Seminar Decision Procedures and Applications

Background Informations

Viorica Sofronie-Stokkermans
University Koblenz-Landau
Equality is the most important relation in mathematics and functional programming.

In principle, problems in first-order logic with equality can be handled by, e.g., resolution theorem provers.

Equality is theoretically difficult: First-order functional programming is Turing-complete.

But: resolution theorem provers cannot even solve problems that are intuitively easy.

Consequence: to handle equality efficiently, knowledge must be integrated into the theorem prover.
Handling Equality Naively

Proposition 1:
Let $F$ be a closed first-order formula with equality. Let $\sim \notin \Pi$ be a new predicate symbol. The set $Eq(\Sigma)$ contains the formulas

$$
\forall x \ (x \sim x)
\forall x, y \ (x \sim y \rightarrow y \sim x)
\forall x, y, z \ (x \sim y \land y \sim z \rightarrow x \sim z)
\forall \vec{x}, \vec{y} \ (x_1 \sim y_1 \land \cdots \land x_n \sim y_n \rightarrow f(x_1, \ldots, x_n) \sim f(y_1, \ldots, y_n))
\forall \vec{x}, \vec{y} \ (x_1 \sim y_1 \land \cdots \land x_n \sim y_n \land p(x_1, \ldots, x_n) \rightarrow p(y_1, \ldots, y_n))
$$

for every $f/n \in \Omega$ and $p/n \in \Pi$. Let $\tilde{F}$ be the formula that one obtains from $F$ if every occurrence of $\approx$ is replaced by $\sim$. Then $F$ is satisfiable if and only if $Eq(\Sigma) \cup \{\tilde{F}\}$ is satisfiable.
Handling Equality Naively

By giving the equality axioms explicitly, first-order problems with equality can in principle be solved by a standard resolution or tableaux prover.

But this is unfortunately not efficient (mainly due to the transitivity and congruence axioms).
How to proceed:

- Arbitrary binary relations.
- Equations (unit clauses with equality):
  - Term rewrite systems.
  - Expressing semantic consequence syntactically.
  - Entailment for equations.
- Equational clauses:
  - Entailment for clauses with equality.
Roadmap

How to proceed:

- Arbitrary binary relations.

- Equations (unit clauses with equality):
  - Term rewrite systems.
  - Expressing semantic consequence syntactically.
  - Entailment for equations.

- Equational clauses:
  - Entailment for clauses with equality.
Abstract Reduction Systems

Abstract reduction system: \((A, \rightarrow)\), where

- \(A\) is a set,
- \(\rightarrow \subseteq A \times A\) is a binary relation on \(A\).
Abstract Reduction Systems

\[ \rightarrow^0 = \{(x, x) \mid x \in A\} \quad \text{identity} \]
\[ \rightarrow^{i+1} = \rightarrow^i \circ \rightarrow \quad \text{i + 1-fold composition} \]
\[ \rightarrow^+ = \bigcup_{i>0} \rightarrow^i \quad \text{transitive closure} \]
\[ \rightarrow^* = \bigcup_{i \geq 0} \rightarrow^i = \rightarrow^+ \cup \rightarrow^0 \quad \text{reflexive transitive closure} \]
\[ \rightarrow^= = \rightarrow \cup \rightarrow^0 \quad \text{reflexive closure} \]
\[ \rightarrow^{-1} = \leftarrow = \{(x, y) \mid y \rightarrow x\} \quad \text{inverse} \]
\[ \leftrightarrow = \rightarrow \cup \leftarrow \quad \text{symmetric closure} \]
\[ \leftrightarrow^+ = (\leftrightarrow)^+ \quad \text{transitive symmetric closure} \]
\[ \leftrightarrow^* = (\leftrightarrow)^* \quad \text{refl. trans. symmetric closure} \]
Abstract Reduction Systems

\[ x \in A \text{ is reducible, if there is a } y \text{ such that } x \rightarrow y. \]

\[ x \text{ is in normal form (irreducible), if it is not reducible.} \]

\[ y \text{ is a normal form of } x, \text{ if } x \rightarrow^* y \text{ and } y \text{ is in normal form.} \]
\[ \text{Notation: } y = x \downarrow \text{ (if the normal form of } x \text{ is unique).} \]

\[ x \text{ and } y \text{ are joinable, if there is a } z \text{ such that } x \rightarrow^* z \leftarrow^* y. \]
\[ \text{Notation: } x \downarrow y. \]
A relation $\rightarrow$ is called

Church-Rosser, if $x \leftrightarrow^* y$ implies $x \downarrow y$.

confluent, if $x \leftarrow^* z \rightarrow^* y$ implies $x \downarrow y$.

