Dèmonstration automatique d'obligations de preuve

S. Ranise

LORIA & INRIA-Lorraine
Nancy (France)

GECCOO–Evaluation meeting
January 15, 2007

Outline

Outline

Outline

Outline

Outline

The Satisfiability Modulo Theories (*SMT*) problem

- Automatically discharging proof obligations amounts to checking the satisfiability of first-order formulae w.r.t. a background theory
  - first-order formulae encoding program + property
  - background theory encoding relevant aspects of the program (e.g., user-defined data types, memory model, type system)
- Since usual deduction techniques cannot solve interesting SMT problems, many SMT solvers are developed:
  Simplify, CVC, STeP, MathSAT, ICS, Barcelogic tool, **haRVey**, Yices, Ario, CVC-Lite, **Ergo**, ...

The Satisfiability Modulo Theories (*SMT*) problem

- Automatically discharging proof obligations amounts to checking the satisfiability of first-order formulae w.r.t. a background theory
  - first-order formulae encoding program + property
  - background theory encoding relevant aspects of the program (e.g., user-defined data types, memory model, type system)
- Since usual deduction techniques cannot solve interesting SMT problems, many SMT solvers are developed:
  Simplify, CVC, STeP, MathSAT, ICS, Barcelogic tool, **haRVey**, Yices, Ario, CVC-Lite, **Ergo**, ...

The Satisfiability Modulo Theories (*SMT*) problem

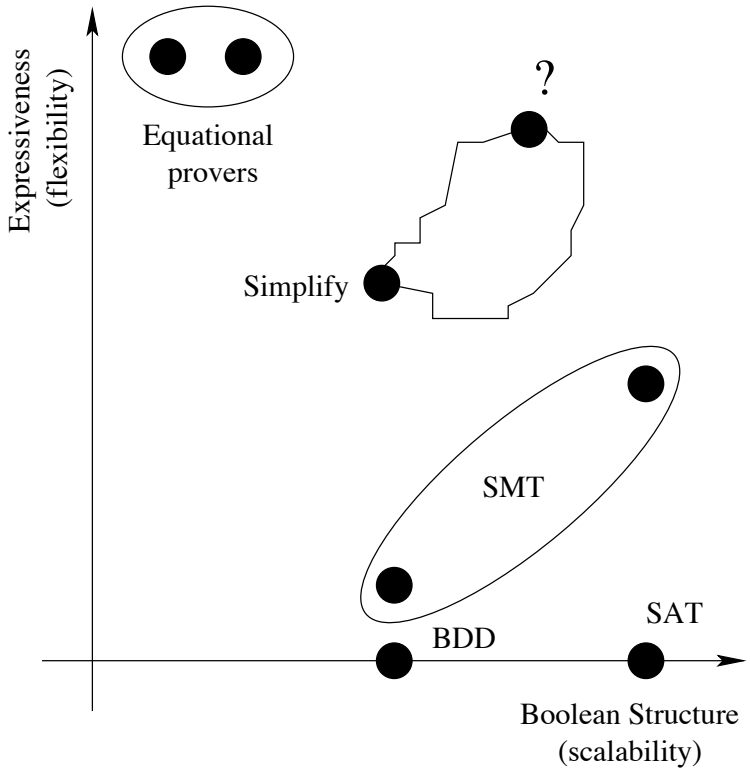
- Automatically discharging proof obligations amounts to checking the satisfiability of first-order formulae w.r.t. a background theory
  - first-order formulae encoding program + property
  - background theory encoding relevant aspects of the program (e.g., user-defined data types, memory model, type system)
- Since usual deduction techniques cannot solve interesting SMT problems, many SMT solvers are developed:
  Simplify, CVC, STeP, MathSAT, ICS, Barcelogic tool, **haRVey**, Yices, Ario, CVC-Lite, **Ergo**, ...

