max planck institut
informatik

Automated Reasoning

**Martin Bromberger, Sibylle Möhle,
Simon Schwarz, Christoph Weidenbach**

Max Planck Institute for Informatics

January 5, 2023

Outline

Equational Logic

In the next couple of lectures we combine first-order logic with equalities.
In general, satisfiability of first-order formulas with respect to equality is undecidable. Even the word problem for conjunctions of equations is undecidable. However, satisfiability for ground first-order formulas is decidable.
Therefore, we start investigating equational logic by investigating conjunctions/sets of ground unit equations. For a set of unit (in)equations we write E.

The theory of equational logic is also known as EUF (equality with uninterpreted function symbols) and is one of the standard theories considered in SMT (Satisfiability Modulo Theories).

Equivalent formulations

An equational clause

$$\forall \vec{x}\,(t_1 \approx s_1 \vee \ldots \vee t_n \approx s_n \vee l_1 \not\approx r_1 \vee \ldots \vee l_k \not\approx r_k)$$

is valid iff

$$\exists \vec{x}\,(t_1 \not\approx s_1 \wedge \ldots \wedge t_n \not\approx s_n \wedge l_1 \approx r_1 \wedge \ldots \wedge l_k \approx r_k)$$

is unsatisfiable iff the Skolemized (ground!) formula

$$(t_1 \not\approx s_1 \wedge \ldots \wedge t_n \not\approx s_n \wedge l_1 \approx r_1 \wedge \ldots \wedge l_k \approx r_k)\{\vec{x} \mapsto \vec{c}\}$$

is unsatisfiable iff the formula

$$(t_1 \approx s_1 \vee \ldots \vee t_n \approx s_n \vee l_1 \not\approx r_1 \vee \ldots \vee l_k \not\approx r_k)\{\vec{x} \mapsto \vec{c}\}$$

is valid.

Equivalent formulations

Please note validity of these transformations depends on the shape of the (starting) formula. Validity is not preserved in case of a quantifier alternation or an existentially quantified formula, in general, or the eventual formula must not be ground. There is no way to transform a first-order (equational) formula into a ground formula preserving validity, in general.

Congruence Closure

Input: $t_1 \not\approx s_1 \wedge \ldots \wedge t_n \not\approx s_n \wedge l_1 \approx r_1 \wedge \ldots \wedge l_k \approx r_k$

Main idea: transform the equations $E = \{l_1 \approx r_1, \ldots, l_k \approx r_k\}$ into an equivalent convergent TRS R and check whether $s_i{\downarrow_R} = t_i{\downarrow_R}$.

If $s_i{\downarrow_R} = t_i{\downarrow_R}$ for some i then $E \models s_i \approx t_i$ and the overall conjunction is unsatisfiable.

If $s_i{\downarrow_R} = t_i{\downarrow_R}$ for no i, i.e., $s_i{\downarrow_R} \neq t_i{\downarrow_R}$ for all i then the overall conjunction is satisfiable.

Flattening

$E = l_1 \approx r_1 \wedge \ldots \wedge l_n \approx r_n$

Flattening: ignore the inequations and transform the equations E so there are only two kinds of equations.

Term equations: $f(c_{i_1}, \ldots, c_{i_n}) \approx c_{i_0}$

Constant equations: $c_i \approx c_j$.

Flattening

$E = l_1 \approx r_1 \wedge \ldots \wedge l_n \approx r_n$

Flattening $E[f(t_1, \ldots, t_n)]_{p_1,\ldots,p_k} \Rightarrow_{\mathsf{CCF}}$
$E[c/p_1, \ldots, p_k] \wedge f(t_1, \ldots, t_n) \approx c$

provided all t_i are constants, the p_j are all positions in E of $f(t_1, \ldots, t_n)$, $|p_j| > 2$ for some $j \in \{1, \ldots, k\}$ or $p_j = q.2$ and $E|_{q.1}$ is not a constant for some q and $j \in \{1, \ldots, k\}$, and c is fresh

Note that $|p_j| > 2$ means that $f(t_1, \ldots, t_n)$ appears in E as an argument of a function symbol.

Note that $p_k = q.2$ and $E|_{q.1}$ is not a constant means that $E|_m$ is an equation with two complex terms on both sides, e.g., $g(a) \approx f(a, a)$.

As a result: only two kinds of equations left.

Term equations: $f(c_{i_1}, \ldots, c_{i_n}) \approx c_{i_0}$

Constant equations: $c_i \approx c_j$.