locally confluent, if $x \leftarrow z \rightarrow y$ implies $x \downarrow y$.

terminating, if there is no infinite decreasing chain $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots$.

normalizing, if every $x \in A$ has a normal form.

convergent, if it is confluent and terminating.
Abstract Reduction Systems

**Theorem 2:** The following properties are equivalent:

(i) \( \rightarrow \) has the Church-Rosser property (\( x \leftrightarrow^* y \) implies \( x \downarrow y \))

(ii) \( \rightarrow \) is confluent (\( x \leftarrow^* z \rightarrow^* y \) implies \( x \downarrow y \))

Proof:
(i) \( \Rightarrow \) (ii): trivial.

(ii) \( \Rightarrow \) (i): by induction on the number of peaks in the derivation \( x \leftrightarrow^* y \).
**Abstract Reduction Systems**

**Lemma 3:** If $\rightarrow$ is terminating, then it is normalizing.

Note: The reverse implication does not hold.

**Lemma 4:**
If $\rightarrow$ is confluent, then every element has at most one normal form.

**Corollary 5:**
If $\rightarrow$ is normalizing and confluent, then every element $x$ has a unique normal form.

**Proposition 6:**
If $\rightarrow$ is normalizing and confluent, then $x \leftrightarrow^* y$ if and only if $x \downarrow = y \downarrow$.
Well-Founded Orderings

**Lemma 7:**

If $\rightarrow$ is a terminating binary relation over $A$, then $\rightarrow^+$ is a well-founded partial ordering.

**Lemma 8:**

If $>$ is a well-founded partial ordering and $\rightarrow \subseteq >$, then $\rightarrow$ is terminating.
Theorem 9 (“Newman’s Lemma”):
If a terminating relation $\rightarrow$ is locally confluent ($x \leftarrow z \rightarrow y$ implies $x \downarrow y$), then it is confluent ($x \leftarrow^* z \rightarrow^* y$ implies $x \downarrow y$).

Proof:
Let $\rightarrow$ be a terminating and locally confluent relation. Then $\rightarrow^+$ is a well-founded ordering. Define $P(z) \iff (\forall x, y : x \leftarrow^* z \rightarrow^* y \Rightarrow x \downarrow y)$.
Prove $P(z)$ for all $z \in A$ by well-founded induction over $\rightarrow^+$:
Case 1: $x \leftarrow^0 z \rightarrow^* y$: trivial.
Case 2: $x \leftarrow^* z \rightarrow^0 y$: trivial.
Case 3: $x \leftarrow^* x' \leftarrow z \rightarrow y' \rightarrow^* y$: use local confluence, then use the induction hypothesis.
Rewrite Systems

**Notation:**

**Positions** of a term $s$:

- $\text{Pos}(x) = \{\varepsilon\}$,
- $\text{Pos}(f(s_1, \ldots, s_n)) = \{\varepsilon\} \cup \bigcup_{i=1}^{n}\{ ip \mid p \in \text{Pos}(s_i) \}$.

**Size** of a term $s$: $|s| = \text{cardinality of Pos}(s)$.

**Subterm** of $s$ at a position $p \in \text{Pos}(s)$:

- $s/\varepsilon = s$,
- $f(s_1, \ldots, s_n)/ip = s_i/p$.

**Replacement** of the subterm at position $p \in \text{Pos}(s)$ by $t$:

- $s[t]_{\varepsilon} = t$,
- $f(s_1, \ldots, s_n)[t]_{ip} = f(s_1, \ldots, s_i[t]_p, \ldots, s_n)$. 
Let $E$ be a set of equations.

The rewrite relation $\rightarrow_E \subseteq T_\Sigma(X) \times T_\Sigma(X)$ is defined by

$$s \rightarrow_E t \text{ iff there exist } (l \approx r) \in E, p \in \text{Pos}(s),$$

and $\sigma : X \rightarrow T_\Sigma(X)$,

such that $s/p = l\sigma$ and $t = s[r\sigma]_p$.

An equation $l \approx r$ is also called a rewrite rule, if $l$ is not a variable and $\text{Var}(l) \supseteq \text{Var}(r)$.

Notation: $l \rightarrow r$.

A set of rewrite rules is called a term rewrite system (TRS).
We say that a set of equations $E$ or a TRS $R$ is terminating, if the rewrite relation $\rightarrow_E$ or $\rightarrow_R$ has this property.

(Analogously for other properties of abstract reduction systems).

Note: If $E$ is terminating, then it is a TRS.
Corollary 10:
If $E$ is convergent (i.e., terminating and confluent),
then $s \approx_E t$ if and only if $s \leftrightarrow_E^* t$ if and only if $s\downarrow_E = t\downarrow_E$.

