Seminar Decision Procedures and Applications

Background Informations

Viorica Sofronie-Stokkermans
University Koblenz-Landau

19 May 2015
Topics for the talks

- **Julia Schönberger**: Automata approach to Presburger arithmetic (in chapter by Hubert Common and Claude Kirchner)

- **David Nagel**: An efficient decision procedure for Unit Two-Variables per Inequality (UTVPI)Constraints (Work by Shuvendu Lahiri and Madanlal Musuvathi)

- **David Friedrich**: Abstract Congruence Closure (work by Leo Bachmair and Ashish Tiwari)

- **Cynthia Engel**: Shostak’s method (Work by R.E. Shostak and Harald Ganzinger)

- **Shuyi Weng**: What’s decidable about arrays (Work by A. Bradley, Z. Manna and H. Sipma)

- **Kilian Laudt**: Decision procedures for recursive data structures with integer constraints (Work by T. Zhang, H.B. Sipma, Z. Manna)

- **Carl Brenk**: ? Terminological cycles in a description logic with existential restriction (Work by Franz Baader) ?
Overview

We give a survey of decidability results in various theories.

- **Reasoning in standard theories**
- **Reasoning in complex theories**
  - theory extensions and combinations
- **Examples of applications**

**Important**: identify decidable tractable fragments
Reasoning about standard datatypes

• **Numbers**  - natural numbers, integers, reals, rationals

• **Data structures**  - theories of lists
  - theory of acyclic lists
  - theory of arrays
  - theories of sets, multisets
Reasoning in theory extensions

- **Numbers**
  - integers, reals, rationals

- **Data structures**
  - theories of lists
  - theory of acyclic lists
  - theory of arrays
  - theories of sets
    + functions (free, rec. def.) e.g.: length, card

- **Functions**
  - e.g. over numeric domains
  - monotone functions
  - bounds on slope
  - convexity/concavity conditions
  - continuity, derivability
Modularity

Modular (i.e. black-box) composition of decision procedures is highly desirable – for saving time and resources.

Example:

$T_1$, $T_0$, $T_2$:

$T_0$: $\Sigma_0$-theory.

$T_i$: $\Sigma_i$-theory;

$T_0 \subseteq T_i$; $\Sigma_0 \subseteq \Sigma_i$.

Can we use provers for $T_1$, $T_2$ as blackboxes to prove theorems in $T_1 \cup T_2$?

Which information needs to be exchanged between the provers?
Structure

Reasoning in standard theories

A crash course: Decidable logical theories and theory fragments

Reasoning in complex theories

Modular reasoning in combinations of theories
  Disjoint signature: The Nelson-Oppen method
  Hierarchical/Modular reasoning in theory extensions
    Local theory extensions: in one of the talks

Applications in the talks
In what follows we will use the following conventions:

**constants** (0-ary function symbols) are denoted with $a, b, c, d, ...$

**function symbols** with arity $\geq 1$ are denoted
- $f, g, h, ...$ if the formulae are interpreted into arbitrary algebras
- $+, -, s, ...$ if the intended interpretation is into numerical domains

**predicate symbols** with arity 0 are denoted $p, q, r, s, ...$

**predicate symbols** with arity $\geq 1$ are denoted
- $P, Q, R, ...$ if the formulae are interpreted into arbitrary algebras
- $\leq, \geq, <, >$ if the intended interpretation is into numerical domains

**variables** are denoted $x, y, z, ...$
Logical theories

Syntactic view
Axiomatized by a set $\mathcal{F}$ of (closed) first-order $\Sigma$-formulae.
the models of $\mathcal{F}$: $\text{Mod}(\mathcal{F}) = \{ \mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F} \}$

Semantic view
given a class $\mathcal{M}$ of $\Sigma$-structures
the first-order theory of $\mathcal{M}$: $\text{Th}(\mathcal{M}) = \{ G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G \}$
Logical theories

Syntactic view
Axiomatized by a set $\mathcal{F}$ of (closed) first-order $\Sigma$-formulae. The models of $\mathcal{F}$: $\text{Mod}(\mathcal{F}) = \{ \mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F} \}$

Semantic view
given a class $\mathcal{M}$ of $\Sigma$-structures the first-order theory of $\mathcal{M}$: $\text{Th}(\mathcal{M}) = \{ G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G \}$

