Reasoning and Solving
Modulo (Intruder) Theories

Christophe Ringeissen

INRIA Nancy - Grand Est & LORIA

Pesto Seminar, January 20, 2023
1 Reasoning and Solving

2 Knowledge in Subterm Modulo

3 Knowledge in Beyond Subterm
Outline

1 Reasoning and Solving

2 Knowledge in Subterm Modulo

3 Knowledge in Beyond Subterm
Research Interests

• First-Order Logic with Equality
• Automated reasoning, e.g., equational theorem proving
• Satisfiability Modulo Theories (SMT)
• Equational theories, e.g., intruder theories
• Constraint solving, e.g., (dis)unification/matching
• Decision procedures for equational reasoning, e.g., the ones for the deduction and the static equivalence
• Rewriting techniques
• Declarative programming, e.g., rule-based programming and constraint programming
• and last but not least ... Combination of reasoners/solvers/procedures for unions of theories
Combination Problem

A general issue: Given reasoners/solvers known for single theories $T_1$ and $T_2$, how to build a reasoner/solver for the union of theories $T_1 \cup T_2$?

Why? Because a problem is usually expressed using several theories.

Theories are usually assumed to be signature-disjoint, the equality being the only shared symbol.

Well-known combination methods (disjoint case):

- Unification: Schmidt-Schauss
- Matching: Nipkow
- (Dis)unifiability: Baader-Schulz
- Satisfiability Modulo Theories: Nelson-Oppen
- Deduction and Static Equivalence: Cortier-Delaune
Combination Method: Disjoint Case

Satisfiability Modulo Theories [Nelson and Oppen, 1979]:

Nelson-Oppen combination method is sound but not always complete. To get completeness, assuming *stably infinite* theories is the usual way, but it is restrictive...

Research directions: go beyond stable infiniteness via *politeness*

- A polite theory is combinable with any disjoint theory
- A theory modeling a data structure should be polite [Chocron et al., 2020, Sheng et al., 2021, Sheng et al., 2022]
- Rewrite-based satisfiability procedures to show *politeness*
- Satisfiability procedures based on congruence closure methods (with Laurent Vigneron)
Combination Method: Non-Disjoint Case

Satisfiability Modulo Theories [Ghilardi, 2004]:

It provides a combination method à la Nelson-Oppen for which completeness is based on a model-theoretical framework introducing the notion of $T_0$-compatibility.

An alternative to non-disjoint combination: consider shared constructor symbols modulo an equational theory $E$, e.g.,

$$E = AC(+) = \{(x + y) + z = x + (y + z), x + y = y + x\}$$

Remark: $AC(+) \text{ is an example of a } \text{permutative} \text{ theory } E$, i.e., for any $l = r \in E$ and any (variable/function) symbol $s$, the number of occurrences of $s$ in $l$ is equal to the one in $r$
E-Constructed Theories

A theory $F$ is $E$-constructed if there exists a normalizing mapping $NF$ satisfying some properties including

$$s =_{F \cup E} t \text{ iff } NF(s) =_E NF(t)$$

and for any function symbol $f$ in $E$,

$$NF(f(t_1, \ldots, t_n)) =_E f(NF(t_1), \ldots, NF(t_n))$$

Consequence: $F \cup E$-equality is decidable if $NF$ is computable and $E$-equality is decidable.

Remark: the definition of an $E$-constructed theory does not require that $NF$ is computable.

Result [Erbatur et al., 2022]: the class of $E$-constructed theories is closed by union sharing only the symbols in $E$. 
**E-Constructed Theories: Examples**

- **Pairing**

  \[
  R_P = \{ \begin{align*}
  \text{fst}(p(x, y)) & \rightarrow x \\
  \text{snd}(p(x, y)) & \rightarrow y
  \end{align*} \}
  \]

  \( (R_P, \emptyset) \) is \( \emptyset \)-constructed, \( \emptyset \) being the empty theory over the binary symbol \( p \)

- **Key Exchange**

  \[
  K = \{ \text{keyex}(x, pk(u), y, pk(v)) = \text{keyex}(u, pk(x), v, pk(y)) \}
  \]

  \( K \) is \( \emptyset \)-constructed, \( \emptyset \) being the empty theory over the unary symbol \( pk \)

