fragment of $\mathcal{T}$ is the set of all $\mathcal{T}$-valid Σ-formulas without any quantifiers.

Example: The *linear* fragment of $\mathcal{T}_{\mathrm{RA}}$ is the set of all $\mathcal{T}$-valid Σ_{RA}-formulas without multiplication ($*$).

Axiomatizability

Definition 12: A theory $\mathcal{T} = \langle \Sigma, M \rangle$ is *recursively axiomatizable* if M is the class of all interpretations satisfying a *decidable set* of first-order axioms $\mathcal{A}$.

Theorem 1 (Lemma): Every recursively axiomatizable theory $\mathcal{T}$ admits a a procedure $E_{\mathcal{T}}$ that *enumerates* all $\mathcal{T}$-valid formulas.

Theorem 2: For every *complete* and *recursively axiomatizable* theory $\mathcal{T}$, $\mathcal{T}$-validity is decidable.

Proof: Given a formula α, use $E_{\mathcal{T}}$ to enumerate all valid formulas. Since $\mathcal{T}$ is complete, either α or $\neg\alpha$ will eventually (after *finite* time) be produced by $E_{\mathcal{T}}$. $\square$

§2 Introduction to SMT

Common Theories in SMT

SMT traditionally focuses on theories with *decidable* quantifier-free *fragments*.

Recall: a formula α is $\mathcal{T}$*-valid* iff $\neg\alpha$ is $\mathcal{T}$*-unsatisfiable*.

Checking the (un)satisfiability of quantifier-free formulas in main background theories efficiently has a large number of applications in:

- hardware and software verification
- model checking
- symbolic execution
- compiler validation
- type checking
- planning and scheduling
- software synthesis
- cyber-security
- verifiable machine learning
- analysis of biological systems

Further, we are going to study:

- A few of those theories and their decision procedures.
- Proof systems to reason modulo theories automatically.

From QF to Cubes

The satisfiability of quantifier-free formulas in a theory $\mathcal{T}$ is decidable iff the satisfiability in $\mathcal{T}$ of *conjunctions of literals (cubes)* is decidable.

We are going to study a general extension of DPLL to SMT that uses decision procedures for *conjunctions of literals*. Thus, we will mostly focus on *conjunctions of literals*.

Theory of Uninterpreted Functions

Given a signature Σ, the most general theory consists of the class of *all* Σ-interpretations.

In fact, this is a *family* of theories parameterized by the signature Σ.

It is known as the theory of *equality with uninterpreted functions* $\mathcal{T}_{\text{EUF}}$, or the *empty theory*, since it is axiomatized by the empty set of axioms.

Validity, and so satisfiability, in $\mathcal{T}_{\text{EUF}}$ is only *semi-decidable* (this is just a validity in FOL).

However, the satisfiability of *conjunctions $\mathcal{T}_{\text{EUF}}$-literals* is *decidable*, in polynomial time, using the *congruence closure* algorithm.

Example: $(a \doteq b) \land (f(a) \doteq b) \land \neg(g(a) \doteq g(f(a)))$ Is this formula satisfiable in $\mathcal{T}_{\text{EUF}}$?

Theory of Real Arithmetic

The theory of real arithmetic $\mathcal{T}_{\mathrm{RA}}$ is a theory of inequalities over the real numbers.

- $\Sigma^S = \{\texttt{Real}\}$
- $\Sigma^F = \{+, -, *, <\} \cup \{q \mid q \text{ is a decimal numeral}\}$
- M is the class of interpretations that interpret `Real` as the set of *real numbers* $\mathbb{R}$, and the function symbols in the usual way.

Satisfiability in the full $\mathcal{T}_{\mathrm{RA}}$ is *decidable* (in worst-case doubly-exponential time).

Restricted fragments of $\mathcal{T}_{\mathrm{RA}}$ can be decided more efficiently.

Example: Quantifier-free linear real arithmetic (`QF_LRA`) is the theory of *linear* inequalities over the reals, where $*$ can only be used in the form of *multiplication by constants (decimal numerals)*.

The satisfiability of conjunctions of literals in `QF_LRA` is *decidable* in *polynomial time*.

Theory of Integer Arithmetic

The theory of integer arithmetic $\mathcal{T}_{\text{IA}}$ is a theory of inequalities over the integers.

- $\Sigma^S = \{\texttt{Int}\}$
- $\Sigma^F = \{+, -, *, <\} \cup \{n \mid n \text{ is an integer numeral}\}$
- M is the class of interpretations that interpret `Int` as the set of *integers* $\mathbb{Z}$, and the function symbols in the usual way.