SMT solvers for Program Verification

Design SMT solvers with the following features:

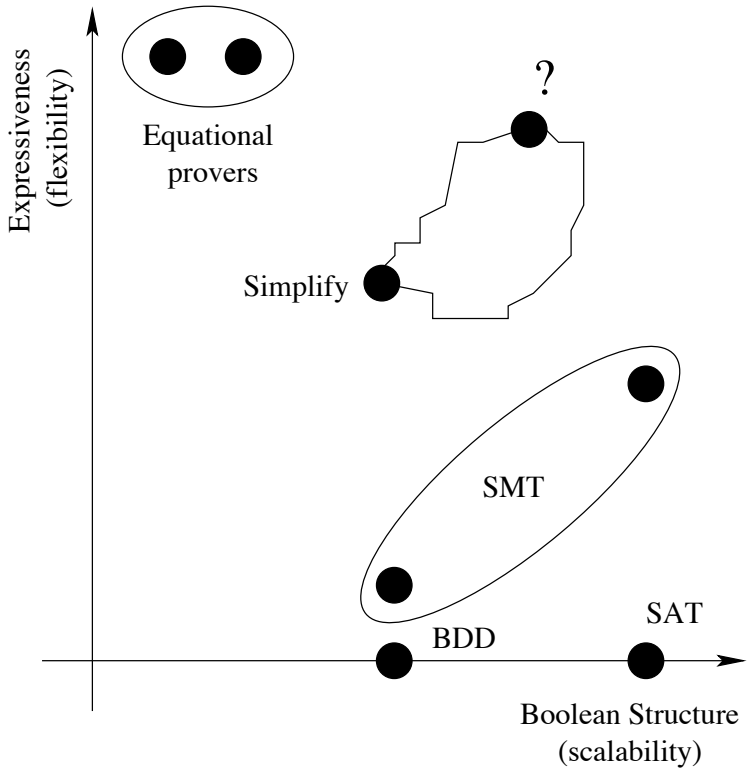
- predictable: by reasoning modulo a decidable theory
- scalable: by carefully integrating efficient tools for handling various aspects of SMT pbs (e.g., Boolean structure, reasoning in the background theory, ...)
- expressive: by developing automated deduction techniques combining richer and richer theories (equality, fragments of Arithmetics, lists, arrays, pointers ...)

The situation (state-of-the-art) and beyond...



- sharp trade-off between expressiveness and scalability
- *Simplify* stands in-between but biased to expressiveness
- Our goal: design and implement new techniques for re-using available tools and ideas (SMT) to build satisfiability solvers ('?') offering a better trade-off between expressiveness and scalability, and providing us the certification of results for integration in larger (skeptical) verification tools

The situation (state-of-the-art) and beyond...



- sharp trade-off between expressiveness and scalability
- *Simplify* stands in-between but biased to expressiveness
- Our goal: design and implement new techniques for re-using available tools and ideas (SMT) to build satisfiability solvers ('?') offering a better trade-off between expressiveness and scalability, and providing us the certification of results for integration in larger (skeptical) verification tools

Outline

haRVey architecture: basics (before GECCOO)

function Bool + *T* (ϕ: *ground formula*)
  $\phi^p \longleftarrow$ *nf*(*abs*(ϕ)) // *Abstract*
  while Bool-*satisfiable*(ϕ^p) **do begin**
    $\beta^p \longleftarrow$ *pick_prop_assignment*(ϕ^p)
    $(\rho, \pi) \longleftarrow$ *T-sat*(*prop2fol*(β^p)) // *Check*
    if $\rho \neq \bot$ **then return** *yes*
    $\phi^p \longleftarrow \phi^p \wedge \neg$*fol2prop*($\pi$) // *Refine*
  end
  return *no*
end

- π is the conflict set
- $\neg$*fol2prop*(π) is the conflict clause

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination *[ARR03]*
  - easy construction of conflict sets from proofs built by equational provers *[DR03]*
  - does not work for fragments of Arithmetics!

    ➠ Need of combining decision procedures!

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination *[ARR03]*
  - easy construction of conflict sets from proofs built by equational provers *[DR03]*
  - does not work for fragments of Arithmetics!
    - ➠ Need of combining decision procedures!

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination ***[ARR03]***
  - easy construction of conflict sets from proofs built by equational provers ***[DR03]***
  - does not work for fragments of Arithmetics*!*

➽ Need of combining decision procedures*!*

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination ***[ARR03]***
  - easy construction of conflict sets from proofs built by equational provers ***[DR03]***
  - does not work for fragments of Arithmetics*!*

➽ Need of combining decision procedures*!*

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination ***[ARR03]***
  - easy construction of conflict sets from proofs built by equational provers ***[DR03]***
  - does not work for fragments of Arithmetics*!*

➽ Need of combining decision procedures*!*

A uniform methodology to design *T-sat*

- Superposition = refutation complete deduction rules for FOL with equality
- Terminates on interesting classes of formulae
- Provides rewrite-based decision procedures for free
  - works for data structures axiomatizations, e.g., uninterpreted function symbols, lists, arrays, and their combination ***[ARR03]***
  - easy construction of conflict sets from proofs built by equational provers ***[DR03]***
  - does not work for fragments of Arithmetics*!*

➸ Need of combining decision procedures*!*

Starting point: Nelson-Oppen combination schema

Nelson and Oppen's key idea: use satisfiability procedures as black boxes and synchronize them by exchanging a certain class of entailed formulae.