$\mathcal{F} \subseteq \text{Th}(\text{Mod}(\mathcal{F}))$ (typically strict)
$\mathcal{M} \subseteq \text{Mod}(\text{Th}(\mathcal{M}))$ (typically strict)

$\text{Th}(\text{Mod}(\mathcal{F}))$ the set of formulae true in all models of $\mathcal{F}$ represents exactly the set of consequences of $\mathcal{F}$
Examples

1. **Linear integer arithmetic.** \( \Sigma = (\{0/0, s/1, +/2\}, \{\leq /2\}) \)

   \( \mathbb{Z}^+ = (\mathbb{Z}, 0, s, +, \leq) \) the standard interpretation of integers.

   \( \{\mathbb{Z}^+\} \subset \text{Mod}(\text{Th}(\mathbb{Z}^+)) \)

2. **Uninterpreted function symbols.** \( \Sigma = (\Omega, \text{Pred}) \)

   \( \mathcal{M} = \Sigma\text{-alg}: \) the class of all \( \Sigma \)-structures

   The theory of uninterpreted function symbols is \( \text{Th}(\Sigma\text{-alg}) \)
   the family of all first-order formulae which are true in all \( \Sigma \)-structures.
Examples

3. Lists. $\Sigma = (\{\text{car/1, cdr/1, cons/2}\}, \emptyset)$

$\mathcal{F} = \begin{cases} 
\text{car}(\text{cons}(x, y)) & \approx x \\
\text{cdr}(\text{cons}(x, y)) & \approx y \\
\text{cons}(\text{car}(x), \text{cdr}(x)) & \approx x 
\end{cases}$

$\text{Mod}(\mathcal{F})$: the class of all models of $\mathcal{F}$
$\text{Th}_{\text{Lists}} = \text{Th}(\text{Mod}(\mathcal{F}))$ theory of lists (axiomatized by $\mathcal{F}$)

4. Arrays. $\Sigma = (\{\text{read/2, write/3}\})$

$\mathcal{F} = \begin{cases} 
\text{read}(\text{write}(a, i, e), i) & = e \\
i \neq j \rightarrow \text{read}(\text{write}(a, i, e), j) & = \text{read}(a, j) 
\end{cases}$

$\text{Mod}(\mathcal{F})$: the class of all models of $\mathcal{F}$
$\text{Th}_{\text{Arrays}} = \text{Th}(\text{Mod}(\mathcal{F}))$ theory of arrays (axiomatized by $\mathcal{F}$)
Decidable theories

$\Sigma = (\Omega, \text{Pred})$ be a signature.

$\mathcal{M}$: class of $\Sigma$-structures. $\mathcal{T} = \text{Th}(\mathcal{M})$ is decidable

iff

there is an algorithm which, for every closed first-order formula $\phi$, can decide (after a finite number of steps) whether $\phi$ is in $\mathcal{T}$ or not.

$\mathcal{F}$: class of (closed) first-order formulae.

The theory $\mathcal{T} = \text{Th}(\text{Mod}(\mathcal{F}))$ is decidable

iff

there is an algorithm which, for every closed first-order formula $\phi$, can decide (in finite time) whether $\mathcal{F} \models \phi$ or not.
Examples

Undecidable theories

• Peano arithmetic

Axiomatized by:

\[ \forall x \neg (x + 1 \approx 0) \] (zero)

\[ \forall x \forall y (x + 1 \approx y + 1 \rightarrow x \approx y) \] (successor)

\[ F[0] \land (\forall x (F[x] \rightarrow F[x + 1]) \rightarrow \forall x F[x]) \] (induction)

\[ \forall x (x + 0 \approx x) \] (plus zero)

\[ \forall x, y (x + (y + 1) \approx (x + y) + 1) \] (plus successor)

\[ \forall x, y (x \ast 0 \approx 0) \] (times zero)

\[ \forall x, y (x \ast (y + 1) \approx x \ast y + x) \] (times successor)

\[ 3 \ast y + 5 > 2 \ast y \] expressed as \[ \exists z (z \neq 0 \land 3 \ast y + 5 \approx 2 \ast y + z) \]

Intended interpretation: \( (\mathbb{N}, \{0, 1, +, \ast\}, \{\approx, \leq\}) \)

