Rewrite-based decision procedures

Maria Paola Bonacina

Dipartimento di Informatica
Università degli Studi di Verona
Verona, Italy, EU

Talk given at Microsoft Research, Redmond, Washington, USA

27 May 2008

Modularity of termination for combination of theories

Modularity of termination:
if $\mathcal{SP}_{\succ}$-strategy terminates on $\mathcal{T}_i$-sat problems then it terminates on $\mathcal{T}$-sat problems for $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$.

Hypotheses:

- No shared function symbols (shared constants allowed)
- *Variable-inactive* theories

Variable-inactivity

Clause C *variable-inactive*: no maximal literal in C is equation $t \simeq x$ where $x \notin Var(t)$

Set of clauses *variable-inactive*: all its clauses are

$\mathcal{T}$ *variable-inactive*: the limit $S_\infty = \bigcup_{j\geq 0} \bigcap_{i\geq j} S_i$ of a fair derivation from $\mathcal{T} \cup S$ is variable-inactive

Examples

$$
\begin{aligned}
C_1 &= \mathrm{car}(\mathrm{cons}(x, y)) \simeq x \\
C_2 &= z \simeq w \vee \mathrm{select}(\mathrm{store}(x, z, v), w) \simeq \mathrm{select}(x, w) \\
C_3 &= \bigvee_{1 \leq j < k \leq n} (x_j \simeq x_k)
\end{aligned}
$$

C_1 variable-inactive
C_2 variable-inactive
C_3 not variable-inactive (*cardinality constraint clause*)

The modularity theorem

Theorem: if

- $\mathcal{T}_i$, $1 \leq i \leq n$, do not share function symbols
- $\mathcal{T}_i$, $1 \leq i \leq n$, variable-inactive
- $\mathcal{SP}_{\succ}$-strategy is a $\mathcal{T}_i$-satisfiability procedure, $1 \leq i \leq n$,

then it is a $\mathcal{T}$-satisfiability procedure for $\mathcal{T} = \bigcup_{i=1}^{n} \mathcal{T}_i$.

All theories considered so far are variable-inactive.

Explanation of the proof of the theorem

- No shared function symbol: no paramodulation from compound terms across theories
- Variable-inactivity: no paramodulation from variables across theories, since for $t \simeq x$ where $x \in Var(t)$ it is $t \succ x$

Only paramodulations from constants into constants: finitely many.

Comment on shared function symbols

- If $\mathcal{T}_1$ contains an axiom where f occurs and $\mathcal{T}_2$ contains another axiom where f occurs, we may have all possible inferences between two general clauses, of whom we know no special properties or restrictions.
- The symbols from the theories appear freely mixed in S, and are separated by flattening (does the job of "purification").

Variable-inactive theories

- *Purely equational theories*:
  no trivial models $\Rightarrow$ variable-inactive
- *Horn theories*:
  no trivial models + maximal unit strategy $\Rightarrow$ variable-inactive
- *Maximal unit strategy*:
  restricts superposition to unit clauses and paramodulates unit clauses into maximal negative literals

Variable inactivity and stable-infiniteness

Lemma: If S_0 is a finite satisfiable set of clauses, then S_0 admits no infinite models if and only if the limit S_∞ of any fair $\mathcal{SP}_\succ$-derivation from S_0 contains a cardinality constraint clause.

Theorem: If $\mathcal{T}$ is variable-inactive, then it is stably-infinite.

Lemma from:
Maria Paola Bonacina, Silvio Ghilardi, Enrica Nicolini, Silvio Ranise and Daniele Zucchelli.
Decidability and undecidability results for Nelson-Oppen and rewrite-based decision procedures.

Proc. 3rd IJCAR, LNAI 4130:513-527, Springer 2006.