Strong Limitations:

- component theories must be signature-disjoint
- component theories must be stably infinite (i.e. satisfiable formulae have infinite models)

  ➽ not always the case, e.g. enumerated data types*!*
- Component decision procedures are considered as black boxes but we combine rewriting-based procedures with black box decision procedures...

  ➽ does this change the problem*?*

Outline

Combination and Black Box Procedures

- Asymmetric extension of the N-O schema
  An extension of the Nelson-Oppen method for combining *polite* theories (subsuming stably infiniteness) and an arbitrary one (e.g., enumerated data types) ***[RRZ05]***

  ➽ beyond Stable Infinite
- Symmetric extension of the N-O schema
  An extension of the Nelson-Oppen method for combining theories in which satisfiability in arbitrary models and satisfiability in infinite models are decidable problems ***[BGN+06]***

  ➽ beyond Stable Infinite

Combination and Rewriting-based Procedures

- If Superposition is a decision procedures for T_1 and T_2, separately, it is so for the union of T_1 and T_2 under reasonable assumptions ***[ABRS05]***

  ➸ modularity of termination

- Rewriting-based methods for automatically checking whether a theory T is either stably-infinite ***[KRRT06]*** or admits only finite models ***[BGN$^+$06]***

  ➸ automatically checking Stable Infiniteness

- Rewriting-based methods for computing the entailed formulae to be exchanged by the Nelson-Oppen combination method ***[KRRT05]***

  ➸ deduction completeness

Decision Procedures for Data Structures

- Problem: data structures are often axiomatized with non-disjoint unions of arbitrary theories (i.e., possibly non-stably infinite)
- Our solution: new techniques (based on instantiation strategies) to avoid certain problems preventing the applicability of Nelson-Oppen method
  - decision procedures for various extensions of the theory of arrays (dimension, sortedness, ...) ***[GNRZ06]***

    ➸ beyond disjointness
  - decision procedure for a theory of pointers parametric in the theory of elements stored in memory cells ***[RZ06]***

    ➸ beyond disjointness and stably infiniteness

Integration of Boolean solvers and Decision Procedures

- Recall that a Boolean solver is available because of the SMT architecture...
- ***Key idea***: use the Boolean solver for the case-splitting on the derived disjunction of equalities between shared variables
- ***How***: every time a clause is derived by a procedure, pass it to the Boolean solver...
- For a similar approach (with less demanding requirements on the procedures being combined) see ***[BBC+05, BBC+06]*** on "*Delayed Theory Combination*"

What about certification ?

- New version of haRVey (based on the integration of a SAT solver and congruence closure) for proof reconstruction into ISABELLE ***[FMM+06]***
  - Combination of
    - ★ proof reconstruction from SAT solver
    - ★ proof reconstruction from congruence closure algorithm
- New prover: Ergo, based on a new combination method (congruence closure and Linear Arithmetic) for (partial) proof reconstruction in Coq
  - Neat and small architecture (3000 lines of Ocaml, close to inference rules)

What about certification ?

- New version of haRVey (based on the integration of a SAT solver and congruence closure) for proof reconstruction into ISABELLE ***[FMM+06]***
  - Combination of
    - ★ proof reconstruction from SAT solver
    - ★ proof reconstruction from congruence closure algorithm
- New prover: Ergo, based on a new combination method (congruence closure and Linear Arithmetic) for (partial) proof reconstruction in Coq
  - Neat and small architecture (3000 lines of Ocaml, close to inference rules)

haRVey after GECCOO

- Two incarnations...
  - haRVey-FOL (BDD/SAT, E prover, Linear Arithmetic Dec Proc, ...): in collaboration with D. Dèharbe (Natal, Brazil)
  - haRVey-SAT (SAT, congruence closure, ...): in collaboration with P. Fontaine and S. Merz (Nancy, France)
  - Available at `http://harvey.loria.fr`
- Better trade-offs between expressiveness and scalability: haRVey-FOL
- Towards certification of results: haRVey-SAT
- Future: merge the two by a hierarchic approach (partly elaborated in ***[KRRT05]***)