(does not capture true arithmetic by Gödel’s incompleteness theorem)

• \( \text{Th}((\mathbb{Z}, \{0, 1, +, \ast\}, \{\leq\})) \)

• \( \text{Th}(\Sigma\text{-alg}) \)
Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments
Examples

In order to obtain decidability results:

- **Restrict the signature**
- **Enrich axioms**
- **Look at certain fragments**

Decidable theories

- Presburger arithmetic decidable in 3EXPTIME [Presburger’29]
  Signature: \((\{0, 1, +\}, \{\sim, \leq\})\) (no \(\ast\))
  
  Axioms \{ (zero), (successor), (induction), (plus zero), (plus successor) \}

  A decision procedure will be presented by Julia Schönberger

- \(\text{Th}(\mathbb{Z}_+)\) \(\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +, \leq)\) the standard interpretation of integers.
Examples

In order to obtain decidability results:

• Restrict the signature

• Enrich axioms

• Look at certain fragments

Decidable theories

• The theory of real numbers (with addition and multiplication) is decidable in 2EXPTIME [Tarski’30]
Examples

In order to obtain decidability results:

• Restrict the signature

• Enrich axioms

• Look at certain fragments \( \mathcal{L} \subseteq \text{Fma}(\Sigma) \)

“Simpler” task: Given \( \phi \) in \( \mathcal{L} \), is it the case that \( \mathcal{T} \models \phi \)?

Common restrictions on \( \mathcal{L} \)

<table>
<thead>
<tr>
<th>( \mathcal{L} )</th>
<th>Pred = ( \emptyset )</th>
<th>( { \phi \in \mathcal{L} \mid \mathcal{T} \models \phi } )</th>
</tr>
</thead>
<tbody>
<tr>
<td>{( \forall x A(x) \mid A \text{ atomic} } )</td>
<td>word problem</td>
<td></td>
</tr>
<tr>
<td>{( \forall x (A_1 \land \ldots \land A_n \rightarrow B) \mid A_i, B \text{ atomic} } )</td>
<td>uniform word problem</td>
<td></td>
</tr>
<tr>
<td>{( \forall x C(x) \mid C(x) \text{ clause} } )</td>
<td>clausal validity problem</td>
<td></td>
</tr>
<tr>
<td>{( \forall x \phi(x) \mid \phi(x) \text{ unquantified} } )</td>
<td>universal validity problem</td>
<td></td>
</tr>
</tbody>
</table>

\( \text{Th}_{\forall_{\text{Horn}}} \)

\( \text{Th}_{\forall,\text{cl}} \)

\( \text{Th}_{\forall} \)
Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments $L \subseteq \text{Fma}(\Sigma)$

**“Simpler” task:** Given $\phi$ in $L$, is it the case that $\mathcal{T} \models \phi$?

Common restrictions on $L$

<table>
<thead>
<tr>
<th>$\Pi = \emptyset$</th>
<th>${\phi \in L \mid \mathcal{T} \models \phi}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L = {\exists x A_1 \land \ldots \land A_n \mid A_i \text{ atomic}}$</td>
<td>unification problem $\text{Th}_{\exists}$</td>
</tr>
<tr>
<td>$L = {\forall x \exists x A_1 \land \ldots \land A_n \mid A_i \text{ atomic}}$</td>
<td>unification with constants $\text{Th}_{\forall \exists}$</td>
</tr>
</tbody>
</table>
Validity of ∀ formulae vs. ground satisfiability

The following are equivalent:

(1) $\mathcal{T} \models \forall x(L_1(x) \lor \cdots \lor L_n(x))$

(2) There is no model of $\mathcal{T}$ which satisfies $\exists x(\neg L_1(x) \land \cdots \land \neg L_n(x))$

(3) There is no model of $\mathcal{T}$ and no valuation for the constants $c$ for which $(\neg L_1(c) \land \cdots \land \neg L_n(c))$ becomes true

(notation: $(\neg L_1(c) \land \cdots \land \neg L_n(c)) \models_{\mathcal{T}} \bot)$

Can reduce any validity problem to a ground satisfiability problem
Many example of theories in which ground satisfiability is decidable:

- The empty theory (no axioms) \( \text{UIF}(\Sigma) \)
- linear (rational or integer) arithmetic
- theories axiomatizing common datatypes (lists, arrays)
The theory of uninterpreted function symbols

Let $\Sigma = (\Omega, \Pi)$ be arbitrary

Let $\mathcal{M} = \Sigma\text{-alg}$ be the class of all $\Sigma$-structures

The theory of uninterpreted function symbols is $\text{Th}(\Sigma\text{-alg})$ the family of all first-order formulae which are true in all $\Sigma$-algebras.