$\mathcal{T}$-decision procedure

$\mathcal{T}$-**decision procedure**: decide satisfiability of a *conjunction of ground clauses* in theory $\mathcal{T}$

S: *set of ground clauses* in the signature of $\mathcal{T}$

$\mathcal{T}$: *presentation* of a theory

$\bowtie$ is either $\simeq$ or $\not\simeq$

Flat and strictly flat

Depth:
$depth(t) = 0$, if t is constant or variable
$depth(t) = 1 + max\{depth(t_i) : 1 \leq i \leq n\}$, if t is $f(t_1, \ldots, t_n)$
$depth(l \bowtie r) = depth(l) + depth(r)$

Term: t is *flat* if $depth(t) \leq 1$, *strictly flat* if $depth(t) = 0$

Literal: $l \simeq r$ is *flat* if $depth(l \simeq r) \leq 1$
$l \not\simeq r$ is *flat* if $depth(l \not\simeq r) = 0$
$l \bowtie r$ is *strictly flat* if $depth(l \bowtie r) = 0$

Clause: C is *(strictly) flat* if all its literals are
$Maxd(C) = max\{depth(t) : t\ occurs\ in\ C\}$

Flattening

S: given set of ground clauses

S': flattened version of S such that

- all unit clauses in S' are flat
- all non-unit clauses in S' are strictly flat
- $\mathcal{T} \cup S \equiv_{\mathrm{s}} \mathcal{T} \cup S'$, where $\equiv_{\mathrm{s}}$ means equisatisfiable

Example

$$S = \{f(f(a)) \simeq b \vee f(c) \not\simeq d\}$$

$$S' = \{f(a) \simeq c_1,\ f(c_1) \simeq c_2,\ f(c) \simeq c_3,\ c_2 \simeq b \vee c_3 \not\simeq d\}$$

"Good" CSO

- Simplification ordering
- *Complete*: *total* on ground terms
- *"Good"*: $t \succ c$ for all ground compound terms t and constants c

Thus, we drop requirements such as $a \succ e \succ i$ for arrays.

Intuition

In a $\mathcal{T}$-decision problem we distinguish:

- $\mathcal{T}_g$: ground clauses
- $\mathcal{T}_1$: non-ground clauses about properties that can be deduced using one interpreted function
- $\mathcal{T}_2$: non-ground clauses about the interaction of two interpreted functions

Example: Arrays

$$\begin{array}{rc} \forall x, z, v. & \mathrm{select}(\mathrm{store}(x, z, v), z) \simeq v \\ \forall x, z, w, v. & z \not\simeq w \supset \mathrm{select}(\mathrm{store}(x, z, v), w) \simeq \mathrm{select}(x, w) \\ \forall x, y. & \forall z.\, \mathrm{select}(x, z) \simeq \mathrm{select}(y, z) \supset x \simeq y \end{array}$$

First two axioms: in $\mathcal{T}_2$
Third axiom (*extensionality*): in $\mathcal{T}_1$

Subterm-inactivity of a 3-tuple

$\mathcal{T}_g, \mathcal{T}_1, \mathcal{T}_2$: disjoint sets of clauses

$\langle \mathcal{T}_g, \mathcal{T}_1, \mathcal{T}_2 \rangle$ is *subterm-inactive* if

- $\mathcal{T}_g$ is ground and flattened
- $\mathcal{T}_1$ is interaction-free from $\mathcal{T}_2$ and satisfies certain closure properties
- $\mathcal{T}_2$ is saturated and satisfies certain closure properties

Subterm-inactivity of a set

Set P is *subterm-inactive* if $P = \mathcal{T}_g \uplus \mathcal{T}_1 \uplus \mathcal{T}_2$ such that $\langle \mathcal{T}_g, \mathcal{T}_1, \mathcal{T}_2 \rangle$ is subterm-inactive.

Say P is presentation $\mathcal{T}$: typically $\mathcal{T}_g = \emptyset$.