Corollary 11:
If $E$ is finite and convergent, then $\approx_E$ is decidable.

Reminder:
If $E$ is terminating, then it is confluent if and only if it is locally confluent.
Rewrite Relations

Problems:

Show local confluence of $E$.

Show termination of $E$.

Transform $E$ into an equivalent set of equations that is locally confluent and terminating.

talk in this seminar: ground TRS (left and right hand side are ground terms)

Simple form: $f(c_1, \ldots, c_n) \rightarrow c$ or $c \rightarrow d$
Critical Pairs

Showing local confluence (Sketch for ground TRS):

Question:
Are there rewrite rules \( l_1 \rightarrow r_1 \) and \( l_2 \rightarrow r_2 \) such that some subterm \( l_1/p \) and \( l_2 \) are equal?

Let \( l_i \rightarrow r_i \ (i = 1, 2) \) be two rewrite rules in a TRS \( R \)
Let \( p \in \text{Pos}(l_1) \) be a position such that \( l_1/p = l_2 \).

Then \( r_1 \leftarrow l_1 \rightarrow (l_1)[r_2]_p \).

\( \langle r_1, (l_1)[r_2]_p \rangle \) is called a critical pair of \( R \).

The critical pair is joinable (or: converges), if \( r_1 \downarrow_R (l_1)[r_2]_p \).
Theorem 12 ("Critical Pair Theorem"): A TRS $R$ is locally confluent if and only if all its critical pairs are joinable.

Proof (Here only for the case of ground TRS):
"only if": obvious, since joinability of a critical pair is a special case of local confluence.

"if": Suppose $s$ rewrites to $t_1$ and $t_2$ using rewrite rules $l_i \rightarrow r_i \in R$ at positions $p_i \in \text{Pos}(s)$, where $i = 1, 2$.
Then $s/p_i = l_i$ and $t_i = s[r_i]_{p_i}$.

We distinguish between two cases: Either $p_1$ and $p_2$ are in disjoint subtrees ($p_1 \parallel p_2$), or one is a prefix of the other (w.l.o.g., $p_1 \leq p_2$).
Critical Pairs

Case 1: \( p_1 \parallel p_2 \).

Then \( s = s[l_1]p_1[l_2]p_2 \),
and therefore \( t_1 = s[r_1]p_1[l_2]p_2 \) and \( t_2 = s[l_1]p_1[r_2]p_2 \).

Let \( t_0 = s[r_1]p_1[r_2]p_2 \).
Then clearly \( t_1 \rightarrow_R t_0 \) using \( l_2 \rightarrow r_2 \) and \( t_2 \rightarrow_R t_0 \) using \( l_1 \rightarrow r_1 \).

Case 2: \( p_1 \leq p_2 \).

Then \( s/p_2 = l_2 \) and \( s/p_2 = (s/p_1)/p = l_1/p \); hence \( l_2 = l_1/p \); and
\( \langle r_1, (l_1)[r_2]_p \rangle \) is a critical pair.

By assumption, it is joinable, so \( r_1 \rightarrow_R^* v \leftarrow_R^* (l_1)[r_2]_p \).

Consequently, \( t_1 = s[r_1]p_1 = s[r_1]p_1 \rightarrow_R^* s[v]p_1 \) and
\( t_2 = s[r_2]p_2 = s[(l_1)[r_2]_p]p_1 = s[(l_1)[r_2]_p]p_1 = s[((l_1)[r_2]_p)]p_1 \rightarrow_R^* s[v]p_1 \).

This completes the proof of the Critical Pair Theorem.
Critical Pairs

Note: Critical pairs between a rule and (a renamed variant of) itself must be considered – except if the overlap is at the root (i.e., $p = \varepsilon$).
Critical Pairs

Corollary 13:
A terminating TRS $R$ is confluent if and only if all its critical pairs are joinable.

Proof:
By Newman’s Lemma and the Critical Pair Theorem.
Critical Pairs

**Corollary 14:**
For a finite terminating TRS, confluence is decidable.

**Proof:**
For every pair of rules and every non-variable position in the first rule there is at most one critical pair \( \langle u_1, u_2 \rangle \).

Reduce every \( u_i \) to some normal form \( u'_i \). If \( u'_1 = u'_2 \) for every critical pair, then \( R \) is confluent, otherwise there is some non-confluent situation

\[
\begin{align*}
  u'_1 & \xrightarrow{R}^* u_1 \xleftarrow{R} s \rightarrow_R u_2 \rightarrow_R^* u'_2.
\end{align*}
\]