haRVey after GECCOO

- Two incarnations...
  - haRVey-FOL (BDD/SAT, E prover, Linear Arithmetic Dec Proc, ...): in collaboration with D. Dèharbe (Natal, Brazil)
  - haRVey-SAT (SAT, congruence closure, ...): in collaboration with P. Fontaine and S. Merz (Nancy, France)
  - Available at `http://harvey.loria.fr`
- Better trade-offs between expressiveness and scalability: haRVey-FOL
- Towards certification of results: haRVey-SAT
- Future: merge the two by a hierarchic approach (partly elaborated in *[KRRT05]*)

haRVey after GECCOO

- Two incarnations...
  - haRVey-FOL (BDD/SAT, E prover, Linear Arithmetic Dec Proc, ...): in collaboration with D. Dèharbe (Natal, Brazil)
  - haRVey-SAT (SAT, congruence closure, ...): in collaboration with P. Fontaine and S. Merz (Nancy, France)
  - Available at `http://harvey.loria.fr`
- Better trade-offs between expressiveness and scalability: haRVey-FOL
- Towards certification of results: haRVey-SAT
- Future: merge the two by a hierarchic approach (partly elaborated in ***[KRRT05]***)

haRVey after GECCOO

- Two incarnations...
  - haRVey-FOL (BDD/SAT, E prover, Linear Arithmetic Dec Proc, ...): in collaboration with D. Dèharbe (Natal, Brazil)
  - haRVey-SAT (SAT, congruence closure, ...): in collaboration with P. Fontaine and S. Merz (Nancy, France)
  - Available at `http://harvey.loria.fr`
- Better trade-offs between expressiveness and scalability: haRVey-FOL
- Towards certification of results: haRVey-SAT
- Future: merge the two by a hierarchic approach (partly elaborated in ***[KRRT05]***)

Outline

Future work

- New combination methods: investigate relationships between politeness and rewriting-based procedures with the aim of finding automatic methods to characterize polite theories
- Heap allocated data structures:
  - generalize to arbitrary recursive data structures, such as trees, DAGs
  - incorporate resource reasoning (time and space consumption)
  - incorporate local reasoning *à la* Separation Logic
- Model checking infinite state systems: integration of standard algorithms for LTL model checking and (combination of) decision procedures for theories of data structures
- Standardization for interface and cooperation mechanisms of reasoning tools around the SMT-LIB initiative?

Future work

- New combination methods: investigate relationships between politeness and rewriting-based procedures with the aim of finding automatic methods to characterize polite theories
- Heap allocated data structures:
  - generalize to arbitrary recursive data structures, such as trees, DAGs
  - incorporate resource reasoning (time and space consumption)
  - incorporate local reasoning *à la* Separation Logic
- Model checking infinite state systems: integration of standard algorithms for LTL model checking and (combination of) decision procedures for theories of data structures
- Standardization for interface and cooperation mechanisms of reasoning tools around the SMT-LIB initiative?

Future work

- New combination methods: investigate relationships between politeness and rewriting-based procedures with the aim of finding automatic methods to characterize polite theories
- Heap allocated data structures:
  - generalize to arbitrary recursive data structures, such as trees, DAGs
  - incorporate resource reasoning (time and space consumption)
  - incorporate local reasoning *à la* Separation Logic
- Model checking infinite state systems: integration of standard algorithms for LTL model checking and (combination of) decision procedures for theories of data structures
- Standardization for interface and cooperation mechanisms of reasoning tools around the SMT-LIB initiative?

Future work

- New combination methods: investigate relationships between politeness and rewriting-based procedures with the aim of finding automatic methods to characterize polite theories
- Heap allocated data structures:
  - generalize to arbitrary recursive data structures, such as trees, DAGs
  - incorporate resource reasoning (time and space consumption)
  - incorporate local reasoning *à la* Separation Logic
- Model checking infinite state systems: integration of standard algorithms for LTL model checking and (combination of) decision procedures for theories of data structures
- Standardization for interface and cooperation mechanisms of reasoning tools around the SMT-LIB initiative*?*


A. Armando, M. P. Bonacina, S. Ranise, and S. Schulz.
On a rewriting approach to satisfiability procedures: extension, combination of theories and an experimental appraisal.
In B. Gramlich, editor, *Proc. of the 5th Int. Workshop on Frontiers of Combining Systems (FroCoS'2005)*, volume 3717 of *Lecture Notes in Computer Science*, pages 65–80, Vienna, Austria, September 2005. Springer.