If presentation $\mathcal{T}$ is subterm-inactive, then $\mathcal{T} \cup S$, where S is ground and flattened, is also: $\mathcal{T} \cup S = S \uplus \mathcal{T}_1 \uplus \mathcal{T}_2$

The subterm-inactivity theorem

If $\mathcal{T}_g \uplus \mathcal{T}_1 \uplus \mathcal{T}_2$ is *subterm-inactive*, then:

- For all persistent clauses D generated by $\mathcal{SP}$:
  - $Maxd(D) \leq max\{Maxd(C) \colon C \; premise\}$ (depth-preserving)
  - If D is ground, $\langle \mathcal{T}_g \cup \{D\}, \mathcal{T}_1, \mathcal{T}_2 \rangle$ is *subterm-inactive*
  - If D is not ground, $\langle \mathcal{T}_g, \mathcal{T}_1 \cup \{D\}, \mathcal{T}_2 \rangle$ is *subterm-inactive*
- $\mathcal{SP}_{\succ}$-strategy is $\mathcal{T}$-decision procedure
- $\mathcal{T}$ is *variable-inactive* (hence easy to combine)

Impact of subterm-inactivity on binary inferences

- An $\mathcal{SP}$-inference between two clauses in $\mathcal{T}_1$ generates a clause in $\mathcal{T}_1$ or $\mathcal{T}_g$ (Closure properties)
- An $\mathcal{SP}$-inference between two clauses in $\mathcal{T}_2$ generates a clause that is deleted eventually (Saturation)
- No $\mathcal{SP}$-inference applies to a clause in $\mathcal{T}_1$ and a clause in $\mathcal{T}_2$ (Interaction-freeness)
- An $\mathcal{SP}$-inference between a clause in $\mathcal{T}_1 \cup \mathcal{T}_2$ and a clause in $\mathcal{T}_g$ generates a clause in $\mathcal{T}_1$ or $\mathcal{T}_g$ (Closure properties + flatness)

Impact of subterm-inactivity on unary inferences

- An $\mathcal{SP}$-inference from a clause in $\mathcal{T}_1$ generates a clause that is in $\mathcal{T}_1$ or in $\mathcal{T}_g$ or is deleted eventually (Closure properties)
- An $\mathcal{SP}$-inference from a clause in $\mathcal{T}_2$ generates a clause that is deleted eventually (Saturation)

Subterm-inactive theories

- Equality
- Arrays with or without extensionality + two variations
- Recursive data structures (including integer offsets and acyclic lists)
- Finite sets with or without extensionality

Not included: Records, integer offsets modulo, possibly empty possibly cyclic lists

Variations on the theory of arrays

- Add to $\mathcal{A}$ an *injectivity predicate* to state than an array is injective:
  $Inj(x) \Leftrightarrow \forall z, w.\ (z \not\simeq w \supset \mathrm{select}(x, z) \not\simeq \mathrm{select}(x, w))$
- Add to $\mathcal{A}$ a *swap predicate*:
  $Swap(x, y, z_1, z_2) \Leftrightarrow$
  $\mathrm{select}(x, z_1) \simeq \mathrm{select}(y, z_2) \wedge$
  $\mathrm{select}(x, z_2) \simeq \mathrm{select}(y, z_1) \wedge$
  $\forall w.\ (w \not\simeq z_1 \wedge w \not\simeq z_2 \supset \mathrm{select}(x, w) \simeq \mathrm{select}(y, w))$

Finite sets

$$
\begin{array}{rc}
\forall x, v. & \mathrm{member}(v, \mathrm{insert}(v, x)) \simeq \mathit{true} \\
\forall x, v, w. & v \not\simeq w \supset \mathrm{member}(v, \mathrm{insert}(w, x)) \simeq \mathrm{member}(v, x) \\
\forall x, y. & \forall v.\, \mathrm{member}(v, x) \simeq \mathrm{member}(v, y) \supset x \simeq y
\end{array}
$$

First two axioms: $\mathcal{FS}$
With third axiom (*extensionality*): $\mathcal{FS}^e$

A non-obvious example

If we add to $\mathcal{A}$ the axiom:
$Const_y(x) \Leftrightarrow \forall z.\ \mathrm{select}(x, z) \simeq y$
the resulting theory is not subterm-inactive.