- **Distributive Exponentiation**

  \[
  R_E = \{ \begin{align*}
  \text{exp}(\text{exp}(x, y), z) & \rightarrow \text{exp}(x, y \otimes z) \\
  \text{exp}(x \ast y, z) & \rightarrow \text{exp}(x, z) \ast \text{exp}(y, z)
  \end{align*} \}
  \]

  \[
  R_F = \{ \text{enc}(\text{enc}(x, y), z) \rightarrow \text{enc}(x, y \oslash z) \}
  \]

  \( (R_E, AC) \) and \( (R_F, AC) \) are \( AC \)-constructed

  for \( AC = AC(\oslash) \)
Hierarchical Solvers

A hierarchical solver $H_E(U)$ for $F \cup E$ is given by:

1. some fixed combination rules, to transform the input into a separate form $\Gamma \cup \Gamma_0$ such that
   - $\Gamma_0$ is built over symbols in $E$
   - $\Gamma$ is built over symbols not in $E$

2. a Solve algorithm to solve $\Gamma_0$ modulo $E$,

3. an additional inference system $U$ to simplify $\Gamma$ modulo $F \cup E$.
Syntactic Theories

A class of theories initially studied by Kirchner, Klay, Nipkow, Jouannaud, Comon, ...

In a syntactic theory, there exists a finite set $U$ of mutation rules such that $U$ plus the classical syntactic decomposition rule is sound and complete to simplify equations.

Example: Commutativity ($+$)

$x + y = u + v \vdash (x = u, y = v) \lor (x = v, y = u)$

Other Examples:

- Shallow theories (any variable occurs at depth at most 1 in any axiom)
- Associativity-Commutativity
- Distributive exponentiation
- Theories with the Finite Variant Property, including subterm convergent Term Rewrite Systems (TRSs)
Combined Hierarchical Unification

Individual theory: if \( F \cup E \) is syntactic and \( F \) is \( E \)-constructed, then \( F \cup E \) admits a hierarchical unification procedure \( H_E(U) \).

Union of theories: Given a hierarchical unification procedure \( H_E(U_i) \) for \( F_i \cup E \) and any \( i = 1, 2 \), under which conditions do we have that \( H_E(U_1 \cup U_2) \) is a hierarchical unification procedure for \( F_1 \cup F_2 \cup E \)?

Problem considered in several recent papers:

- Terminating hierarchical unification procedures [Erbatur et al., 2020b]
- Hierarchical unification for theories closed by equational paramodulation [Erbatur et al., 2021]
- Hierarchical matching [Erbatur et al., 2022]
Reasoning and Solving Tools

- **UNIF**: a solver implementing several equational unification algorithms, developed by M. Adi (1989-), with a focus on \( AC \)-unification
- **ELAN**: a rewrite engine for efficient equational rewriting, developed in the Protheo group (1992-), with a focus on \( AC \)-rewriting and similar to Maude
- **TOM**: a matching engine embedded into an imperative programming language (C/Java), developed in the follow-up of Protheo (1999-)
- **haRVey**: a SMT solver implementing rewrite-based satisfiability procedures, developed by S. Ranise and D. Déharbe (2002-)
Outline

1. Reasoning and Solving
2. Knowledge in Subterm Modulo
3. Knowledge in Beyond Subterm
Two Notions of Knowledge

Two decision problems used to express the knowledge modulo an equational theory

1. Deduction: given a sequence of messages $S$ and a message $M$, can we deduce/compute $M$ from $S$?

   ➤ Example: a secret $m$ can be deduced from the messages $X = enc(m, k)$ and $Y = k$ by considering $dec(X, Y)$ and the axiom $dec(enc(V, K), K) = V$.

2. Static Equivalence: given two sequences of messages $S_1$ and $S_2$, can we distinguish an instance of a protocol running $S_1$ from one running $S_2$?

   ➤ important for voting protocols.