- in general undecidable

- Satisfiability of conjunctions of ground literals is decidable (in PTIME)

**Method 1:** Ackermann’s reduction

\[
f(a, b) = a \land f(f(a, b), b) \neq a \quad \Rightarrow \quad \begin{cases} 
    f(a, b) = c, f(c, b) = d \\
    c = a \land d \neq a \\
    a = c \land b = b \rightarrow c = d 
\end{cases}
\]
The theory of uninterpreted function symbols

Let $\Sigma = (\Omega, \Pi)$ be arbitrary

Let $\mathcal{M} = \Sigma$-alg be the class of all $\Sigma$-structures

The theory of uninterpreted function symbols is $\text{Th}(\Sigma$-alg) the family of all first-order formulae which are true in all $\Sigma$-algebras.

- in general undecidable

- Satisfiability of conjunctions of ground literals is decidable (in PTIME)

**Method 2:** DAG encoding [Downey-Sethi, Tarjan’76; Nelson-Oppen’80]

$$f(a, b) = a \models f(f(a, b), b) = a \rightarrow$$

Compute the “congruence closure” $R^c$ of $R$ / check whether $(v_1, v_3) \in R^c$
The theory of uninterpreted function symbols

Let $\Sigma = (\Omega, \Pi)$ be arbitrary

Let $\mathcal{M} = \Sigma\text{-alg}$ be the class of all $\Sigma$-structures

The theory of uninterpreted function symbols is $\text{Th}(\Sigma\text{-alg})$ the family of all first-order formulae which are true in all $\Sigma$-algebras.

- in general undecidable

- Satisfiability of conjunctions of ground literals is decidable (in PTIME)

**Method 3:** Use term rewrite systems to define normal forms “(representatives of equivalence classes) [Bachmair, Tiwari]

D. Friedrich: Abstract congruence closure

(I will present some background on term-rewrite systems in a next meeting.)
Linear (rational or integer) arithmetic

Syntax

• Signature \( \Sigma = (\{0/0, s/1, +/2\}, \{\leq /2, < /2\}) \)

• Terms, atomic formulae – as usual

    Example:

    \[ 3 \times x_1 + 2 \times x_2 \leq 5 \times x_3 \text{ abbreviation for} \]

    \[ (x_1 + x_1 + x_1) + (x_2 + x_2) \leq (x_3 + x_3 + x_3 + x_3 + x_3) \]

• Formulae: conjunctions of atomic formulae

A very simple fragment of linear arithmetic ("Unit two-variables per inequality") will be presented by David Nagel
Application: Program Verification

i := 1;       [** where 1 <= n < m  **]
while i < n
  do
    i := i + 1;
    [** part of a program in which  i  is used as an index in an array
       which was declared to be of size s > m (and i is not changed) **]
  od

Task: Check whether  i  ≤  s  always during the execution of this program.
**Application: Program Verification**

\[
i := 1; \quad \text{[** where } 1 \leq n < m \text{ **]}
\]
while \( i < n \)
do
\[
i := i + 1;
\]
[** part of a program in which \( i \) is used as an index in an array which was declared to be of size \( s > m \) (and \( i \) is not changed) **]

\[
\ldots
\]
\[
\text{od}
\]

**Task:** Check whether \( i \leq s \) always during the execution of this program.

**Solution:** Show that this is true at the beginning and remains true after every update of \( i \).
Application: Program Verification

\[ i := 1; \quad [** \text{where } 1 \leq n < m \text{ **}] \]
while \( i < n \)
do
  \[ i := i + 1; \]
  [** part of a program in which \( i \) is used as an index in an array which was declared to be of size \( s > m \) (and \( i \) is not changed) **]
  \[ \ldots \]
do

