Seminar Decision Procedures and Applications

Background Information

Viorica Sofronie-Stokkermans

University Koblenz-Landau

29 May 2018

Topics for the talks

- **Tobias Justinger:** Difference Logic and UTVPI Constraints
- **Christoph Noll:** Automata approach to Presburger arithmetic
- **Sebastian Beck:** Quantifier elimination for linear arithmetic over the integers
- **Florian Kähne:** Reasoning about uninterpreted function symbols

- **Johannes Thielen:** Instantiation-based decision procedures for theories of arrays.
- **Christopher Biehl:** Decision procedures for recursive data structures with integer constraints
- **Jan Savelsberg:** Data Structure Specifications via Local Equality Axioms.
- **Thomas Senkowski:** Decision procedures for sets of cardinalities

- **Alexander Scheid-Rehder:** Invariant checking; Bounded model checking
- **Isabelle Kuhlmann:** Interpolation
- **Jan Krämer:** Verification by abstraction/refinement.

Overview

We give a survey of decidability results in various theories.

- **Reasoning in standard theories**

- **Reasoning in complex theories**

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

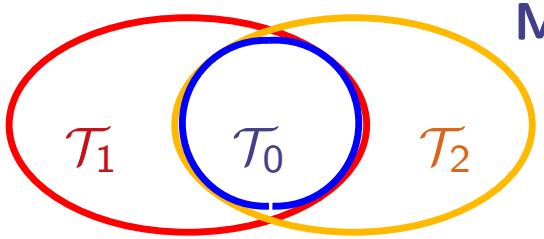
- **Numbers** - integers, reals, rationals

- **Data structures**
  - theories of lists of integers, reals, ...
  - theory of acyclic lists of integers, reals, ...
  - theory of arrays of integers, reals, ...
  - theories of sets of integers, reals, ...

  + functions (free, rec. def.) e.g : length, card

Modularity

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



Modular Reasoning

$\mathcal{T}_0$: Σ_0-theory.

$\mathcal{T}_i$: Σ_i-theory; $\quad \mathcal{T}_0 \subseteq \mathcal{T}_i \quad \Sigma_0 \subseteq \Sigma_i$.

Example:

lists$(\mathbb{R}) \cup$ **arrays**$(\mathbb{R})$

Can we use provers for $\mathcal{T}_1, \mathcal{T}_2$ as blackboxes to prove theorems in $\mathcal{T}_1 \cup \mathcal{T}_2$?
Which information needs to be exchanged between the provers?

Structure

Reasoning in standard theories

Presburger arithmetic: Christoph Noll, Sebastian Beck

Simpler fragments: UTVPI Tobias Justinger

Theory of uninterpreted function symbols

Graph theoretic approach: Florian Kähne

Theories of constructors and selectors: Christopher Biehl:

Theories of sets: Thomas Senkowski

Structure

Reasoning in complex theories

Modular reasoning in combinations of theories

Disjoint signature: The Nelson-Oppen method

- **Applications: complex data types**

  Fragment of theory of arrays: Johannes Thielen

  Recursive data types with length constraints: Christopher Biehl

  Fragment of theory of pointers: Jan Savelsberg

  Sets with cardinalities: Thomas Senkowski

Structure

Applications: verification, interpolation

Invariant checking, BMC: Alexander Scheid-Rehder:

Interpolation: Isabelle Kuhlmann

Abstraction/Refinement: Jan Krämer

Conventions

In what follows we will use the following conventions:

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

function symbols with arity ≥ 1 are denoted

- $f, g, h, \ldots$ if the formulae are interpreted into arbitrary algebras
- $+, -, s, \ldots$ if the intended interpretation is into numerical domains

predicate symbols with arity 0 are denoted $p, q, r, s, \ldots$

predicate symbols with arity ≥ 1 are denoted

- $P, Q, R, \ldots$ if the formulae are interpreted into arbitrary algebras
- $\leq, \geq, <, >$ if the intended interpretation is into numerical domains

variables are denoted $x, y, z, \ldots$

Logical theories

Syntactic view

Axiomatized by a set $\mathcal{F}$ of (closed) first-order Σ-formulae.

the models of $\mathcal{F}$: $\quad \mathsf{Mod}(\mathcal{F}) = \{\mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$

Semantic view

given a class $\mathcal{M}$ of Σ-structures

the first-order theory of $\mathcal{M}$: $\mathsf{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$

Logical theories

Syntactic view

Axiomatized by a set $\mathcal{F}$ of (closed) first-order Σ-formulae.

the models of $\mathcal{F}$: $\quad \mathrm{Mod}(\mathcal{F}) = \{\mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$

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

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

Semantic view

given a class $\mathcal{M}$ of Σ-structures

the first-order theory of $\mathcal{M}$: $\mathrm{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$

$\mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$ 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.