It is not variable-inactive either:
$S = \{store(a, i, e_1) \simeq a',\ Const_e(a),\ Const_{e'}(a')\} \Leftrightarrow$
$\{store(a, i, e_1) \simeq a',\ \mathrm{select}(a, z) \simeq e,\ \mathrm{select}(a', z) \simeq e'\}$

Superposition of $store(a, i, e_1) \simeq a'$ into axiom
$z \simeq w \vee \mathrm{select}(\mathrm{store}(x, z, v), w) \simeq \mathrm{select}(x, w)$
generates $w \simeq i \vee \mathrm{select}(a, w) \simeq \mathrm{select}(a', w)$
which is simplified to
$w \simeq i \vee e \simeq e'$
which is not variable-inactive because $w \simeq i$ is maximal.

Discussion

- Termination results for T-satisfiability procedures were obtained by analyzing all possible $\mathcal{SP}$-inferences
- Subterm-inactivity is obtained by generalizing those analyses
- Its conditions are syntactic and most of them could be tested automatically
- However, they are very complicated, inter-twined and not intuitive
- Simpler, hence better, approach: T-decision procedures assuming only *variable-inactivity*

Flattening again

S: given set of ground clauses

$S_1 \uplus S_2$: flattened version of S such that

- S_1: unit flat clauses
- S_2: strictly flat non-unit clauses
- $\mathcal{T} \cup S \equiv_s \mathcal{T} \cup S_1 \cup S_2$, where $\equiv_s$ means equisatisfiable

Example

$$S = \{f(a) \not\simeq f(b) \vee f(a) \not\simeq f(c)\}$$

$$S_1 = \{f(a) \simeq a',\ f(b) \simeq b',\ f(c) \simeq c'\}$$

$$S_2 = \{a' \not\simeq b' \vee a' \not\simeq c'\}$$

where a', b', c' are new constants

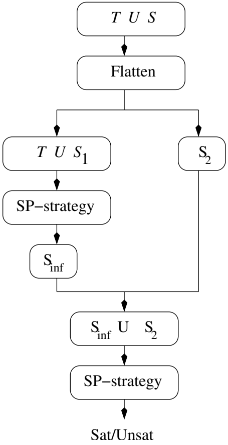
$\mathcal{T}$-decision procedures based on variable-inactivity

Theorem: if

- $\mathcal{T}$ is variable inactive
- $\mathcal{SP}_{\succ}$-strategy is $\mathcal{T}$-satisfiability procedure

then it is also $\mathcal{T}$-decision procedure.

$\mathcal{T}$-decision scheme



Explanation: analysis of inferences

Lemma:
C: variable-inactive clause
C': strictly flat ground clause

1. C' paramodulates into C:
   $C = l[a] \bowtie r \vee D$
   $C' = a \simeq a'' \vee D'$
   Generated clause: $l[a''] \bowtie r \vee D \vee D'$
2. C paramodulates into C':
   $C = a \simeq a' \vee D$ (also strictly flat)
   $C' = a \bowtie a'' \vee D'$
   Generated clause: $a' \bowtie a'' \vee D \vee D'$ (also strictly flat)

In both cases mgu is empty.

Proof of the $\mathcal{T}$-decision theorem

- S_∞ limit of derivation from $\mathcal{T} \cup S_1$ is
  - finite
  - variable-inactive
- S_2 is strictly flat
- All inferences between S_∞ and S_2 are paramodulations from constants into constants: finitely many in a finite signature

Putting it all together

- Variable-inactivity is the fundamental requirement for both
  - Combination of theories
  - Generalization to $\mathcal{T}$-decision problems
- The $\mathcal{T}$-decision scheme applies also when $\mathcal{T}$ is a union of variable-inactive theories
- The two applications of $\mathcal{SP}$ are only for clarity: if S_∞ limit of derivation from $\mathcal{T} \cup S_1$ is finite and variable-inactive, it will be such also in a single run from $\mathcal{T} \cup S_1 \cup S_2$, and it will have only finitely many inferences with strictly flat S_2.

Decomposition: unite FOL+= and SMT strengths

- *Decomposition*: definitional and operational part
- *Theory compilation*: apply FOL+= prover to "compile" the definitional part: theory reasoning, non-ground equational reasoning
- *Decision*: apply SMT-solver to subset of saturated set (without $\mathcal{T}$-axioms) + operational part
- Sufficient conditions to preserve satisfiability

Decomposition

Decomposition: generalization of flattening, where S is decomposed into S_1 and S_2; it suffices that S_1 be made of *flat unit clauses*.