**Task:** Check whether \( i \leq s \) always during the execution of this program.

**Solution:** Show that \( i \leq s \) is an invariant of the program:

1) It holds at the first line: \( i = 1 \rightarrow i \leq s \)

2) It is preserved under the updates in the while loop:
\[ \forall n, m, s, i, i' \quad (1 \leq n < m < s \land i < n \land i \leq s \land i' \approx i + 1 \rightarrow i' \leq s) \]
Linear arithmetic over \( \mathbb{N} \) or \( \mathbb{Z} \)

Presburger arithmetic decidable in 3EXPTIME [Presburger’29]

• automata theoretic method (will be presented by Julia Schöenberger)

Linear arithmetic over \( \mathbb{Z} \):
check satisfiability of conjunctions of (in)equalities over \( \mathbb{Z} \): NP-hard

• Integer linear programming
  use branch-and-bound/cutting planes

• The Omega test – use variable elimination
Linear arithmetic over \( \mathbb{R} \) or \( \mathbb{Q} \)

- \( \text{Th}(\mathbb{R}) \)
  \( \mathbb{R} = (\mathbb{R}, \{0, 1, +\}, \{<\}) \) the standard interpretation of real numbers;

- \( \text{Th}(\mathbb{Q}) \)
  \( \mathbb{Q} = (\mathbb{Q}, \{0, 1, +\}, \{<\}) \) the standard interpretation of rational numbers.
Linear arithmetic over $\mathbb{R}$ or $\mathbb{Q}$

Axiomatization:

The equational part of linear rational arithmetic is described by the theory of divisible torsion-free abelian groups:

- $\forall x, y, z (x + (y + z) \approx (x + (y + z)))$ (associativity)
- $\forall x, y (x + y \approx y + x)$ (commutativity)
- $\forall x (x + 0 \approx x)$ (identity)
- $\forall x \exists y (x + y \approx 0)$ (inverse)
- For all $n \geq 1$: $\forall x (\underbrace{x + \cdots + x}_{n \text{ times}} \approx 0 \rightarrow x \approx 0)$ (torsion-freeness)
- For all $n \geq 1$: $\forall x \exists y (\underbrace{y + \cdots + y}_{n \text{ times}} \approx x)$ (divisibility)
- $\neg 1 \approx 0$ (non-triviality)

Note: Quantification over natural numbers is not part of our language. We really need infinitely many axioms for torsion-freeness and divisibility.
Linear arithmetic over $\mathbb{R}$ or $\mathbb{Q}$

By adding the axioms of a compatible strict total ordering, we define ordered divisible abelian groups:

\begin{align*}
\forall x \; (\neg x < x) & \quad \text{(irreflexivity)} \\
\forall x, y, z \; (x < y \land y < z \rightarrow x < z) & \quad \text{(transitivity)} \\
\forall x, y \; (x < y \lor y < x \lor x \approx y) & \quad \text{(totality)} \\
\forall x, y, z \; (x < y \rightarrow x + z < y + z) & \quad \text{(compatibility)} \\
0 < 1 & \quad \text{(non-triviality)}
\end{align*}

**Note:** The second non-triviality axiom renders the first one superfluous.

Moreover, as soon as we add the axioms of compatible strict total orderings, torsion-freeness can be omitted.

Every ordered divisible abelian group is obviously torsion-free. In fact the converse holds: Every torsion-free abelian group can be ordered [F.-W. Levi, 1913].

**Examples:** $\mathbb{Q}, \mathbb{R}, \mathbb{Q}^n, \mathbb{R}^n, \ldots$
The signature can be extended by further symbols:

- $\leq /2, > /2, \geq /2, \not\approx /2$: defined using $<$ and $\approx$
- $- /1$: Skolem function for inverse axiom
- $- /2$: defined using $+ /2$ and $- /1$
- $\text{div}_n /1$: Skolem functions for divisibility axiom for all $n \geq 1$.
- $\text{mult}_n /1$: defined by $\forall x (\text{mult}_n (x) \approx x + \cdots + x$ for all $n \geq 1$. $n$ times
- $\text{mult}_q /1$: defined using $\text{mult}_n, \text{div}_n, -$ for all $q \in \mathbb{Q}$.
  (We usually write $q \cdot t$ or $qt$ instead of $\text{mult}_q (t)$.)
- $q /0$ (for $q \in \mathbb{Q}$): defined by $q \approx q \cdot 1$.