$\{\mathbb{Z}_+\} \subset \mathsf{Mod}(\mathsf{Th}(\mathbb{Z}_+))$

2. Uninterpreted function symbols. $\Sigma = (\Omega, \mathsf{Pred})$

$\mathcal{M} = \Sigma$-alg: the class of all Σ-structures

The theory of uninterpreted function symbols is Th(Σ-alg)
the family of all first-order formulae which are true in all Σ-structures.

Examples

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

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

$\mathsf{Mod}(\mathcal{F})$: the class of all models of $\mathcal{F}$

$\mathsf{Th}_{\mathsf{Lists}} = \mathsf{Th}(\mathsf{Mod}(\mathcal{F}))$ theory of lists (axiomatized by $\mathcal{F}$)

Decidable theories

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

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

iff

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

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

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

iff

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

Examples

Undecidable theories

- Peano arithmetic

Axiomatized by:

$$\forall x\, \neg(x + 1 \approx 0) \qquad \text{(zero)}$$

$$\forall x \forall y\, (x + 1 \approx y + 1 \rightarrow x \approx y \qquad \text{(successor)}$$

$$F[0] \wedge (\forall x\, (F[x] \rightarrow F[x + 1]) \rightarrow \forall x F[x]) \qquad \text{(induction)}$$

$$\forall x\, (x + 0 \approx x) \qquad \text{(plus zero)}$$

$$\forall x, y\, (x + (y + 1) \approx (x + y) + 1) \qquad \text{(plus successor)}$$

$$\forall x, y\, (x * 0 \approx 0) \qquad \text{(times zero)}$$

$$\forall x, y\, (x * (y + 1) \approx x * y + x) \qquad \text{(times successor)}$$

$3 * y + 5 > 2 * y$ expressed as $\exists z(z \not\approx 0 \wedge 3 * y + 5 \approx 2 * y + z)$

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

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

- $\mathrm{Th}((\mathbb{Z}, \{0, 1, +, *\}, \{\leq\}))$
- $\mathrm{Th}(\Sigma\text{-alg})$

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: $(\{0, 1, +\}, \{\approx, \leq\})$ (no $*$)

  Axioms { (zero), (successor), (induction), (plus zero), (plus successor) }

A decision procedure will be presented by Christoph Noll

A quantifier-elimination method with be presented by Sebastian Beck

A simple fragment (UTVPI) with be presented by Tobias Justinger

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 $\mathcal{L} \subseteq \mathsf{Fma}(\Sigma)$

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

Common restrictions on $\mathcal{L}$

	$\mathsf{Pred} = \emptyset$	$\{\phi \in \mathcal{L} \mid \mathcal{T} \models \phi\}$
$\mathcal{L}=\{\forall x A(x) \mid A \text{ atomic}\}$	word problem	
$\mathcal{L}=\{\forall x(A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B \text{ atomic}\}$	uniform word problem	$\mathsf{Th}_{\forall_{\mathsf{Horn}}}$
$\mathcal{L}=\{\forall x C(x) \mid C(x) \text{ clause}\}$	clausal validity problem	$\mathsf{Th}_{\forall,\mathsf{cl}}$
$\mathcal{L}=\{\forall x \phi(x) \mid \phi(x) \text{ unquantified}\}$	universal validity problem	$\mathsf{Th}_{\forall}$

Validity of $\forall$ formulae vs. ground satisfiability

The following are equivalent:

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

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

(3) There is no model of $\mathcal{T}$ and no valuation for the constants c

for which $(\neg L_1(c) \wedge \cdots \wedge \neg L_n(c))$ becomes true

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

Can reduce any validity problem to a ground satisfiability problem

Useful theories

Many example of theories in which ground satisfiability is decidable:

- The empty theory (no axioms) $UIF(\Sigma)$
- linear (rational or integer) arithmetic
- theories axiomatizing common datatypes (lists, arrays)

The theory of uninterpreted function symbols

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

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

The theory of uninterpreted function symbols is Th(Σ-alg) the family of all first-order formulae which are true in all Σ-algebras.

- in general undecidable

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

The theory of uninterpreted function symbols

$\Sigma = (\Omega, \Pi)$ be arbitrary; $\mathcal{M} = \Sigma$-alg the class of all Σ-structures

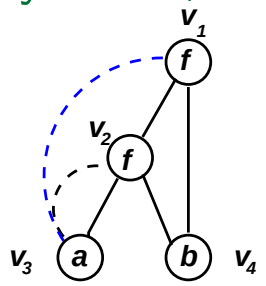
The theory of uninterpreted function symbols is Th(Σ-alg) the family of all first-order formulae which are true in all Σ-algebras.