Both problems are static: only messages are considered, without taking into account the processes that generate them.
Proof System for the Deduction

Remark: The following inference system generates all the terms deducible from $\phi$, but it does not provide a decision procedure...

\[
\begin{align*}
\frac{\nu \tilde{n}.\sigma \vdash_E M}{\exists x \in \text{Dom}(\sigma) \text{ s.t. } x\sigma = M} \\
\frac{\nu \tilde{n}.\sigma \vdash_E s}{\text{if } s \not\in \tilde{n}} \\
\frac{\phi \vdash_E M_1, \ldots, \phi \vdash_E M_k}{\phi \vdash_E f(M_1, \ldots, M_k)} \text{ if } f \in \Sigma \\
\frac{\phi \vdash_E M}{\phi \vdash_E M'} \text{ if } M =_E M'
\end{align*}
\]

Figure: Deduction Axioms
Knowledge Decidability

Undecidable in general, but critical to the analysis of security protocols. However, decision procedures are known for particular theories

- Subterm convergent theories
- Theories of Homomorphism
- Blind signatures
- Trap-door commitments
- Malleable encryption
- and more
Computing Knowledge in Combined Theories

Decision procedures have already been developed for the two notions of knowledge in combined theories $F \cup E$

- $F$ and $E$ are signature disjoint [Cortier and Delaune, 2010]
- $F$ and $E$ share only constructors modulo the empty theory [Erbatur et al., 2017]
- some particular theories $F \cup E$ where $E$ is the empty theory or AC [Abadi and Cortier, 2006]
- $F$ is given by a subterm $E$-convergent TRS where $E$ is syntactic permutative [Erbatur et al., 2020a]
Subterm Equational Convergent TRS

**Definition:** A *subterm* $E$-convergent TRS is a TRS such that $\rightarrow_{R,E}$ is convergent modulo $E$ and for any $l \rightarrow r$ in $R$, $r$ is a strict subterm of $l$ or a ground constant.

**Example:** Abelian Pre Group

\[
APG = \left\{ \begin{array}{l} 
x \ast e \rightarrow x \\
x \ast i(x) \rightarrow e \\
i(i(x)) \rightarrow x \\
i(e) \rightarrow e \\
\end{array} \right\} \cup \{x \ast y = y \ast x\}
\]

$APG$-unification successfully studied in [Yang et al., 2014] using a variant-based approach.

What about the deduction and static equivalence in $APG$?
Knowledge in Subterm Modulo Shallow Permutative Theories

Decision procedures for the two notions of knowledge in combined theories $RE = R \cup E$, where

- $R$ is a subterm $E$-convergent TRS
- $E$ is shallow permutative, e.g., $C$ (Commutativity) via Reduction Lemmas to the empty theory. These reductions hold since $E$ is shallow.

See [Erbatur et al., 2020a] for more details
Knowledge in Subterm Modulo
Syntactic Permutative Theories

Decision procedures for the two notions of knowledge in combined theories $RE = R \cup E$, where

- $R$ is a subterm $E$-convergent TRS
- $E$ is syntactic permutative and the size of $R$ modulo $E$ is computable via Reduction Lemmas to $E$ instead of the empty theory used for the shallow permutative case

See [Erbatur et al., 2020a] for more details
1. Reasoning and Solving

2. Knowledge in Subterm Modulo

3. Knowledge in Beyond Subterm
Beyond Subterm

The procedures developed for the knowledge problems have been proven to work for the class of subterm convergent theories.

Many of these same procedures also work for theories that are beyond subterm.

However, since these examples don’t fit into a known class of theories for which soundness and completeness proofs already exist, they must be proven on an individual basis.
Beyond Subterm: Example

For example, the procedures of [Abadi and Cortier, 2006, Ștefan Ciobâcă et al., 2012] are shown to work on the theory of blind signatures:

**Subterm:**

\[
\text{checksign}(\text{sign}(x, y), \text{pk}(y)) \rightarrow x, \\
\text{unblind}(\text{blind}(x, y), y) \rightarrow x, 
\]

**Non-subterm:**

\[
\text{unblind}(\text{sign}(\text{blind}(x, y), z), y) \rightarrow \text{sign}(x, z) 
\]
Can we develop a, hopefully simple, definition that extends the subterm convergent definition and encompasses the “beyond subterm” examples?

