SMT: Past, Present & Future

Clark Barrett
New York University

Leonardo de Moura
Microsoft Research

Silvio Ranise
FBK Trento

Aaron Stump
The University of Iowa

Cesare Tinelli
The University of Iowa

HVC 2010, Haifa, Oct 7, 2010

Clark Barrett

Leo de Moura

Silvio Ranise

Aron Stump

Cesare Tinelli

Many Thanks to

- The HVC Award Committee
- All of our students and collaborators
- Our colleagues in the SMT community

Formal Verification and Logic

- Formal verification requires checking the satisfiability of formulas in some symbolic logic

- Often, the logic is propositional

  $\neg p \wedge (q \vee r) \Rightarrow s, \quad \Box p \Rightarrow \Diamond s, \quad qUr \wedge Gq$

- In many cases, it is first-order

  $(p(x) \wedge x > 3) \Rightarrow y + x = 2, \quad f(x,a) = g(y)$

Satisfiability Modulo Theories

- In the first-order case, we are not interested in satisfiability in arbitrary models
- But in those that fix the interpretation of certain predicate and function symbols
  (=, <, +, 3, cons, cdr, read, write, …)
- We are interested in satisfiability modulo a certain theory of these symbols (SMT)

Satisfiability Modulo Theories

$$b + 2 = c \ \ and \ \ f(read(write(a,b,3), c\text{-}2) \neq f(c\text{-}b+1)$$

Satisfiability Modulo Theories

$$\boxed{b + 2 = c} \text{ and } f(read(write(a,b\boxed{,3}), \boxed{c\text{-}2}) \neq f\boxed{(c\text{-}b+1})$$

Arithmetic

Satisfiability Modulo Theories

$$b + 2 = c \;\; and \;\; f(\boxed{read}(\boxed{write}(a,b,3), c\text{-}2) \neq f(c\text{-}b+1)$$

Array Theory

Satisfiability Modulo Theories

$$b + 2 = c \;\; and \;\; \boxed{f}(read(write(a,b,3), c\text{-}2) \neq \boxed{f}(c\text{-}b+1)$$

Uninterpreted Functions

Satisfiability Modulo Theories

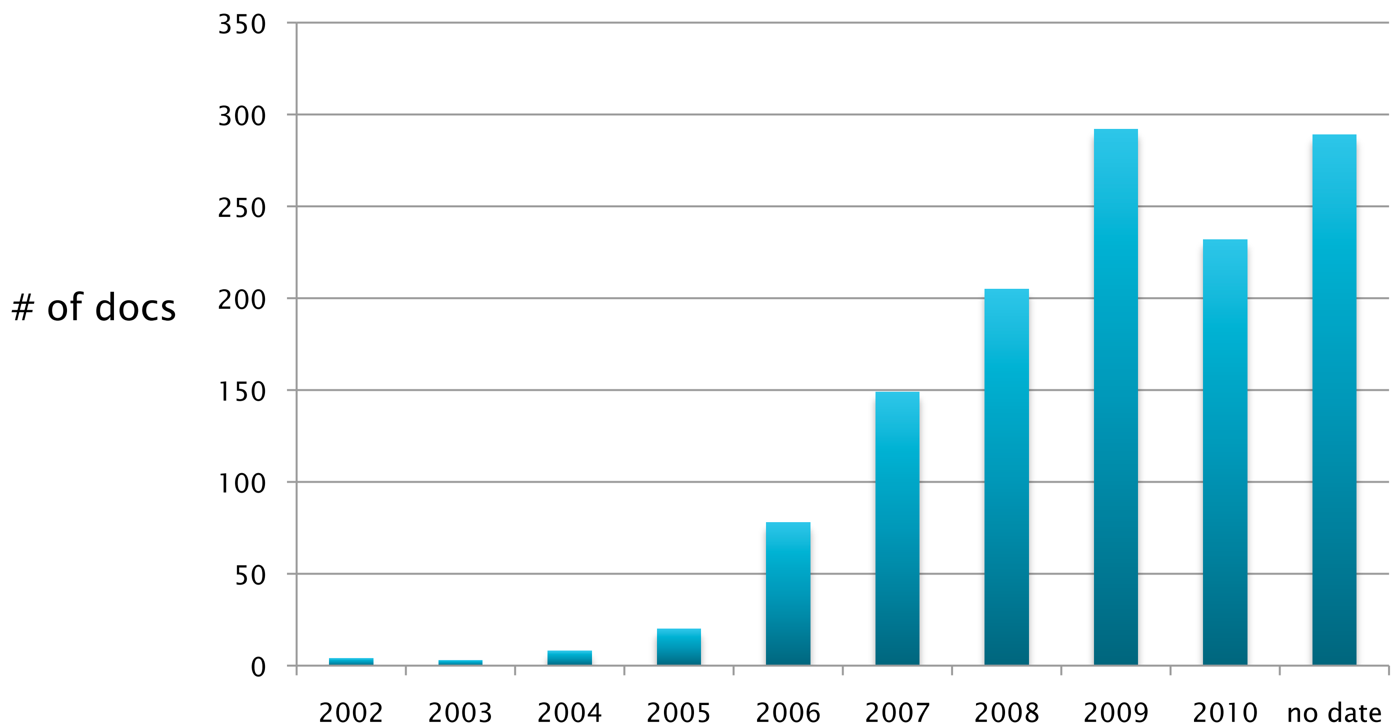
- **Problem:** traditional deduction techniques for FOL are inadequate for SMT:
  - some theories are not finitely axiomatizable
  - general FOL calculi are not efficient enough
- **Fact:** the satisfiability of sets of literals is decidable, efficiently, in several theories
- **Catch:** checking the satisfiability of qffs is at least as hard as in the propositional case