- in general undecidable

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

Method 1: DAG encoding [Downey-Sethi, Tarjan'76; Nelson-Oppen'80]

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



v_1 : $f(f(a, b), b)$
v_2 : $f(a, b)$
v_3 : a
v_4 : b
R : $\{(v_2, v_3)\}$

Compute the "congruence closure" R^c of R / check whether $(v_1, v_3) \in R^c$

Florian Kähne

Reasoning in combinations of theories

We are interested in testing satisfiability of ground formulae

Combination of theories

Combinations of theories and models

Forgetting symbols

Let $\Sigma = (\Omega, \Pi)$ and $\Sigma' = (\Omega', \Pi')$ s.t. $\Sigma \subseteq \Sigma'$, i.e., $\Omega \subseteq \Omega'$ and $\Pi \subseteq \Pi'$

For $\mathcal{A} \in \Sigma'$-alg, we denote by $\mathcal{A}_{|\Sigma}$ the Σ-structure for which:

$$U_{\mathcal{A}_{|\Sigma}} = U_{\mathcal{A}}, \quad f_{\mathcal{A}_{|\Sigma}} = f_{\mathcal{A}} \quad \text{for } f \in \Omega;$$
$$P_{\mathcal{A}_{|\Sigma}} = P_{\mathcal{A}} \quad \text{for } P \in \Pi$$

(ignore functions and predicates associated with symbols in $\Sigma' \backslash \Sigma$)

$\mathcal{A}_{|\Sigma}$ is called the restriction (or the reduct) of $\mathcal{A}$ to Σ.

Example: $\Sigma' = (\{+/2, */2, 1/0\}, \{\leq /2, \text{even}/1, \text{odd}/1\})$

$\Sigma = (\{+/2, 1/0\}, \{\leq /2\}) \subseteq \Sigma'$

$\mathcal{N} = (\mathbb{N}, +, *, 1, \leq, \text{even}, \text{odd})$ $\quad$ $\mathcal{N}_{|\Sigma} = (\mathbb{N}, +, 1, \leq)$

One possibility of combining theories

Syntactic view: $\mathcal{T}_1 + \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \subseteq F_{\Sigma_1 \cup \Sigma_2}(X)$

$\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2\}$

where $\Sigma_1 \cup \Sigma_2 = (\Omega_1, \Pi_1) \cup (\Omega_2, \Pi_2) = (\Omega_1 \cup \Omega_2, \Pi_1 \cup \Pi_2)$

One possibility of combining theories

Syntactic view: $\mathcal{T}_1 + \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \subseteq F_{\Sigma_1 \cup \Sigma_2}(X)$

$\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2\}$

Semantic view: Let $\mathcal{M}_i = \mathsf{Mod}(\mathcal{T}_i), i = 1, 2$

$\mathcal{M}_1 + \mathcal{M}_2 = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i \text{ for } i = 1, 2\}$

One possibility of combining theories

Syntactic view: $\mathcal{T}_1 + \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \subseteq F_{\Sigma_1 \cup \Sigma_2}(X)$

$\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2\}$

Semantic view: Let $\mathcal{M}_i = \mathsf{Mod}(\mathcal{T}_i), i = 1, 2$

$\mathcal{M}_1 + \mathcal{M}_2 = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i \text{ for } i = 1, 2\}$

$$
\begin{array}{lll}
\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) & \text{iff} & \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2 \\
& \text{iff} & \mathcal{A}_{|\Sigma_i} \models G, \text{ for all } G \text{ in } \mathcal{T}_i, i = 1, 2 \\
& \text{iff} & \mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i, i = 1, 2 \\
& \text{iff} & \mathcal{A} \in \mathcal{M}_1 + \mathcal{M}_2
\end{array}
$$

One possibility of combining theories

Syntactic view: $\mathcal{T}_1 + \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \subseteq F_{\Sigma_1 \cup \Sigma_2}(X)$

$\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2\}$

Semantic view: Let $\mathcal{M}_i = \mathsf{Mod}(\mathcal{T}_i), i = 1, 2$

$\mathcal{M}_1 + \mathcal{M}_2 = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i \text{ for } i = 1, 2\}$

Remark: $\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2)$ iff $(\mathcal{A}_{|\Sigma_1} \in \mathsf{Mod}(\mathcal{T}_1)$ and $\mathcal{A}_{|\Sigma_2} \in \mathsf{Mod}(\mathcal{T}_2))$

Consequence: $\mathsf{Th}(\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2)) = \mathsf{Th}(\mathcal{M}_1 + \mathcal{M}_2)$

Example

1. Presburger arithmetic + UIF

$\mathrm{Th}(\mathbb{Z}_+) \cup \mathit{UIF}$ $\qquad$ $\Sigma = (\Omega, \Pi)$

Models: $(A, 0, s, +, \{f_A\}_{f \in \Omega}, \leq, \{P_A\}_{P \in \Pi})$

where $(A, 0, s, +, \leq) \in \mathrm{Mod}(\mathrm{Th}(\mathbb{Z}_+))$.