Note that each fresh constant corresponds to a shared label for a syntactically equivalent subterm defined by one of the term equations.

Congruence Closure

The congruence closure algorithm is presented as a set of abstract rewrite rules operating on a pair of equations E and a set of rules R, $(E; R)$.

$f(a_1, \ldots, a_n) \approx c$

$f(a_1, \ldots, a_n) \longrightarrow c$

$$(E_0; R_0) \Rightarrow_{\mathsf{CC}} (E_1; R_1) \Rightarrow_{\mathsf{CC}} (E_2; R_2) \Rightarrow_{\mathsf{CC}} \ldots$$

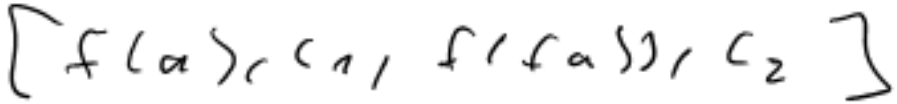
At the beginning, $E = E_0$ is a set of constant equations and $R = R_0$ is the set of term equations oriented from left-to-right. At termination, E is empty and R contains the result.

This means we split our flattened set of equations E' into E containing all constant equations of E, R containing all term equations oriented as rules; the inequations will be ignored for now.

Simplify $(E \uplus \{c \stackrel{\cdot}{\approx} c'\}; R \uplus \{c \to c''\}) \Rightarrow_{\mathsf{CC}} (E \cup \{c'' \stackrel{\cdot}{\approx} c'\}; R \cup \{c \to c''\})$

Delete $(E \uplus \{c \approx c\}; R) \Rightarrow_{\mathsf{CC}} (E; R)$

Orient $(E \uplus \{c \stackrel{\cdot}{\approx} c'\}; R) \Rightarrow_{\mathsf{CC}} (E; R \cup \{c \to c'\})$
if $c \succ c'$



Deduce $(E; R \uplus \{t \to c,\ t \to c'\}) \Rightarrow_{\mathsf{CC}}$
$(E \cup \{c \approx c'\}; R \cup \{t \to c\})$

Collapse $(E; R \uplus \{t[c]_p \to c',\ c \to c''\}) \Rightarrow_{\mathsf{CC}}$
$(E; R \cup \{t[c'']_p \to c',\ c \to c''\})$

$p \neq \epsilon$

For rule Deduce, t is either a term of the form $f(c_1, \ldots, c_n)$ or a constant c_i.
For rule Collapse, t is always of the form $f(c_1, \ldots, c_n)$

Usual Strategy: Simplify, Delete and Orient are preferred over Deduce and Collapse. Then if Collapse becomes applicable, it is applied exhaustively.

Implementation Recommendations:

Instead of fixing the ordering $\prec$ in advance, it is preferable to define it on the fly during the algorithm: if an equation $c \approx c'$ between two constants is oriented, a good heuristic is to make that constant symbol larger that occurs less often in R, hence producing afterwards fewer Collapse steps.

Represent terms as DAGs (directed acyclic graphs). Then it is not necessary to introduce fresh constants (explicitly).

Lemma (CC Termination)

CC always terminates and always in a state of the form $(\emptyset; R)$.

With the right tricks, the algorithm has worst-case run time complexity O(m log m), where m is the number of edges in the graph representation of the initial constant and term equations.

Lemma (CC Soundness)

Let E' be a set of flattened equations, where E_0 are all constant equations of E' and R_0 contains all term equations oriented as rules. Let $(E_0; R_0) \Rightarrow_{\mathsf{CC}} \ldots \Rightarrow_{\mathsf{CC}} (E_n; R_n)$. Then $E' \models s_i \approx t_i$ if $s_i \approx t_i \in E_n$ or $s_i \to t_i \in R_n$.

Idea: The conclusions are entailed by the premises, so every model of the premises is a model of the conclusions.

Lemma (CC Completeness)

Let E' be a set of flattened equations, where E_0 are all constant equations of E' and R_0 contains all term equations oriented as rules. Let $(\emptyset; R)$ be the end state of a terminating CC run with start state $(E_0; R_0)$. Let s and t be two terms. Then $s{\downarrow_R} = t{\downarrow_R}$ if $E' \models s \approx t$. $\not\leftrightarrow$ $E' \wedge s \not\approx t \vDash \bot$

Combining Theories

Here I discuss a basic variant of the Nelson-Oppen (NO) combination procedure for two theories $\mathcal{T}_1$ and $\mathcal{T}_2$ over two respective signatures Σ_1 and Σ_2 that do not share any function, constant, or predicate symbols, but may share sorts. The idea of the procedure is to reduce satisfiability of a quantifier-free formula over $\Sigma_1 \cup \Sigma_2$ to satisfiability of two separate formulas over Σ_1 and Σ_2, respectively.

