Data Structures with Arithmetic Constraints: a Non-Disjoint Combination

E. Nicolini, C. Ringeissen, and M. Rusinowitch

LORIA & INRIA Nancy Grand Est

FroCoS'09

Outline

Outline

Building and Combining Decision Procedures

- Use Rewriting techniques
  - use a *superposition calculus* for FOL with Equality and prove its termination for useful cases in verification

➻ Application to data structures [ARR03, ABRS09, BE07, dMB08]

- Use Combination techniques
  - use procedures available for individual theories and try to build a procedure for the union of theories

➻ Application to disjoint unions of data structures and fragments of arithmetic [KRRT05]

Our approach

Use both Rewriting an Combination techniques to consider non-disjoint unions of data structures and fragments of arithmetic
➻ Application of the combination method proposed by Ghilardi-Nicolini-Zucchelli [GNZ08]: a combination method à la Nelson-Oppen [NO79] for non-disjoint unions of theories

Outline

Data structures using arithmetic operators

Lists :nil : LISTS, cons : ELEM $\times$ LISTS $\rightarrow$ LISTS, ℓ : LISTS $\rightarrow$ NUM

$$\ell(\mathrm{nil}) = 0$$
$$\ell(\mathrm{cons}(x, y)) = \mathrm{s}(\ell(y))$$

Trees :bin : ELEM $\times$ TREES $\times$ TREES $\rightarrow$ TREES, null : TREES, size_L : TREES $\rightarrow$ NUM, size_R : TREES $\rightarrow$ NUM

$$\mathrm{size}_L(\mathrm{null}) = 0 \qquad \mathrm{size}_R(\mathrm{null}) = 0$$
$$\mathrm{size}_L(\mathrm{bin}(e, t_1, t_2)) = \mathrm{s}(\mathrm{size}_L(t_1)) \qquad \mathrm{size}_R(\mathrm{bin}(e, t_1, t_2)) = \mathrm{s}(\mathrm{size}_R(t_2))$$

Records : sel_i : RECS $\rightarrow$ NUM, inc : RECS $\rightarrow$ RECS

$$sel_i(inc(r)) = \mathrm{s}(sel_i(r))$$

for any index i of sort NUM.

The Shared Theory of Increment

$$
\begin{array}{ll}
\text{(Inj)} & \forall x, y \;\; s(x) = s(y) \rightarrow x = y \\
\text{(Acy)} & \forall x \;\; x \neq s^n(x) \text{ for all } n \in \mathbb{N}^+ \\
\text{(S0)} & \forall x \; s(x) \neq 0
\end{array}
$$

1. Theory of Integer Offsets [NRR09b]: $T_I = \{Inj, Acy, S0\}$
2. Theory of Increment (this paper): $T_S = \{Inj, Acy\}$

Superposition Calculus

Superposition	$\dfrac{l[u'] = r \quad u = t}{(l[t] = r)\sigma}$	$(i), (ii), (iii), (iv)$
Paramodulation	$\dfrac{l[u'] \neq r \quad u = t}{(l[t] \neq r)\sigma}$	$(i), (ii), (iii), (iv)$
Reflection	$\dfrac{u' \neq u}{\bot}$	(i)

where *(i)* σ is the most general unifier of u and u', *(ii)* u' is not a variable , *(iii)* $u\sigma \not\preceq t\sigma$, *(iv)* $l[u']\sigma \not\preceq r\sigma$.

Figure: Expansion Inference Rules.

Superposition Calculus (for a successor function)

Ad hoc rules to be applied to ground terms:

R1 (for Inj)	$\dfrac{S \cup \{s(u) = s(v)\}}{S \cup \{u = v\}}$	
R2 (for Inj)	$\dfrac{S \cup \{s(u) = t, s(v) = t\}}{S \cup \{s(v) = t, u = v\}}$	if $s(u) \succ t$, $s(v) \succ t$ and $u \succ v$
C1 (for Acy)	$\dfrac{S \cup \{s^n(t) = t\}}{S \cup \{s^n(t) = t\} \cup \bot}$	if $n \in \mathbb{N}$

where S is a set of literals and $\bot$ is the symbol for the inconsistency.

Figure: Ground reduction Inference Rules.

Superposition Calculus as Decision Procedure

Result

An appropriate Superposition Calculus leads to a decision procedure for a class of theories **DST** modelling data-structures with the unary successor function.

DST includes: Lists with length, Trees with size, Records with increment.

Proof: For any theory $T \in$ **DST** and any set of ground flat literals G, any saturation of $Ax(T) \cup G$ is as follows:

- It must be finite.
- Some forms of non-ground equalities must be excluded.