2. The theory of reals + the theory of a monotone function f

$\mathrm{Th}(\mathbb{R}) \cup \mathrm{Mon}(f)$ $\qquad$ $\mathrm{Mon}(f) : \forall x, y (x \leq y \rightarrow f(x) \leq f(y))$

Models: $(A, +, *, f_A, \{\leq\})$, where

where $(A, +, *, \leq) \in \mathrm{Mod}(\mathrm{Th}(\mathbb{R}))$.

$(A, f_A, \leq) \models \mathrm{Mon}(f)$, i.e. $f_A : A \rightarrow A$ monotone.

Note: The signatures of the two theories share the $\leq$ predicate symbol

Combinations of theories

Definition. A theory is consistent if it has at least one model.

Question: Is the union of two consistent theories always consistent?

Answer: No. (Not even when the two theories have disjoint signatures)

Example: $\Sigma_1 = (\Omega_1, \emptyset), \quad \Sigma_2 = (\{c/0, d/0\}, \emptyset), \quad c, d \notin \Omega_1$

$\mathcal{T}_1 = \{\exists x, y, z(x \not\approx y \ \wedge \ x \not\approx z \ \wedge \ y \not\approx z)\}$

$\mathcal{T}_2 = \{\forall x(x \approx c \ \vee \ x \approx d)\}$

$\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_1) \quad \text{iff} \quad |U_{\mathcal{A}}| \geq 3.$

$\mathcal{B} \in \mathsf{Mod}(\mathcal{T}_2) \quad \text{iff} \quad |U_{\mathcal{B}}| \leq 2.$

Combinations of theories

The combined decidability problem

For $i = 1, 2$

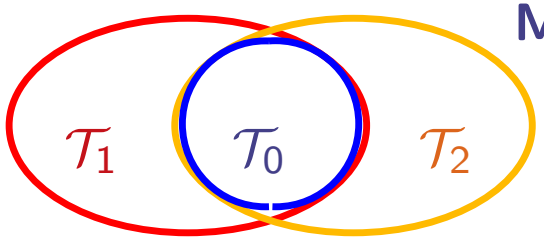
- let $\mathcal{T}_i$ be a first-order theory in signature Σ_i
- assume the $\mathcal{T}_i$ ground satisfiability problem is decidable

Let $\mathcal{T}_1 \bigoplus \mathcal{T}_2$ be a combination of $\mathcal{T}_1$ and $\mathcal{T}_2$

Question:

Is the $\mathcal{T}_1 \bigoplus \mathcal{T}_2$ ground satisfiability problem decidable?

Goal: Modularity



Modular Reasoning

$\mathcal{T}_0$: Σ_0-theory.

$\mathcal{T}_i$: Σ_i-theory; $\quad \mathcal{T}_0 \subseteq \mathcal{T}_i \quad \Sigma_0 \subseteq \Sigma_i$.

Example:

lists$(\mathbb{R}) \cup$ **arrays**$(\mathbb{R})$

Can use provers for $\mathcal{T}_1, \mathcal{T}_2$ as blackboxes to prove theorems in $\mathcal{T}_1 \cup \mathcal{T}_2$?

Which information needs to be exchanged between the provers?

Combination of theories over disjoint signatures

The Nelson/Oppen procedure

Given: $\mathcal{T}_1, \mathcal{T}_2$ stably infinite first-order theories with signatures Σ_1, Σ_2

Assume that $\Sigma_1 \cap \Sigma_2 = \emptyset$ (share only $\approx$)

P_i decision procedures for satisfiability of ground formulae w.r.t. $\mathcal{T}_i$

ϕ quantifier-free formula over $\Sigma_1 \cup \Sigma_2$

Task: Check whether ϕ is satisfiable w.r.t. $\mathcal{T}_1 \cup \mathcal{T}_2$

Note: Restrict to conjunctive quantifier-free formulae

$\phi \mapsto DNF(\phi)$

$DNF(\phi)$ satisfiable in $\mathcal{T}$ iff one of the disjuncts satisfiable in $\mathcal{T}$

Example

[Nelson & Oppen, 1979]

Theories

$\mathcal{R}$	theory of rationals	$\Sigma_{\mathcal{R}} = \{\leq, +, -, 0, 1\}$	$\approx$
$\mathcal{L}$	theory of lists	$\Sigma_{\mathcal{L}} = \{\mathsf{car}, \mathsf{cdr}, \mathsf{cons}\}$	$\approx$
$\mathcal{E}$	theory of equality (UIF)	Σ: free function and predicate symbols	$\approx$