Sources of SMT Success

The current success of SMT derives from

1. A long line of old and new efficient decision procedures for many theories
2. Spectacular advances in SAT solving
3. Smart new ways of combining 1 and 2
4. A substantial standardization effort
5. A large set of applications waiting in the wings

An Explosion of Interest in SMT



- 2002: birth year of the term SMT
- Google Scholar entries per year for "SMT Satisfiability Modulo Theories" in Engineering, CS and Math

Where are SMT solvers now used?

- processor verification
- equivalence checking
- (un)bounded model checking
- predicate abstraction
- static analysis
- symbolic execution
- automated test case generation
- extended static checking
- scheduling and optimization
- …

SMT just at Microsoft

- The Spec# Programming System
- HAVOC
- ForμLa
- Hyper-V Microsoft Virtualization
- F7
- Terminator T-2
- VCC
- SLAM
- Yogi
- NModel
- Vigilante
- SpecExplorer
- Pex
- SAGE

SMT Prehistory [late '70s to '80s]

- Pioneers:
  - R. Boyer, J Moore, G. Nelson, D. Oppen, R. Shostak
- Influential results:
  - N&O congruence closure procedure
  - N&O combination method
  - Shostak combination method
- Influential systems:
  - Nqthm prover [Boyer & Moore]
  - Simplify [Nelson et al.]

SMT Beginnings [late 1990s]

- Game changer:
  - Advances in SAT
  - Very fast solvers (Chaff, Berkmin, …)
- Main new ideas:
  - "eager" encodings of SMT problems into SAT
    [Bryant, Velev, Strichman, Lahiri, Seisha,…, –'02]
  - "lazy" encodings into SAT + decision procedures
    [Armando et al.'00, Audemard et al.'02, Ruess & de Moura'02, Barrett et al.'02]

State of the art on SMT in 2002

- Many different solvers
  - based on different variants of FOL
  - working with different theories
  - dealing with different classes of formulas
  - having different interfaces and input formats
- Solver's theory unclear
- Arduous to assess the relative merits of techniques or solvers
- Each solver good on its own benchmarks
- Difficult even to a evaluate a single solver