A. Armando, S. Ranise, and M. Rusinowitch.
A Rewriting Approach to Satisfiability Procedures.
Journal of Information and Computation — Special Issue on Rewriting Techniques and Applications (RTA'01), 183(2):140–164, June 2003.


M. Bozzano, R. Bruttomesso, A. Cimatti, T. Junttila, P. van Rossum, S. Ranise, and R. Sebastiani.
Efficient Satisfiability Modulo Theories via Delayed Theory Combination.

In Springer, editor, *Proc. of the Int. Conf. on Computer-Aided Verification (CAV 2005)*, number 3576 in Lecture Notes in Computer Science, Edinburg, Scotland (UK), 2005.

M. Bozzano, R. Bruttomesso, A. Cimatti, T. Junttila, P. van Rossum, S. Ranise, and R. Sebastiani.
Efficient Theory Combination via Boolean Search.
Journal of Information and Computation, 10(204):1411–1596, 2006.
Special Issue on Combining Logical Systems.

M. P. Bonacina, S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli.

Decidability and Undecidability Results for Nelson-Oppen and Rewrite-Based Decision Procedures.
In *Proc. of the 3rd Int. Conference on Automated Reasoning (IJCAR)*, number 4130 in LNAI, pages 513–527, Seattle, WA, USA, August 2006.

D. Déharbe and S. Ranise.

Light-Weight Theorem Proving for Debugging and Verifying Units of Code.
In *Proc. of the International Conference on Software Engineering and Formal Methods (SEFM03)*, Brisbane, Australia, September 2003. IEEE Computer Society Press.


P. Fontaine, J.-Y. Marion, S. Merz, L. Prensa Nieto, and A. Tiu.
Expressiveness + automation + soundness: Towards combining SMT solvers and interactive proof assistants.
In *TACAS*, volume 3920 of *Lecture Notes in Computer Science*, pages 167–181. Springer-Verlag, 2006.


S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli.
Deciding Extensions of the Theory of Arrays by Integrating Decision Procedures and Instantiation Strategies.
In *Proc. of the 10th European Conference on Logics in Artificial Intelligence (Jelia)*, number ???? in LNCS, September 2006. Extended version with full proofs, motivations, and examples in a technical report available at
`http://www.loria.fr/~ranise/pubs`.


H. Kirchner, S. Ranise, C. Ringeissen, and D.-K. Tran.
On Superposition-Based Satisfiability Procedures and their Combination.
In Dang Van Hung and Martin Wirsing, editors, *Proc. of the 2nd International Conference on Theoretical Aspects of Computing (ICTAC'05)*, volume 3722 of *Lecture Notes in Computer Science*, pages 594–608, Hanoi, Vietnam, October 2005. Springer.


Hélène Kirchner, Silvio Ranise, Christophe Ringeissen, and Duc-Khanh Tran.
Automatic Combinability of Rewriting-Based Satisfiability Procedures.
In *Proc. of the 13th Int. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, To appear in Lecture Notes in Artificial Intelligence, Phnom Penh, Cambodia, November 2006. Springer.


Silvio Ranise, Christophe Ringeissen, and Duc-Khanh Tran.

Nelson-Oppen, Shostak and the Extended Canonizer: A Family Picture with a Newborn.
In Keijiro Araki and Zhiming Liu, editors, *First International Colloquium on Theoretical Aspects of Computing - ICTAC 2004*, volume 3407 of *Lecture Notes in Computer Science*, pages 372–386, Guiyang, Chine, September 2004. Springer. Post-event proceedings. Also available as Research Report A04-R-193, LORIA, France.

S. Ranise, C. Ringeissen, and C. G. Zarba.
Combining data structures with nonstably infinite theories using many-sorted logic.
In B. Gramlich, editor, *Proc. of the 5th Int. Workshop on Frontiers of Combining Systems (FroCoS'2005)*, volume 3717 of *Lecture Notes in Computer Science*, pages 48–64, Vienna, Austria, September 2005. Springer.

S. Ranise and C. Zarba.
A Theory of Singly-Linked Lists and its Extensible Decision Procedure.

In *Proc. of the 4th IEEE International Conference on Software Engineering and Formal Methods (SEFM)*, Pune, India, September 2006. IEEE Computer Society Press.