**Note:** Every formula using the additional symbols is ODAG-equivalent to a formula over the base signature.

When $\cdot$ is considered as a binary operator, (ordered) divisible torsion-free abelian groups correspond to (ordered) rational vector spaces.
Linear arithmetic over \( \mathbb{R} \) or \( \mathbb{Q} \)

**Theorem.**

(1) The satisfiability of any conjunction of (strict and non-strict) linear inequalities can be checked in PTIME [Khakian’79].

(2) The complexity of checking the satisfiability of sets of clauses in linear arithmetic is in NP [Sonntag’85].

**Literature**


Linear arithmetic over \( \mathbb{R} \) or \( \mathbb{Q} \)

**Methods** The algorithms currently used are not PTIME.

- The simplex method
- The Fourier-Motzkin method – use variable elimination
**Problem:** check satisfiability of conjunctions of atomic formulae in linear arithmetic over a numerical domain $D$

**Complexity:** $D = \mathbb{R}$: PTIME; $D = \mathbb{Z}$: NP-hard

**Methods**

- The simplex method ($D = \mathbb{R}$)
- Integer linear programming ($D = \mathbb{Z}$)
  - use branch-and-bound/cutting planes
- The Fourier-Motzkin method ($D = \mathbb{R}$)
  - use variable elimination
- The Omega test ($D = \mathbb{Z}$)
  - use variable elimination
Theories axiomatizing common datatypes
Theories axiomatizing common datatypes

Example 1: McCarthy’s theory of arrays.
\[ \Sigma = \{ \text{write}/3, \text{read}/2 \} \]

Axioms:

\[
\begin{align*}
\text{read}(\text{write}(x, i, y), i) &= y \\
i \neq j \rightarrow \text{read}(\text{write}(x, i, y), j) &= \text{read}(x, j) \\
a = b &\iff \forall i(\text{read}(a, i) = \text{read}(b, i))
\end{align*}
\]

- the full first-order theory of arrays is undecidable
- the ground satisfiability problem is decidable (in NP)

Methods:
- refinements of resolution which efficiently handle equality [ARR’02]
- instantiation Shuyi Weng: What’s decidable about arrays?
Theories axiomatizing common datatypes

Example 2: The theory of acyclic lists

Axioms:

\[
\begin{align*}
\text{car}(\text{cons}(x, y)) &= x \\
\text{cdr}(\text{cons}(x, y)) &= y \\
\text{cons}(\text{car}(x), \text{cdr}(x)) &= x \\
\text{t}(x) &\neq x \quad \text{t contains only cons}
\end{align*}
\]

The full first-order theory is decidable (but non-elementary)
Satisfiability of conjunctions of ground literals decidable (in PTIME).

Methods:
- refinements of resolution which efficiently handle equality [ARR’02]
- instantiation/bidirectional closure: Killian Laudt
Theories axiomatizing common datatypes

**Example 3:** doubly-linked lists (cf. also [Necula, McPeak 2005])

\[
\forall p \ (p \neq \text{null} \land p.p\text{next} \neq \text{null} \rightarrow p.p\text{next}.p\text{rev} = p)
\]

\[
\forall p \ (p \neq \text{null} \land p.p\text{rev} \neq \text{null} \rightarrow p.p\text{rev}.p\text{next} = p)
\]

\[
\land \ c \neq \text{null} \land c.p\text{next} \neq \text{null} \land d \neq \text{null} \land d.p\text{next} \neq \text{null} \land c.p\text{next}=d.p\text{next} \land c \neq d \models \bot
\]
Example 3: doubly-linked lists (cf. also [Necula, McPeak 2005])

\[
\begin{align*}
(c \neq \text{null} \land c.\text{next} \neq \text{null} \rightarrow c.\text{next}.\text{prev} = c) \\
(d \neq \text{null} \land d.\text{next} \neq \text{null} \rightarrow d.\text{next}.\text{prev} = d)
\end{align*}
\]
Overview

• **Reasoning in standard theories**
  
  A crash course: Decidable logical theories and theory fragments

Reasoning in complex theories

  Modular reasoning in combinations of theories
  
  disjoint signature: the Nelson-Oppen method

• **Applications**