FroCoS 2002: a call for arms

- G. Nelson gives invited talk on Simplify's work
- Excitement about the promise of the field
- Unhappiness about lack of standard benchmarks
- Chair A. Armando calls for the creation of a common library of benchmarks
- SR and CT agree to lead the initiative
- Several participants promise assistance and contributions

FroCoS 2002 aftermath

- R&T soon realize that a common library would need to develop a standard:
  1. background logic
  2. catalog of rigorously defined theories
  3. specification of relevant fragments of these theories
  4. concrete syntax for benchmarks

- This becomes the blueprint for the SMT-LIB initiative

The SMT-LIB Initiative

International effort supported by several research groups worldwide

The SMT-LIB Initiative

- Goals:
  - Collect a large on-line library of SMT benchmarks
  - Promote the adoption of common languages and interfaces for SMT solvers
- Sister initiatives:
  - SMT-COMP, solver competition
  - SMT-EXEC, solver execution service
- Funding:
  - NSF, SRC, Intel, Microsoft, U. of Iowa

The SMT-LIB Initiative today

- 94,000+ benchmarks in online repository
- 22 logics
- SMT-LIB format (V. 1.2) adopted by all major SMT solvers (12+)
- Version 2, major new version, of SMT-LIB format and library released in 2010
- SMT-COMP'10 run with Version 2.0

SMT-LIB Format

Three main components:

1. Theory declarations, semi-formal specifications of background theories of interest (e.g., integers, reals, arrays, bit vectors, . . .)
2. Logic declarations, semi-formal specifications of fragments of (combinations of) background theories (e.g., quantifier-free linear real arithmetic, integer difference constraints, . . .)
3. Benchmarks, formulas to be checked for satisfiability (previously), or scripts (now)

SMT-LIB Repository

Three main components:

1. Catalog of theory declarations
2. Catalog of logic declarations
3. Library of benchmarks

www.smt-lib.org

SMT-LIB 2 format highlights

- Command language
  - Allows more sophisticated interaction with solvers
  - Stack-based, tell-and-ask execution model
  - Benchmarks are command scripts
- Concrete syntax
  - Sublanguage of Common Lisp S-expressions
  - Few syntactic categories
- Powerful underlying logic
  - Many-sorted FOL with (pseudo-)parametric sorts
  - Function symbol overloading

SMT-LIB chronology

SMT-LIB chronology

Aug 2002: Initial website, SMT-LIB is born

Sep-Dec 2002: Email discussion on SMT-LIB standard led by SR, CT

- initial feedback by A. Armando, CB, G. Nelson, H. Ruess, N. Shankar, AS

Oct 2002: A few external subsites, with benchmarks in different formats

- by SR, O. Strichman, AS

SMT-LIB chronology

Jul 2003: White paper on SMT-LIB standard
- drafted and circulated by SR, CT

Aug 2003: First PDPAR workshop, with panel on SMT-LIB standard
- organized by SR, CT
- panelists: CB, G. Nelson, R. Sebastiani, G. Sutcliffe, AS

Jul 2004: Version 1 of SMT-LIB standard
- written and released by SR, CT

SMT-LIB chronology

Jul 2004: SMT-LIB panel at PDPAR

- Call for a solver competition by A. Armando
- CB, LdM, AS agree to organize SMT-COMP in 2005

Aug 2004-Oct 2004 Several rounds of discussion on SMT-COMP'05

- by CB, LdM, SR, AS, CT
- major feedback from A. Armando and A. Cimatti

SMT-LIB chronology

Sep 2004–Apr 2005 Lots of work by all on

- refining the SMT-LIB format, into Version 1.1
- defining an initial set of theories and logics
- collecting existing benchmarks in other formats
- translating them into the SMT-LIB format
- producing some utility tools for the community

Apr 2005: First version of SMT-LIB repository

- 11 logics
- 1,350 benchmarks

SMT-LIB chronology

Jul 2005 First SMT-COMP

- organized by CB, LdM, AS
- 12 solvers, 7 divisions

Jul 2005: PDPAR

- chaired by A. Cimatti, A. Armando
- E. Singerman calls for SMT solvers to support bit vectors