Joint work with Saraid Dwyer Satterfield (UMW), Serdar Erbatur (UT Dallas), Andrew Marshall (UMW), presented at the UNIF 2022 workshop
Graph-embedding

We define, $\rightarrow_{R_{gemb}}^*$, to be the reduction relation induced by the set of rewrite rules created after instantiating the following rule schema, $R_{gemb}$, with $\Sigma$:

For any $f \in \Sigma$

(1) $f(x_1, \ldots, x_n) \rightarrow x_i$

(2) $f(x_1, \ldots, x_{i-1}, x_i, x_{i+1} \ldots, x_n) \rightarrow f(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$

For any $f, g \in \Sigma$

(3) $f(x_1, \ldots, x_{i-1}, g(\bar{z}), x_{i+1} \ldots, x_m) \rightarrow g(x_1, \ldots, x_{i-1}, \bar{z}, x_{i+1}, \ldots, x_m)$

(4) $f(x_1, \ldots, x_{i-1}, g(\bar{z}), x_{i+1}, \ldots, x_m) \rightarrow f(x_1, \ldots, x_{i-1}, \bar{z}, x_{i+1}, \ldots, x_m)$
Graph-embedded Systems

We say a term $t'$ is graph embedded in a term $t$, denoted $t' \geq_{gemb} t$, if $t'$ is a well-formed term and $t \rightarrow^{*}_{R_{gemb}} s \approx t'$ for some well-formed term $s$.

- $s \approx t'$ represent equivalence modulo an appropriate form of permutation (extending leaf permutation)

A TRS $R$ is graph-embedded if for any $l \rightarrow r \in R$, $r \geq_{gemb} l$. 
Example: Malleable Encryption

Theory of malleable encryption is defined by $R_{mal}$:

\[ dec(enc(x, y), y) \rightarrow x \]
\[ mal(enc(x, y), z) \rightarrow enc(z, y) \]

Simple toy example used as a test case for several procedures. For the final rule:

\[ mal(enc(x, y), z) \rightarrow_{R_{gemb}} enc(x, y, z) \]
\[ \rightarrow_{R_{gemb}} enc(y, z) \approx enc(z, y) \]

Notice that $enc(x, y, z)$ is not well formed since it violates the arity of $enc()$. However, the final term is well formed, as required.
Example: Trap-Door Commitment

Theory of trap-door commitment, $R_{tdc}$, from [Ștefan Ciobâcă et al., 2012], is also graph-embedded:

\[
\begin{align*}
\text{open}(td(x, y, z), y) & \rightarrow x \\
\text{open}(td(x, y, z), f(x_1, y, z, x_2)) & \rightarrow x_2 \\
td(x_2, f(x_1, y, z, x_2), z) & \rightarrow td(x_1, y, z) \\
f(x_2, f(x_1, y, z, x_2), z, x_3) & \rightarrow f(x_1, y, z, x_3)
\end{align*}
\]
Example: Blind Signatures

The theory of blind signatures is also a graph-embedded TRS. All but the final rule are subterm. For the final rule:

\[
\text{unblind}(\text{sign}(\text{blind}(x, y), z), y) \rightarrow_{R_{gemb}} \text{sign}(\text{blind}(x, y), z) \text{ via rule (1)}
\]
\[
\text{sign}(\text{blind}(x, y), z) \rightarrow_{R_{gemb}} \text{sign}(x, y, z) \text{ via rule (3)}
\]
\[
\text{sign}(x, y, z) \rightarrow_{R_{gemb}} \text{sign}(x, z) \approx \text{sign}(x, z) \text{ via rule (2)}
\]
Local Stability

[Abadi and Cortier, 2006]:

• A convergent TRS, $R$

• For every frame $\phi = \nu \bar{n}.\{M_1/x_1, \ldots, M_k/x_k\}$, there exists a finite set $\text{sat}(\phi)$ such that:
  • each $M_i$ is in $\text{sat}(\phi)$,
  • any subterm of $\phi$ that can be formed from elements of $\text{sat}(\phi)$ by application of function symbols is also in $\text{sat}(\phi)$,
  • and it is closed under the application of small context.

Basically, it represents the intruder’s knowledge based on what they can see as the protocol runs.
Local Stability: Examples

Subterm Convergent Theories are locally stable [Abadi and Cortier, 2006].

The procedure of [Abadi and Cortier, 2006] also works for many other examples but local stability must be proven individually:

- blind signatures
- theory of addition
- theory of prefix with pairing
- and more
Contracting Convergent Systems

Possibility to identify a “large” subclass of graph-embedded convergent systems, called **contracting** convergent systems, for which any system in that subclass is locally stable.

