

SC² challenges: when Satisfiability Checking and Symbolic Computation join forces

James H. Davenport, Pascal Fontaine, Alberto Griggio
(and the others partners whose details wouldn't fit)¹
University of Bath, LORIA, Fondazione Bruno Kessler
J.H.Davenport@bath.ac.uk,
Pascal.Fontaine@loria.fr, Griggio@FBK.eu

6 August 2017

¹Thanks to EU H2020-FETOPEN-2016-2017-CSA project SC² (712689):
www.sc-square.org

SC² (www.sc-square.org) is a European Network to bridge the two communities of Symbolic Computation (Bath, Coventry, Genoa, Kassel, Maplesoft, RISC Linz) and Satisfiability Checking (Aachen, FBK, LORIA, MPI, Oxford). July 2016–August 2018, with three workshops (Timișoara, Kaiserslautern and Oxford: 2018 with FLoC) and a Summer School (50 attendees).

Two communities often solving similar problems, but with different methodologies (algebra first versus logic first), different proxies for efficiency (complexity theory versus contests) and different languages (“algebra” versus “arithmetic”; “existentially quantified” versus “quantifier-free”)

Challenge 1: Ordering of Variables

Theory Variables versus Logic Variables

SAT-solvers have been moving towards ever more frequent restarts, and concomitant reordering of the logic variables. Conversely, symbolic computation, particularly when reasoning over \mathbf{R}^n or \mathbf{C}^n strongly prefers a fixed order of the theory variables. **How to reconcile?**

A related issue is that simplistic SMT has no idea of the links between the logic variables and the theory variables. How should they be connected?

Challenge 2: Inexpensive Theory Deductions

If one of the theory constructs is $x^2 + y^2 < 1$, then the theory can deduce $x > -1 \wedge x < 1$, which are linear constraints in fewer variables. **But** these would be new literals as far as the logic solver was concerned.

- 1 Does this belong in a pre-processor?
- 2 When is such a constraint “simpler”?
- 3 How much of this should one do?

Challenge 3: Weakly Nonlinear Reasoning

Symbolic Computation systems are typically geared for “hard” problems. But how far can we get by working with abstractions expressed over linear arithmetic with uninterpreted functions, where nonlinear multiplication is modeled as an uninterpreted function. See [CGI⁺17]. More recent work has extended this to other functions such as sin.


What are the limitations of these techniques?


Challenge 4: Combining Decision Procedures

Its doubly-exponential complexity restricts the practical applicability of the CAD method, the only available complete decision procedure for real arithmetic. A unique feature of the SMT-RAT solver [CKJ⁺15] is that it allows the user to define her own strategic combination of decision procedures, optimised towards the given problem type, and with the possibility to exploit parallelisation.

How can this be extended, and how can the burden move from the user? To heuristics?

- Dialogue between systems is needed, and needs to be richer
- Dialogue between individuals is being fostered in SC²
- Dialogue between methodologies needs to take place
- Dialogue between communities needs to take place

-  A. Cimatti, A. Griggio, A. Irfan, M. Roveri, and R. Sebastiani.
Invariant Checking of NRA Transition Systems via Incremental Reduction to LRA with EUF.
In *Proceedings TACAS 2017*, volume 10205 of *LNCS*, pages 58–75. Springer, 2017.

-  F. Corzilius, G. Kremer, S. Junges, S. Schupp, and E. Ábrahám.
SMT-RAT: An open source C++ toolbox for strategic and parallel SMT solving.
In *Proceedings SAT 2015*, volume 9340 of *LNCS*, pages 360–368. Springer, 2015.