Jan-May 2006: work on defining an SMT-LIB theory of bit vectors

- by SR, CT, with major feedback from CB, LdM, AS

SMT-LIB chronology

Jan-May 2006: Thousands of contributed benchmarks translated and added to SMT-LIB by CB, LdM

Aug 2006: Version 1.2 of SMT-LIB format released by SR, CT

Aug 2006: SMT-COMP organized by CB, LdM, AS

- 11 divisions, including one on bit vectors
- 42,100 benchmarks
- 12 solvers (4 new)

SMT-LIB chronology

Jun 2007: SMT-EXEC cluster set up by AS

Jul 2007: PDPAR workshop renamed SMT
- chaired by S. Krstic, A. Oliveras
- SMT-LIB panel discusses commands and parametric type extensions to format

Jul 2007: SMT-COMP runs on SMT-EXEC cluster
- organized by CB, M. Deters, A. Oliveras, AS
- live results with a fancy interface by M. Deters
- 55,400 benchmarks from 12 divisions
- 9 solvers (4 new)

SMT-LIB chronology

Jan 2008: CB, AS, CT create 3 workgroups

- each on a major improvement to SMT-LIB format

Jan 2008: SMT-EXEC open to public use

Jul 2008: SMT workshop chaired by CB, LdM

- record attendance (75)

Jul 2008: SMT-COMP

- organized by CB, M. Deters, A. Oliveras, AS
- 70K benchmarks from 12 divisions
- 13 solvers

SMT-LIB chronology

May 2009: SMT workshop gets Steering Committee and bylaws

- bylaws edited by CT with input from past PC chairs

Jul 2009: Web-based query facility for SMT-LIB repository

- by M. Deters, with inputs from CB, CT

Aug 2009: Draft of Version 2 of SMT-LIB format posted to the community

- produced by 3 workgroups led by CB, AS, CT, resp.

SMT-LIB chronology

Aug 2009: SMT'09 largest workshop at IJCAR

- 60 registrants
- chaired by B. Dudertre, O. Strichman

Mar 2010: SMT-LIB Version 2 document officially released by CB, AS, CT

May 2010: SMT-LIB benchmarks (90K+) ported to Version 2 by CB, C. Conway, M. Deters

SMT-LIB chronology

July 2010: SMT'10 largest workshop at FLoC

- chaired by A. Gupta & D. Kroening
- 65 registrants

July 2010: SMT-COMP uses SMT-LIB 2

- organized by CB, M. Deters, A. Oliveras, AS
- 94K benchmarks in 18 divisions
- 10 solvers

Oct 2010: HVC 2010 Award!

SMT-LIB immediate future

- Fresh blood in SMT-COMP
  - 2011,12 by R. Bruttomesso, M. Deters, A. Griggio
- SMT-LIB tutorial, by D. Cok
- Formalization contributions by the community
  - a theory of floating point arithmetic, by P. Ruemmer, T. Wahl, et al.
  - several theories of container data structures, by P. Ruemmer, CT, et al.

    (lists with length, finite maps, finite sets with cardinality)
  - a theory of character strings, by V. Ganesh et al.

SMT-LIB Future

- Benchmarks with more complex scripts
- An expanded command language
- An extension of the format with algebraic data type declarations
- A common standard for SMT proofs
  - based on an extension of LF, by AS, CT
- More logics

SMT Future

- Bit vector solvers dynamically combining algebraic reasoning and reduction to SAT
- Novel FP arithmetic solvers
- Non-linear integer/real arithmetic solvers

SMT Future

- Proof-production
  - proofs of unsatisfiable queries

- Interpolation
  - interpolants of unsat queries $F \wedge G$

- Projections
  - given $F(\mathbf{x},\mathbf{y})$, producing a (suitable over approxim.) of $\exists \mathbf{x}\, F(\mathbf{x},\mathbf{y})$

SMT Future

- Quantifiers, quantifiers, quantifiers
  - needed in some proof obligations
  - used to formalize non-built-in theory symbols
- Inductive reasoning on functions over algebraic data types

Thank you!