Outline

Linear Arithmetic

$\Sigma_{\mathbb{Q}} := \{0, 1, +, -, \{q\_\}_{q \in \mathbb{Q}}, \mathsf{s}, <\}$, where $0, 1$ are constants, and $-$, $q\_$, s are unary function symbols.

Let $T_{\mathbb{Q}}$ be the set of all the $\Sigma_{\mathbb{Q}}$-sentences that are true in $\mathbb{Q}$.

Fact

A $T_{\mathbb{Q}}$-satisfiability procedure can be obtained by using

1. Fourier-Motzkin Elimination (for inequalities)
   ➻ to detect unsatisfiability or to compute implicit equalities
2. Gauss Elimination (for equalities)
   ➻ a function **solve** to compute the solved form of a set of equalities
3. Disequality Handler
   ➻ a function **canon** over arithmetic expressions to check whether an disequality can be canonized into an unsatisfiable disequality $u \neq u$.

Non-Linear Arithmetic: The Theory of $\mathbb{Q}$-Algebras

$T_{\mathbb{Q}\text{-}alg}$ is $AC(+) \cup AC(\times) \cup U(+, 0) \cup U(\times, 1)$ plus

$$\forall x\ x + (-x) = 0 \tag{1}$$

$$0 \neq 1 \tag{2}$$

$$\forall x\ \mathsf{s}(x) = x + 1 \tag{3}$$

$$\forall x, y, z\ (x + y)z = xz + yz \tag{4}$$

$$\forall x, y\ q(x + y) = qx + qy \tag{5}$$

$$\forall x\ (q_1 \oplus q_2)x = q_1 x + q_2 x \tag{6}$$

$$\forall x\ (q_1 \cdot q_2)x = q_1(q_2 x) \tag{7}$$

$$\forall x\ 1_{\mathbb{Q}} x = x \tag{8}$$

$$\forall x, y\ q(xy) = x(qy) \tag{9}$$

Fact

A $T_{\mathbb{Q}\text{-}alg}$-satisfiability procedure can be obtained by using the Buchberger algorithm for the computation of Groebner bases.

Outline

A combination problem

$$\Gamma_1 = \left\{ \begin{array}{l} y = \ell(a) \\ b = cons(e, a) \\ x = \ell(b) \end{array} \right\}$$

$$\Gamma_2 = \left\{ \begin{array}{l} u \geq 0 \\ x + u = y \end{array} \right\}$$

Satisfiability of $\Gamma_1 \cup \Gamma_2$?

$\Gamma_1 \cup \Gamma_2$ is unsatisfiable since

- $\Gamma_1 \to x = \mathrm{s}(y)$
- $\Gamma_2 \cup \{x = \mathrm{s}(y)\}$ is T_2-unsatisfiable:

$$\Gamma_2 \cup \{x = \mathrm{s}(y)\} \leftrightarrow \{u \geq 0, u = -1\}$$

A combination problem

$$\Gamma_1 = \left\{ \begin{array}{l} y = \ell(a) \\ b = cons(e, a) \\ x = \ell(b) \end{array} \right\}$$

$$\Gamma_2 = \left\{ \begin{array}{l} u \geq 0 \\ x + u = y \end{array} \right\}$$

Satisfiability of $\Gamma_1 \cup \Gamma_2$?

$\Gamma_1 \cup \Gamma_2$ is unsatisfiable since

- $\Gamma_1 \rightarrow x = \mathsf{s}(y)$
- $\Gamma_2 \cup \{x = \mathsf{s}(y)\}$ is T_2-unsatisfiable:

$$\Gamma_2 \cup \{x = \mathsf{s}(y)\} \leftrightarrow \{u \geq 0, u = -1\}$$

Non-disjoint combination method (à la Nelson-Oppen)

Combination method developed by Ghilardi-Nicolini-Zucchelli [GNZ08]:

Let $T_0 = T_1 \cap T_2$ and $\Sigma_0 = \Sigma_1 \cap \Sigma_2$

Purification Given a set of $T_1 \cup T_2$-constraints Γ, produce an equisatisfiable set of pure constraints $\Gamma_1 \cup \Gamma_2$;

Propagation the T_1-constraint solving procedure and the T_2-constraint solving procedure fairly exchange shared positive Σ_0-clauses that are entailed by $T_1 \cup \Gamma_1$ and by $T_2 \cup \Gamma_2$

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

Pseudo-code

1. If T_0-basis$_{T_i}(\Gamma_i) = \Delta_i$ and $\bot \notin \Delta_i$ for each $i \in \{1, 2\}$, then
   - 1.1. For each $D \in \Delta_i$ such that $T_j \cup \Gamma_j \not\models D$, $(i \neq j)$, add D to Γ_j
   - 1.2. If Γ_1 or Γ_2 has been changed in **1.1**, then rerun **1.**

   Else **return** *Unsatisfiable*
2. **Return** *Satisfiable*.