- **Records**: S_1 contains the clauses $\mathrm{rstore}_i(a, e) \simeq b$ and S_2 contains everything else
- **Integer offsets**: same as flattening
- **Arrays**: S_1 contains the clauses $\mathrm{store}(a, i, e) \simeq a'$ and S_2 contains everything else

Framework of sufficient conditions

$\mathcal{T}$-compatibility: A is $\mathcal{T}$-compatible with S if A entails every clause generated from premise in S and premise in $\mathcal{T}$

Theorem: $\bar{S}$ is $\mathcal{T}$-compatible with S where $S_\infty = \mathcal{T} \cup \bar{S}$ is the limit generated by $\mathcal{SP}$ from $\mathcal{T} \cup S$

$\mathcal{T}$-stability: ensures that $\mathcal{T}$-compatibility is preserved by all inferences:
if A is $\mathcal{T}$-compatible with S and $\mathcal{T} \cup S \vdash \mathcal{T} \cup S'$ then A is $\mathcal{T}$-compatible with S'.

$\mathcal{T}$-decision by stages: the main theorem

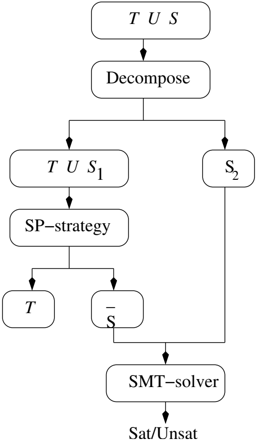
Theorem: under $\mathcal{T}$-stability, if A and A' are sets of clauses such that

- $\mathcal{T} \cup S_1 \models A$
- $\mathcal{T} \cup S_2 \models A'$
- A is $\mathcal{T}$-compatible with S_1
- A' is $\mathcal{T}$-compatible with S_2

then $\mathcal{T} \cup S_1 \cup S_2 \equiv_{\mathrm{s}} A \cup A'$.

Instance of the theorem:A' is S_2 itself; A is $\bar{S}$ where $S_\infty = \mathcal{T} \cup \bar{S}$ is the limit generated by $\mathcal{SP}$ from $\mathcal{T} \cup S_1$

$\mathcal{T}$-decision by stages



Application to the theories

- **Records**: $\bar{S} \cup S_2$ is ground: its satisfiability can be decided by decision procedure for equality (reduction to EUF)
- **Integer offsets**: same as for records
- **Arrays**: $\bar{S} \cup S_2$ is not ground: $\bar{S}$ contains $\mathrm{select}(a, x) \simeq \mathrm{select}(a', x) \vee x \simeq i_1 \vee \ldots \vee x \simeq i_n \vee B$
  where B is possibly empty, ground, strictly flat
  It falls in the *array property fragment*.

Postponing theories

How about theories such as arithmetic or bitvectors that do not lend themselves to general first-order deduction?

Those parts of the problem can be left into S_2 and passed on directly to the SMT-solver.

References

- Alessandro Armando, Maria Paola Bonacina, Silvio Ranise and Stephan Schulz. *New results on rewrite-based satisfiability procedures*. *ACM Trans. on Computational Logic*, To appear. (Presented in part at *FroCoS 2005* and *PDPAR 2005*)
- Maria Paola Bonacina and Mnacho Echenim. *Rewrite-based decision procedures*. *Proc. 6th STRATEGIES Workshop, FLoC 2006*, ENTCS 174(11):27-45, Elsevier 2007.
- Maria Paola Bonacina and Mnacho Echenim. *On variable-inactivity and polynomial T-satisfiability procedures*. *Journal of Logic and Computation*, 18(1): 77-96, Feb. 2008. (Presented in part at *PDPAR 2006*)
- Maria Paola Bonacina and Mnacho Echenim. *Theory decision by decomposition*. Submitted to journal, April 2008. (Presented in part at *CADE 2007*)