Stably Infinite

There are several properties needed for the Nelson-Oppen procedure to work. One of them is that the theory models always include models with an infinite domain. Consider the two theories

$$\mathcal{T}_1 = \{\forall x.(x \approx a \vee x \approx b)\}$$

and

$$\mathcal{T}_2 = \{\forall x.\exists y, z.(x \not\approx y \wedge x \not\approx z \wedge y \not\approx z)\}$$

that do not share any signature symbols. Models of $\mathcal{T}_1$ have at most two elements, models of $\mathcal{T}_2$ at least three. So the conjunction $(\mathcal{T}_1 \cup \mathcal{T}_2)$ is already unsatisfiable. In order to ensure that different models for the respective theory can be combined, the Nelson-Oppen procedure requires the existence of models with infinite cardinality.

Stably Infinite

7.1.2 Definition (Stably-Infinite Theory)

A theory $\mathcal{T}$ is *stably-infinite* if for every quantifier-free formula ϕ, if $\mathcal{T} \models \phi$, then then there exists also a model $\mathcal{A}$ of infinite cardinality, such that $\mathcal{A} \models_{\mathcal{T}} \phi$

Convex Theory

7.1.1 Definition (Convex Theory)

A theory $\mathcal{T}$ is *convex* if for a conjunction ϕ of literals with $\phi \models_{\mathcal{T}} x_1 \approx y_1 \vee \ldots \vee x_n \approx y_n$ then $\phi \models_{\mathcal{T}} x_k \approx y_k$ for some k.

Holds for EUF and LRA, but not LIA!

Example:

$$1 < x_{\text{LIA}} \wedge x_{\text{LIA}} < 4 \models_{\text{LIA}} x_{\text{LIA}} = 2 \vee x_{\text{LIA}} = 3$$

but none of the two single disjuncts is a consequence.

Nelson-Oppen Combination

7.1.3 Definition (Nelson-Oppen Basic Restrictions)

Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two theories. Then the *Nelson-Oppen Basic Restrictions* are:

(i) There are decision procedures for $\mathcal{T}_1$ and $\mathcal{T}_2$.

(ii) Each decision procedure returns a complete set of variable identities as consequence of a formula. $\phi \models_{\mathcal{T}_i} x_j \approx x_k$

(iii) $\Sigma_1 \cap \Sigma_2 = \emptyset$ except for common sorts.

(iv) Both theories are convex.

(v) $\mathcal{T}_1$ and $\mathcal{T}_2$ are stably-infinite.

$\{x \approx y, y \approx x\} \models_{CRA} x \approx y$

$\{x \approx 1, y \approx 1\} \models_{CRA} x \approx y$

Actually, restriction 7.1.3-2 is not needed, because a given finite quantifier-free formula ϕ over $\Sigma_1 \cup \Sigma_2$ contains only finitely many different variables. Now instead of putting the burden to identify variables on the decision procedure, all potential variable identifications can be guessed and tested afterwards. The disadvantage of this approach is, of course, that there are exponentially many identifications with respect to a fixed number of variables. Therefore, assuming 7.1.3-2 results in a more efficient procedure and is also supported by many procedures from Section 6.

Restriction 7.1.3-5 can be further relaxed to assume that the domains of all shared sorts of all models are either infinite or have the same number of elements.

Purification

Purify $N \uplus \{L[t[s]_i]_p\} \Rightarrow_{\mathsf{NO}} N \uplus \{L[t[z]_i]_p, z \approx s\}$

if $t = f(t_1, \ldots, t_n)$, $s = h(s_1, \ldots, s_m)$, the function symbols f and h are from different signatures, $1 \leq i \leq n$, (i.e., $t_i = s$) and z is a fresh variable of appropriate sort

$\tau_1 = \mathrm{LRA} \qquad \tau_2 = \mathrm{EUF} \qquad \Sigma_2 = \{a, g, f\}$