Non-disjoint combination method (à la Nelson-Oppen)

Combination method developed by Ghilardi-Nicolini-Zucchelli [GNZ08]:

Let $T_0 = T_1 \cap T_2$ and $\Sigma_0 = \Sigma_1 \cap \Sigma_2$

Purification Given a set of $T_1 \cup T_2$-constraints Γ, produce an equisatisfiable set of pure constraints $\Gamma_1 \cup \Gamma_2$;

Propagation the T_1-constraint solving procedure and the T_2-constraint solving procedure fairly exchange shared positive Σ_0-clauses that are entailed by $T_1 \cup \Gamma_1$ and by $T_2 \cup \Gamma_2$

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

Pseudo-code

1. If $T_0\text{-basis}_{T_i}(\Gamma_i) = \Delta_i$ and $\bot \notin \Delta_i$ for each $i \in \{1, 2\}$, then

   1.1. For each $D \in \Delta_i$ such that $T_j \cup \Gamma_j \not\models D$, $(i \neq j)$, add D to Γ_j

   1.2. If Γ_1 or Γ_2 has been changed in **1.1**, then rerun **1.**

  Else **return** *Unsatisfiable*

2. Return *Satisfiable*.

Combination method: critical points

1. How to obtain the T_0-bases, which are logical consequences of a constraint Γ w.r.t. a theory T_0 over a given sub-signature
   ➼ Computability of T_0-bases
2. How to guarantee the termination of the exchange loop
   ➼ *Noetherianity* of T_0
3. How to ensure its completeness
   ➼ T_0-*compatibility* (extends the assumption on *stably infinite theories* used in the disjoint case)

Our work

How to face these issues when dealing with a combination of

1. a data structure in **DST**
2. a theory of arithmetic in $\{T_{\mathbb{Q}}, T_{\mathbb{Q}\text{-}alg}\}$

where the shared theory T_0 is the theory of Increment T_S.

Computation of T_S-bases for data structures

Result

Our Superposition Calculus computes T_S-bases for any $T \in$ **DST**.

How to compute T_S-bases: collect all the shared equalities in a saturation of Γ not containing $\bot$.

Example (theory of Lists with length)

The saturation of

$$\Gamma_1 = \{y = \ell(a), b = cons(e, a), x = \ell(b)\}$$

contains

$$x = \mathsf{s}(y)$$

Remark

Similar result in [NRR09b] for the shared theory of Integer Offsets.

Computation of T_S-bases for fragments of arithmetic

Result

T_S-bases are computable for $T_{\mathbb{Q}}$ and $T_{\mathbb{Q}\text{-}alg}$.

Proof Idea:

1. (Linear case) Assume Γ is a set of linear equalities. We have

$$T \cup \Gamma \models a_1 = \mathrm{s}^n(a_2) \iff \textbf{canon}(a_1\gamma - a_2\gamma) = n$$

where $\gamma = \textbf{solve}(\Gamma)$.

2. (Non-linear case) It is possible to compute the set of all entailed linear equalities by using a slight adaptation of the Buchberger algorithm, as shown in Nicolini's thesis. Then proceed as in (1).

Computation of T_S-bases: example for the arithmetic

Example (theory of arithmetic $T_{\mathbb{Q}}$)

$$\Gamma_2 = \begin{cases} x = c \\ 1 + 2c + y = 2 + 3d \\ 2c = d + x \end{cases}$$

Γ_2 is equivalent to the solved form:

$$solve(\Gamma_2) = \begin{cases} x = c \\ y = c + 1 \\ d = c \end{cases}$$

Therefore:

$$\Gamma_2 \rightarrow y = s(x)$$

Computation of T_S-bases: example for the arithmetic

Example (theory of arithmetic $T_{\mathbb{Q}}$)

$$\Gamma_2 = \begin{cases} x = c \\ 1 + 2c + y = 2 + 3d \\ 2c = d + x \end{cases}$$

Γ_2 is equivalent to the solved form:

$$solve(\Gamma_2) = \begin{cases} x = c \\ y = c + 1 \\ d = c \end{cases}$$

Therefore:

$$\Gamma_2 \to y = s(x)$$

Computation of T_S-bases: example for the arithmetic

Example (theory of arithmetic $T_{\mathbb{Q}}$)

$$\Gamma_2 = \begin{cases} x = c \\ 1 + 2c + y = 2 + 3d \\ 2c = d + x \end{cases}$$