A (tentative) definition:

- Rule (3) is forbidden.
- When rule (1) $f(\bar{x}) \rightarrow x_i$ is applied below the root position, only a variable instance applies, and there exists a rule $l'[f(\bar{x})] \rightarrow x_i$ if $x_i$ is not removed later.
- When rule (4) $f(\ldots, g(\bar{z}), \ldots) \rightarrow f(\ldots, \bar{z}, \ldots)$ is applied, only a variable instance applies, and there exists a rule $l'_i[g(\bar{z})] \rightarrow z_i$ for each $z_i$ not removed later.
- $\approx$ corresponds to the permutation of the direct subterms of the root term plus the permutation of leaves. If this is a way to get a rule $l[C[x]] \rightarrow r$ where $x$ occurs in $r$ without its cap $C$, then there exists a rule $l'[C[x]] \rightarrow x$. 
Main Results (Work in Progress)

**Theorem (decidability result):** Any contracting convergent TRS $R$ is locally stable. Consequently, both deduction and static equivalence are decidable for $R$. 

Proof: an encoding of the (modified) PCP `a la [Anantharaman et al., 2012] used initially to get undecidability of unification. 

[New] The same TRS as in [Anantharaman et al., 2012] can be applied to deduction as well, considering PCP.
Main Results (Work in Progress)

**Theorem (decidability result):** Any contracting convergent TRS $R$ is locally stable. Consequently, both deduction and static equivalence are decidable for $R$.

**Theorem (undecidability result):** There exists a graph-embedded convergent TRS, say $PE$, for which deduction modulo $PE$ is undecidable.

Proof: an encoding of the (modified) PCP (Post Correspondence Problem) à la [Anantharaman et al., 2012] used initially to get undecidability of unification.

[New] The same TRS as in [Anantharaman et al., 2012] can be applied to deduction as well, considering PCP.
Future Work

- A conference submission on *beyond subterm*
- Constructors defined via normalizing mappings vs. constructors defined via reduction orderings
- A journal submission on hierarchical unification
- Knowledge problems in unions of theories sharing only constructors modulo \( E \)
- Hierarchical approach applied to disunification? And to the knowledge problems?
- Congruence closure methods and syntactic theories
References I

Deciding knowledge in security protocols under equational theories.

Unification modulo homomorphic encryption.

Politeness and combination methods for theories with bridging functions.

Easy intruder deductions.

Decidability and combination results for two notions of knowledge in security protocols.

Computing knowledge in security protocols under convergent equational theories.
Notions of knowledge in combinations of theories sharing constructors.

Computing knowledge in equational extensions of subterm convergent theories.

Terminating non-disjoint combined unification.

Non-disjoint combined unification and closure by equational paramodulation.

Combined hierarchical matching: the regular case.


UNIF 2023 is the 37th event in a series of international meetings devoted to unification theory and its applications.

Submissions on applications of unification to security protocols are very welcome!

Submission deadline: April 21, 2023
Deduction Problem: Reduction Lemma

\[ \phi \vdash_{RE} t \text{ iff } \phi_* \vdash t \]

where \( \phi_* \) is a new frame defined as the \textit{completion} of \( \phi \)

Fortunately, \( \phi_* \) is \textit{computable} thanks to a fixpoint computation enumerating the finitely many subterms occurring in \( \phi \)
Static Equivalence: Reduction Lemma

\[ \phi \equiv_{RE} \psi \text{ iff } \psi \models Eq(\phi) \text{ and } \phi \models Eq(\psi) \]

where

- \( \psi \models Eq(\phi) \) denotes the fact that for any \( s = t \in Eq(\phi) \), \( (s =_{RE} t) \psi \)
- \( \zeta_\phi \) is the recipe substitution associated to \( \phi_* \)
- \( Eq(\phi) \) contains only finitely many equalities \( s\zeta_\phi = t\zeta_\phi \) such that \( (s\zeta_\phi =_{RE} t\zeta_\phi) \phi \) and \( s, t \) are bounded “public” terms
Soundness and completeness are proven using two technical lemmas:

**Equational step** Assume $\psi \models Eq(\phi)$. If $s\phi_* =_E t\phi_*$, then $(s\zeta_\phi)\psi =_{RE} (t\zeta_\phi)\psi$

**Rewrite step** Assume $\psi \models Eq(\phi)$. If $s\phi_* \rightarrow_R t$, then there exists a term $u$ satisfying the name restriction such that $t = u\phi_*$ and $(s\zeta_\phi)\psi =_{RE} (u\zeta_\phi)\psi$