$f(x_1, 0) \geq x_3$

$\Downarrow_{\mathsf{NO}}$

$x_4 \geq x_3 \wedge f(x_1, 0) \approx x_4$

$\Downarrow_{\mathsf{Ni}}$

$x_4 \geq x_3 \wedge f(x_1, x_5) \approx x_4 \wedge x_5 \approx 0$

Nelson-Oppen Calculus

Now a Nelson-Oppen problem state is a five tuple (N_1, E_1, N_2, E_2, s) with $s \in \{\top, \bot, \text{fail}\}$, the sets E_1 and E_2 contain variable equations, and N_1, N_2 literals over the respective signatures, where

$(N_1; \emptyset; N_2; \emptyset; \bot)$	is the start state for some purified set of atoms $N = N_1 \cup N_2$ where the N_i are built from the respective signatures only
$(N_1; E_1; N_2; E_2; \text{fail})$	is a final state, where $N_1 \cup N_2 \cup E_1 \cup E_2$ is unsatisfiable
$(N_1; E_1; N_2; E_2; \bot)$	is an intermediate state, where $N_1 \cup E_2$ and $N_2 \cup E_1$ have to be checked for satisfiability
$(N_1; \emptyset; N_2; \emptyset; \top)$	is a final state, where $N_1 \cup N_2$ is satisfiable

Solve $(N_1; E_1; N_2; E_2; \bot) \Rightarrow_{\mathsf{NO}} (N_1'; E_1'; N_2'; E_2'; \bot)$
if $N_1' = N_1 \cup E_1 \cup E_2$ and $N_2' = N_2 \cup E_1 \cup E_2$ are both $\mathcal{T}_i$-satisfiable, respectively, E_1' are all new variable equations derivable from N_1', E_2' are all new variable equations derivable from N_2' and $E_1' \cup E_2' \neq \emptyset$

Success $(N_1; E_1; N_2; E_2; \bot) \Rightarrow_{\mathsf{NO}} (N_1'; \emptyset; N_2'; \emptyset; \top)$
if $N_1' = N_1 \cup E_1 \cup E_2$ and $N_2' = N_2 \cup E_1 \cup E_2$ are both $\mathcal{T}_i$-satisfiable, respectively, E_1' are all new variable equations derivable from N_1', E_2' are all new variable equations derivable from N_2' and $E_1' \cup E_2' = \emptyset$

Fail $(N_1; E_1; N_2; E_2; \bot) \Rightarrow_{\mathsf{NO}} (N_1; E_1; N_2; E_2; \mathsf{fail})$
if $N_1' = N_1 \cup E_1 \cup E_2$ or $N_2' = N_2 \cup E_1 \cup E_2$ is $\mathcal{T}_i$-unsatisfiable, respectively

7.1.6 Definition (Arrangement)

Given a (finite) set of parameters X, an *arrangement* A over X is a (finite) set of equalities and inequalities over X such that for all $x_1, x_2 \in X$ either $x_1 \approx x_2 \in A$ or $x_1 \not\approx x_2 \in A$.

7.1.7 Proposition (Nelson-Oppen modulo Arrangement)

Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two theories satisfying the restrictions of Definition 7.1.3 except for restriction 2. Let ϕ be a conjunction of literals over $\Sigma_1 \cup \Sigma_2$. Let N_1 and N_2 be the purified literal sets out of ϕ. Then ϕ is satisfiable iff there is an arrangement A over $\mathsf{vars}(\phi)$ such that $N_1 \cup A$ is $\mathcal{T}_1$-satisfiable and $N_2 \cup A$ is $\mathcal{T}_2$-satisfiable.

7.1.8 Theorem (Nelson-Oppen is Sound, Complete and Terminating)

Let $\mathcal{T}_1, \mathcal{T}_2$ be two theories satisfying the Nelson-Oppen basic restrictions. Let ϕ be a conjunction of literals over $\Sigma_1 \cup \Sigma_2$ and N_1, N_2 be the result of purifying ϕ.
(i) All sequences $(N_1; \emptyset; N_2; \emptyset; \bot) \Rightarrow_{\text{NO}}^{*} \ldots$ are finite.
Let $(N_1; \emptyset; N_2; \emptyset; \bot) \Rightarrow_{\text{NO}}^{*} (N_1; E_1; N_2; E_2; s)$ be a derivation with finite state $(N_1; E_1; N_2; E_2; s)$,
(ii) If $s = \text{fail}$ then ϕ is unsatisfiable in $\mathcal{T}_1 \cup \mathcal{T}_2$.
(iii) If $s = \top$ then ϕ is satisfiable in $\mathcal{T}_1 \cup \mathcal{T}_2$.