Example

[Nelson & Oppen, 1979]

Theories

$\mathcal{R}$	theory of rationals	$\Sigma_{\mathcal{R}} = \{\leq, +, -, 0, 1\}$	$\approx$
$\mathcal{L}$	theory of lists	$\Sigma_{\mathcal{L}} = \{\mathsf{car}, \mathsf{cdr}, \mathsf{cons}\}$	$\approx$
$\mathcal{E}$	theory of equality (UIF)	Σ: free function and predicate symbols	$\approx$

Problems:

1. $\mathcal{R} \cup \mathcal{L} \cup \mathcal{E} \models \forall x, y (x \leq y \wedge y \leq x + \mathsf{car}(\mathsf{cons}(0, x)) \wedge P(h(x) - h(y)) \rightarrow P(0))$

2. Is the following conjunction:

$$c \leq d \ \wedge \ d \leq c + \mathsf{car}(\mathsf{cons}(0, c)) \ \wedge \ P(h(c) - h(d)) \ \wedge \ \neg P(0)$$

satisfiable in $\mathcal{R} \cup \mathcal{L} \cup \mathcal{E}$?

An Example

	$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
Σ	$\{\leq, +, -, 0, 1\}$	$\{\mathsf{car}, \mathsf{cdr}, \mathsf{cons}\}$	$F \cup P$
Axioms	$x + 0 \approx x$	$\mathsf{car}(\mathsf{cons}(x, y)) \approx x$	
	$x - x \approx 0$	$\mathsf{cdr}(\mathsf{cons}(x, y)) \approx y$	
(univ.	$+$ is A, C	$\mathsf{at}(x) \vee \mathsf{cons}(\mathsf{car}(x), \mathsf{cdr}(x)) \approx x$	
quantif.)	$\leq$ is R, T, A	$\neg\mathsf{at}(\mathsf{cons}(x, y))$	
	$x \leq y \vee y \leq x$		
	$x \leq y \rightarrow x + z \leq y + z$		

Is the following conjunction:

$$c \leq d \ \wedge \ d \leq c + \mathsf{car}(\mathsf{cons}(0, c)) \ \wedge \ P(h(c) - h(d)) \ \wedge \ \neg P(0)$$

satisfiable in $\mathcal{R} \cup \mathcal{L} \cup \mathcal{E}$?

Step 1: Purification

Given: ϕ conjunctive quantifier-free formula over $\Sigma_1 \cup \Sigma_2$

Task: Find ϕ_1, ϕ_2 s.t. ϕ_i is a pure Σ_i-formula and $\phi_1 \wedge \phi_2$ equivalent with ϕ

$$
\begin{array}{lcl}
f(s_1, \ldots, s_n) \approx g(t_1, \ldots, t_m) & \mapsto & u \approx f(s_1, \ldots, s_n) \wedge u \approx g(t_1, \ldots, t_m) \\
f(s_1, \ldots, s_n) \not\approx g(t_1, \ldots, t_m) & \mapsto & u \approx f(s_1, \ldots, s_n) \wedge v \approx g(t_1, \ldots, t_m) \wedge u \not\approx v \\
(\neg) P(\ldots, s_i, \ldots) & \mapsto & (\neg) P(\ldots, u, \ldots) \wedge u \approx s_i \\
(\neg) P(\ldots, s_i[t], \ldots) & \mapsto & (\neg) P(\ldots, s_i[t \mapsto u], \ldots) \wedge u \approx t
\end{array}
$$

where $t \approx f(t_1, \ldots, t_n)$

Termination: Obvious

Correctness: $\phi_1 \wedge \phi_2$ and ϕ equisatisfiable.

Step 1: Purification

$$c \leq d \ \wedge \ d \leq c + \mathsf{car}(\mathsf{cons}(0, c)) \ \wedge \ P(h(c) - h(d)) \ \wedge \ \neg P(0)$$

Step 1: Purification

$$c \leq d \ \wedge \ d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \ \wedge \ P(h(c) - h(d)) \ \wedge \ \neg P(0)$$

Step 1: Purification

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{h(c) - h(d)}_{c_2}) \;\wedge\; \neg P(0)$$

Step 1: Purification

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

Step 1: Purification

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$

Step 1: Purification

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
satisfiable	satisfiable	satisfiable

Step 2: Propagation

$$c \leq d \ \wedge \ d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \ \wedge \ P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \ \wedge \ \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$

deduce and propagate equalities between constants entailed by components

Step 2: Propagation

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
	$c_1 \approx c_5$	

Step 2: Propagation

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
$c_1 \approx c_5$	$c_1 \approx c_5$	
$c \approx d$		