Satisfiability in $\mathcal{T}_{\text{IA}}$ is *not even semi-decidable*!

Satisfiability of quantifier-free Σ-formulas in $\mathcal{T}_{\text{IA}}$ is *undecidable* as well.

Linear integer arithmetic (`LIA`, also known as *Presburger arithmetic*) is decidable, but not efficiently (in worst-case triply-exponential time).

Theory of Arrays with Extensionality

The theory of arrays $\mathcal{T}_\mathrm{A}$ is useful for modelling RAM or array data structures.

- $\Sigma^S = \{\mathtt{A}, \mathtt{I}, \mathtt{E}\}$ (arrays, indices, elements)
- $\Sigma^F = \{\mathrm{read}, \mathrm{write}\}$, where $\mathrm{rank}(\mathrm{read}) = \langle \mathtt{A}, \mathtt{I}, \mathtt{E} \rangle$ and $\mathrm{rank}(\mathrm{write}) = \langle \mathtt{A}, \mathtt{I}, \mathtt{E}, \mathtt{A} \rangle$

Let a be a variable of sort $\mathtt{A}$, variable i of sort $\mathtt{I}$, and variable v of sort $\mathtt{E}$.

- $\mathrm{read}(a, i)$ denotes the value stored in array a at index i.
- $\mathrm{write}(a, i, v)$ denotes the array that stores value v at index i and is otherwise identical to a.

Example: $\mathrm{read}(\mathrm{write}(a, i, v), i) \doteq_{\mathtt{E}} v$

- Is this formula *intuitively* valid/satisfiable/unsatisfiable in $\mathcal{T}_\mathrm{A}$?

Example: $\forall i.\ (\mathrm{read}(a, i) \doteq_{\mathtt{E}} \mathrm{read}(a', i)) \rightarrow (a \doteq_{\mathtt{A}} a')$

- Is this formula *intuitively* valid/satisfiable/unsatisfiable in $\mathcal{T}_\mathrm{A}$?

Theory of Arrays with Extensionality [2]

The theory of arrays $\mathcal{T}_{\mathrm{A}} = \langle \Sigma, M \rangle$ is finitely axiomatizable.

M is the class of interpretations that satisfy the following axioms:

1. $\forall a. \forall i. \forall v. \, (\mathrm{read}(\mathrm{write}(a, i, v), i) \doteq_{\mathsf{E}} v)$
2. $\forall a. \forall i. \forall j. \forall v. \, \neg(i \doteq_{\mathsf{I}} j) \rightarrow (\mathrm{read}(\mathrm{write}(a, i, v), j) \doteq_{\mathsf{E}} \mathrm{read}(a, j))$
3. $\forall a. \forall b. \, (\forall i. \, (\mathrm{read}(a, i) \doteq_{\mathsf{E}} \mathrm{read}(b, i))) \rightarrow (a \doteq_{\mathsf{A}} b)$

Note: The last axiom is called *extensionality* axiom. It states that two arrays are equal if they have the same values at all indices. It can be omitted to obtain a theory of arrays *without extensionality*.

Satisfiability in $\mathcal{T}_{\mathrm{A}}$ is *undecidable*.

There are several *decidable fragments* of $\mathcal{T}_{\mathrm{A}}$.

§3 Extra slides

Decidability and Complexity

Theory	Description	Full	QF	Full complexity	QFC complexity
PL	Propositional logic	yes	yes	NP-complete	$\Theta(n)$
$\mathcal{T}_{\mathrm{E}}$	equality	no	yes	undecidable	$\mathcal{O}(n \log n)$
$\mathcal{T}_{\mathrm{PA}}$	Peano Arithmetic	no	no	undecidable	undecidable
$\mathcal{T}_{\mathbb{N}}$	Presburger Arithmetic	yes	yes	$\Omega\left(2^{2^n}\right)$, $\mathcal{O}\left(2^{2^{2^{kn}}}\right)$	NP-complete
$\mathcal{T}_{\mathbb{Z}}$	linear integers	yes	yes	$\Omega\left(2^{2^n}\right)$, $\mathcal{O}\left(2^{2^{2^{kn}}}\right)$	NP-complete
$\mathcal{T}_{\mathbb{R}}$	reals (with $\cdot$)	yes	yes	$\mathcal{O}\left(2^{2^{kn}}\right)$	$\mathcal{O}\left(2^{2^{kn}}\right)$
$\mathcal{T}_{\mathbb{Q}}$	rationals (without $\cdot$)	yes	yes	$\Omega(2^n)$, $\mathcal{O}\left(2^{2^{kn}}\right)$	PTIME
$\mathcal{T}_{\mathrm{RDS}}$	recursive data structures	no	yes	undecidable	$\mathcal{O}(n \log n)$
$\mathcal{T}^{+}_{\mathrm{RDS}}$	acyclic recursive data structures	yes	yes	not elementary recursive	$\Theta(n)$
$\mathcal{T}_{\mathrm{A}}$	arrays	no	yes	undecidable	NP-complete
$\mathcal{T}^{=}_{\mathrm{A}}$	arrays with extensionality	no	yes	undecidable	NP-complete