Γ_2 is equivalent to the solved form:

$$solve(\Gamma_2) = \begin{cases} x = c \\ y = c + 1 \\ d = c \end{cases}$$

Therefore:

$$\Gamma_2 \rightarrow y = \mathrm{s}(x)$$

Data structures with arithmetic constraints

Example (Previous Examples Continued)

- In the theory of Lists with length:
  given $\Gamma_1 = \{y = \ell(a), b = cons(e, a), x = \ell(b)\}$, we have:

$$\Gamma_1 \rightarrow x = \mathrm{s}(y)$$

- In the theory of arithmetic $T_{\mathbb{Q}}$:
  given $\Gamma_2 = \{x = c, 1 + 2c + y = 2 + 3d, 2c = d + x\}$, we have:

$$\Gamma_2 \rightarrow y = \mathrm{s}(x)$$

- In the union of theories:

$$\Gamma_1 \cup \Gamma_2 \text{ is unsatisfiable}$$

since $\{x = \mathrm{s}(y), y = \mathrm{s}(x)\}$ is T_S-unsatisfiable

Main result

We have identified a class of theories **DST** modelling data structures modulo T_S such that for any $T \in \mathbf{DST} \cup \{T_{\mathbb{Q}}, T_{\mathbb{Q}\text{-}alg}\}$:
the Ghilardi-Nicolini-Zucchelli combination method is

1. effective: T_S-basis$_T$ is computable
2. terminating: T_S is *Noetherian*
3. complete: T is T_S-*compatible*

Theorem

For any Σ_1-theory $T_1 \in$ **DST** *and any Σ_2-theory $T_2 \in \{T_{\mathbb{Q}}, T_{\mathbb{Q}\text{-}alg}, T_{\mathbb{Q}} \cup T_{\mathbb{Q}\text{-}alg}\} \cup$* **DST** *such that $\Sigma_1 \cap \Sigma_2 = \Sigma_S$, $T_1 \cup T_S \cup T_2$ has a decidable constraint satisfiability problem.*

Outline

Conclusion and future work

- sharing the theory of Increment (this paper): two possible theories of arithmetic over the the rationals, $T_{\mathbb{Q}}$ and $T_{\mathbb{Q}\text{-}alg}$
- sharing the theory of Integer Offsets [NRR09b]: which theory of arithmetic over the integers?
  ➻ Computation of bases seems more difficult for the integers!
- sharing the theory of Abelian Groups [NRR09a]: which theory of arithmetic sharing the $+$ operator?
  ➻ Computation of bases?
- How to deal with a *non-convex* data structure such as arrays?
  ➻ adaptation of the superposition calculus, to handle clauses instead of unit clauses

References

- Alessandro Armando, Maria P. Bonacina, Silvio Ranise, and Stephan Schulz.
  New results on rewrite-based satisfiability procedures.
  ACM Transactions on Computational Logic, 10(1), 2009.
- Alessandro Armando, Silvio Ranise, and Michaël Rusinowitch.
  A rewriting approach to satisfiability procedures.
  Information and Computation, 183(2):140–164, 2003.
- Maria Paola Bonacina and Mnacho Echenim.
  T-decision by decomposition.
  In *Proc. of CADE'07*, volume 4603 of *LNCS*, pages 199–214. Springer, July 2007.
- Leonardo Mendonça de Moura and Nikolaj Bjørner.
  Engineering DPLL(T) + Saturation.
  In *Proc. of IJCAR'08*, volume 5195 of *LNCS*, pages 475–490. Springer, 2008.
- Silvio Ghilardi, Enrica Nicolini, and Daniele Zucchelli.
  A comprehensive combination framework.
  ACM Transactions on Computational Logic, 9(2):1–54, 2008.
- Hélène Kirchner, Silvio Ranise, Christophe Ringeissen, and Duc-Khanh Tran.
  On superposition-based satisfiability procedures and their combination.
  In D. Van Hung and M. Wirsing, editors, *Proc. of ICTAC 2005*, volume 3722 of *LNCS*, pages 594–608, Hanoi (Vietnam), 2005. Springer-Verlag.
- Greg Nelson and Derek C. Oppen.

Simplification by cooperating decision procedures.
ACM Transaction on Programming Languages and Systems, 1(2):245–257, 1979.

Enrica Nicolini, Christophe Ringeissen, and Michaël Rusinowitch.
Combinable extensions of abelian groups.
In *Proc. of CADE'09*, volume 5663 of *LNAI*, pages 51–66. Springer, 2009.

Enrica Nicolini, Christophe Ringeissen, and Michaël Rusinowitch.
Satisfiability procedures for combination of theories sharing integer offsets.
In *Proc. of TACAS'09*, volume 5505 of *LNCS*, pages 428–442. Springer, 2009.