Step 2: Propagation

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
$c_1 \approx c_5$	$c_1 \approx c_5$	$c \approx d$
$c \approx d$		$c_3 \approx c_4$

Step 2: Propagation

$$c \leq d \;\wedge\; d \leq c + \underbrace{\mathsf{car}(\mathsf{cons}(0, c))}_{c_1} \;\wedge\; P(\underbrace{\underbrace{h(c)}_{c_3} - \underbrace{h(d)}_{c_4}}_{c_2}) \;\wedge\; \neg P(\underbrace{0}_{c_5})$$

$\mathcal{R}$	$\mathcal{L}$	$\mathcal{E}$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
$c_1 \approx c_5$	$c_1 \approx c_5$	$c \approx d$
$c \approx d$		$c_3 \approx c_4$
$c_2 \approx c_5$		$\bot$

The Nelson-Oppen algorithm

ϕ conjunction of literals

Step 1. Purification $\mathcal{T}_1 \cup \mathcal{T}_2 \cup \phi \mapsto (\mathcal{T}_1 \cup \phi_1) \cup (\mathcal{T}_2 \cup \phi_2)$:

where ϕ_i is a pure Σ_i-formula and $\phi_1 \wedge \phi_2$ is equisatisfiable with ϕ.

Step 2. Propagation.

The decision procedure for ground satisfiability for $\mathcal{T}_1$ and $\mathcal{T}_2$ fairly exchange information concerning entailed unsatisfiability of constraints in the shared signature

i.e. clauses over the shared variables.

until an inconsistency is detected or a saturation state is reached.

The Nelson-Oppen algorithm

ϕ conjunction of literals

Step 1. Purification $\mathcal{T}_1 \cup \mathcal{T}_2 \cup \phi \mapsto (\mathcal{T}_1 \cup \phi_1) \cup (\mathcal{T}_2 \cup \phi_2)$:

where ϕ_i is a pure Σ_i-formula and $\phi_1 \wedge \phi_2$ is equisatisfiable with ϕ.

not problematic; requires linear time

Step 2. Propagation.

The decision procedure for ground satisfiability for $\mathcal{T}_1$ and $\mathcal{T}_2$ fairly exchange information concerning entailed unsatisfiability of constraints in the shared signature

i.e. clauses over the shared variables.

until an inconsistency is detected or a saturation state is reached.

not problematic; termination guaranteed

Sound: if inconsistency detected input unsatisfiable

Complete: under additional assumptions

Implementation

ϕ conjunction of literals

Step 1. Purification: $\mathcal{T}_1 \cup \mathcal{T}_2 \cup \phi \mapsto (\mathcal{T}_1 \cup \phi_1) \cup (\mathcal{T}_2 \cup \phi_2)$,
where ϕ_i is a pure Σ_i-formula and $\phi_1 \wedge \phi_2$ is equisatisfiable with ϕ.

Step 2. Propagation: The decision procedure for ground satisfiability for $\mathcal{T}_1$ and $\mathcal{T}_2$ fairly exchange information concerning entailed unsatisfiability of constraints in the shared signature
i.e. clauses over the shared variables.

until an inconsistency is detected or a saturation state is reached.

How to implement Propagation?

Guessing: guess a maximal set of literals containing the shared variables; check it for $\mathcal{T}_i \cup \phi_i$ consistency.

Backtracking: identify disjunction of equalities between shared variables entailed by $\mathcal{T}_i \cup \phi_i$; make case split by adding some of these equalities to ϕ_1, ϕ_2. Repeat as long as possible.

The Nelson-Oppen algorithm

Termination: only finitely many shared variables to be identified

Soundness: If procedure answers "unsatisfiable" then ϕ is unsatisfiable

Completeness: Under additional hypotheses

Completeness

Example:

E_1	E_2
$f(g(x), g(y)) \approx x$	$k(x) \approx k(x)$
$f(g(x), h(y)) \approx y$	
non-trivial	non-trivial

$g(c) \approx h(c) \wedge k(c) \not\approx c$

$g(c) \approx h(c)$	$k(c) \not\approx c$
satisfiable in E_1	satisfiable in E_2

no equations between shared variables; Nelson-Oppen answers "satisfiable"

Completeness

Example:

E_1	E_2
$f(g(x), g(y)) \approx x$	$k(x) \approx k(x)$
$f(g(x), h(y)) \approx y$	
non-trivial	non-trivial

$g(c) \approx h(c) \wedge k(c) \not\approx c$

$g(c) \approx h(c)$ satisfiable in E_1 $\qquad$ $k(c) \not\approx c$ satisfiable in E_2

no equations between shared variables; Nelson-Oppen answers "satisfiable"

A model of E_1 satisfies $g(c) \approx h(c)$ iff $\exists e \in A$ s.t. $g(e) = h(e)$.