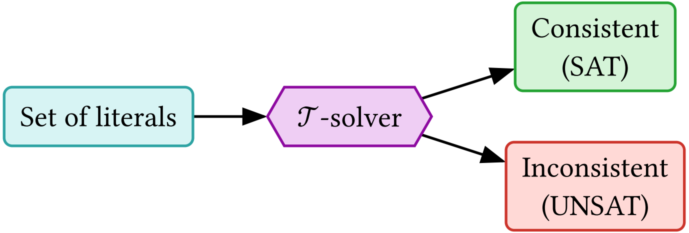
Decidability and Complexity [2]

- **"Full"** denotes the decidability of a complete theory *with* quantifiers.
- **"QF"** denotes the decidability of a *quantifier-free* theory.
- **"Full complexity"** denotes the complexity of the satisfiability in a complete theory *with* quantifiers.
- **"QFC complexity"** denotes the complexity of the satisfiability in a quantifier-free *conjunctive* fragment of a theory.
- "*Not elementary recursive*" means the runtime cannot be bounded by a fixed-height stack of exponentials.

§4 Theory Solvers

Theory Solvers

Definition 13 ($\mathcal{T}$-solver): A *theory solver*, or *$\mathcal{T}$-solver*, is a specialized decision procedure for the satisfiability of conjunctions of literals in a theory $\mathcal{T}$.



Difference Logic

Definition 14: *Difference logic* is a fragment of linear integer arithmetic consisting of conjunctions of literals of the very restricted form:

$$x - y \bowtie c$$

where x and y are integer variables, c is a numeral, and $\bowtie \in \{=, <, \leq, >, \geq\}$.

A solver for difference logic consists of three steps:

1. Literals normalization.
2. Conversion to a graph.
3. Cycle detection.

Difference Logic [2]

Step 1: Rewrite each literal using $\leq$ by applying the following rules:

1. $(x - y = c) \longrightarrow (x - y \leq c) \wedge (x - y \geq c)$
2. $(x - y \geq c) \longrightarrow (y - x \leq -c)$
3. $(x - y > c) \longrightarrow (y - x < -c)$
4. $(x - y < c) \longrightarrow (x - y \leq c - 1)$

Step 2: Construct a weighted directed graph G with a vertex for each variable and an edge $x \xrightarrow{c} y$ for each literal $(x - y \leq c)$.

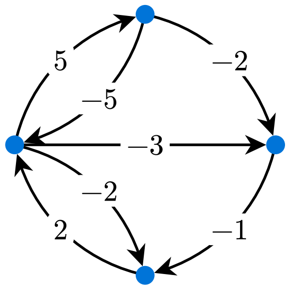
Step 3: Check for *negative cycles* in G.

- Use, for example, the Bellman-Ford algorithm.
- If G contains a negative cycle, the set of literals is *inconsistent* (UNSAT).
- Otherwise, the set of literals is *consistent* (SAT).

Difference Logic Example

$$(x - y = 5) \land (z - y \geq 2) \land (z - x > 2) \land (w - x = 2) \land (z - w < 0)$$

$$\begin{aligned}
(x - y = 5) &\longrightarrow (x - y \leq 5) \land (y - x \leq -5) \\
(z - y \geq 2) &\longrightarrow y - z \leq -2 \\
(z - x > 2) &\longrightarrow x - z \leq -3 \\
(w - x = 2) &\longrightarrow (w - x \leq 2) \land (x - w \leq -2) \\
(z - w < 0) &\longrightarrow z - w \leq -1
\end{aligned}$$



UNSAT because of the negative cycle: $-3, -1, 2$.