Then, for all $a \in A$: $a = f_A(g(a), g(e)) = f_A(g(a), h(e)) = e$

$g(c) \approx h(c) \wedge k(c) \not\approx c$ unsatisfiable

Completeness

Another example

$\mathcal{T}_1$ theory admitting models of cardinality at most 2

$\mathcal{T}_2$ theory admitting models of any cardinality

$$f_1 \in \Sigma_1, f_2 \in \Sigma_2 \quad \text{such that} \quad \mathcal{T}_i \not\models \forall x, y \quad f_i(x) = f_i(y).$$

$$\phi \;=\; f_1(c_1) \not\approx f_1(c_2) \;\wedge\; f_2(c_1) \not\approx f_2(c_3) \;\wedge\; f_2(c_2) \not\approx f_2(c_3)$$

$$\phi_1 \;=\; f_1(c_1) \not\approx f_1(c_2) \qquad \phi_2 \;=\; f_2(c_1) \not\approx f_2(c_3) \;\wedge\; f_2(c_2) \not\approx f_2(c_3)$$

The Nelson-Oppen procedure returns "satisfiable"

$$\mathcal{T}_1 \cup \mathcal{T}_2 \models \forall x, y, z (f_1(x) \not\approx f_1(y) \;\wedge\; f_2(x) \not\approx f_2(z) \;\wedge\; f_2(y) \not\approx f_2(z) \rightarrow (x \not\approx y \wedge x \not\approx z \wedge y \not\approx z))$$

$f_1(c_1) \not\approx f_1(c_2) \;\wedge\; f_2(c_1) \not\approx f_2(c_3) \;\wedge\; f_2(c_2) \not\approx f_2(c_3)$ unsatisfiable

Completeness

Cause of incompleteness

There exist formulae satisfiable in finite models of bounded cardinality

Solution: Consider stably infinite theories.

> $\mathcal{T}$ is stably infinite iff for every quantifier-free formula ϕ
>
> ϕ satisfiable in $\mathcal{T}$ iff ϕ satisfiable in an infinite model of $\mathcal{T}$.

Note: This restriction is not mentioned in [Nelson Oppen 1979];

introduced by Oppen in 1980.

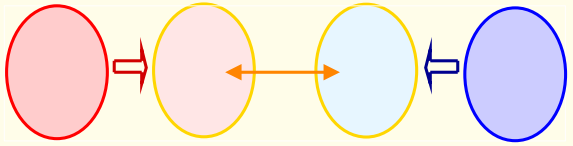
Completeness

Guessing version: C set of constants shared by ϕ_1, ϕ_2

R equiv. relation assoc. with partition of $C \mapsto ar(C,R) = \bigwedge_{R(c,d)} c \approx d \wedge \bigwedge_{\neg R(c,d)} c \not\approx d$

Lemma. Assume that there exists a partition of C s.t. $\phi_i \wedge ar(C,R)$ is $\mathcal{T}_i$-satisfiable. Then $\phi_1 \wedge \phi_2$ is $\mathcal{T}_1 \cup \mathcal{T}_2$-satisfiable.

Idea of proof: Let $\mathcal{A}_i \in \mathrm{Mod}(\mathcal{T}_i)$ s.t. $\mathcal{A}_i \models \phi_i \wedge ar(C,R)$. Then $c_{A_1} = d_{A_1}$ iff $c_{A_2} = d_{A_2}$.
Let $i : \{c_{A_1} \mid c \in C\} \to \{c_{A_2} \mid c \in C\}$, $i(c_{A_1}) = c_{A_2}$ well-defined; bijection.
Stable infinity: can assume w.l.o.g. that $\mathcal{A}_1, \mathcal{A}_2$ have the same cardinality
Let $h : \mathcal{A}_1 \to \mathcal{A}_2$ bijection s.t. $h(c_{A_1}) = c_{A_2}$
Use h to transfer the Σ_1-structure on $\mathcal{A}_2$.



Theorem. If $\mathcal{T}_1, \mathcal{T}_2$ are both stably infinite and the shared signature is empty then the Nelson-Oppen procedure is sound, complete and terminating.
Thus, it transfers decidability of ground satisfiability from $\mathcal{T}_1, \mathcal{T}_2$ to $\mathcal{T}_1 \cup \mathcal{T}_2$.

Applications

1. Decision Procedures for data types

Theories of arrays

We consider the theory of arrays in a many-sorted setting.

Theory of arrays $\mathcal{T}_{arrays}$:

- $\mathcal{T}_i$ (theory of indices): Presburger arithmetic
- $\mathcal{T}_e$ (theory of elements): arbitrary
- Axioms for read, write

$$\begin{aligned} \mathit{read}(\mathit{write}(a, i, e), i) &\approx e \\ j \not\approx i \vee \mathit{read}(\mathit{write}(a, i, e), j) &= \mathit{read}(a, j). \end{aligned}$$

Theories of arrays

We consider the theory of arrays in a many-sorted setting.

Theory of arrays $\mathcal{T}_{arrays}$:

- $\mathcal{T}_i$ (theory of indices): Presburger arithmetic
- $\mathcal{T}_e$ (theory of elements): arbitrary
- Axioms for read, write

$$\begin{aligned} read(write(a, i, e), i) &\approx e \\ j \not\approx i \vee read(write(a, i, e), j) &= read(a, j). \end{aligned}$$

Fact: Undecidable in general.

Goal: Identify a fragment of the theory of arrays which is decidable.

A decidable fragment

- **Index guard** a positive Boolean combination of atoms of the form $t \leq u$ or $t = u$ where t and u are either a variable or a ground term of sort Index

  Example: $(x \leq 3 \vee x \approx y) \wedge y \leq z$ is an index guard

  Example: $x + 1 \leq c, \quad x + 3 \leq y, \quad x + x \leq 2$ are not index guards.

- **Array property formula** [Bradley,Manna,Sipma'06]

  $(\forall i)(\varphi_I(i) \rightarrow \varphi_V(i))$, where:

  φ_I: index guard

  φ_V: formula in which any universally quantified i occurs in a direct array read; no nestings

  Example: $c \leq x \leq y \leq d \rightarrow a(x) \leq a(y)$ is an array property formula

  Example: $x < y \rightarrow a(x) < a(y)$ is not an array property formula

Johannes Thielen: Decision procedure for the array property fragment

Theories of recursive data structures with size

Theories of constructors/selectors

Lists (cons/car/cdr)

Binary trees (tree/left/right)

Size functions:

Lists:

$\text{size}(\text{nil}) = 0$
$\text{size}(\text{cons}(a, l)) = 1 + \mathit{size}(l)$

Trees

$\text{size}(\text{nil}) = 0$
$\text{size}(\text{tree}(t_1, t_2)) = 1 + \mathit{size}(t_1) + \mathit{size}(t_2)$

Pointer Structures

[McPeak, Necula 2005]

- pointer sort p, scalar sort s; pointer fields (p $\rightarrow$ p); scalar fields (p $\rightarrow$ s);
- axioms: $\forall p \;\; \mathcal{E} \vee \mathcal{C}$; $\mathcal{E}$ contains disjunctions of pointer equalities
  $\mathcal{C}$ contains scalar constraints

Assumption: If $f_1(f_2(\ldots f_n(p)))$ occurs in axiom, the axiom also contains:
$p{=}\text{null} \vee f_n(p){=}\text{null} \vee \cdots \vee f_2(\ldots f_n(p))){=}\text{null}$

Example: doubly-linked lists; ordered elements

$$\forall p\ (p \neq \text{null} \wedge p.\text{next} \neq \text{null} \rightarrow p.\text{next.prev} = p)$$
$$\forall p\ (p \neq \text{null} \wedge p.\text{prev} \neq \text{null} \rightarrow p.\text{prev.next} = p)$$
$$\forall p\ (p \neq \text{null} \wedge p.\text{next} \neq \text{null} \rightarrow p.\text{info} \leq p.\text{next.info})$$

Jan Savelsberg: decision procedure for a fragment of the theory of pointers

Applications

2. Program Verification

Program	$\mapsto$	$T = (\Sigma, \text{Init}, \text{Update}(\Sigma, \Sigma'))$
Safety Property	$\mapsto$	Formula Safe

Task: Prove that the safety property always holds (in general difficult)

Invariant checking

$\text{Init} \models \text{Safe}$

$\text{Safe} \wedge \text{Update}(\Sigma, \Sigma') \models \text{Safe}'$

Bounded model checking: given $k \in \mathbb{N}$. Prove that for all $n \leq k$:

$\text{Init}(\Sigma^0) \wedge \text{Update}|(\Sigma^0, \Sigma^1) \wedge \cdots \wedge \text{Update}|(\Sigma^{n-1}, \Sigma^n) \models \text{Safe}(\Sigma^n)$

Applications

2. Program Verification

Abstraction/Refinement

- Approximate system with a finite state system
- Unsafe state reachable from initial state in finite state system?

  No: System safe

  Yes: Check whether path corresponds to a real path in concrete system

    Yes: Concrete system unsafe

    No: Refine abstraction/ use e.g. interpolants

Isabelle Kuhlmann: Interpolation

Jan Krämer: Verification by abstraction/refinement

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

- **Applications**