Future integration of Symbolic Computation and Satisfiability Checking

James Davenport[1]
University of Bath
J.H.Davenport@bath.ac.uk

3 June 2016

[1]Thanks to Matthew England (Coventry) and many others, EPSRC EP/J003247/1, EU $\mathcal{SC}^2$ project 712689

Satisfiability Checking — SAT

Given a formula $\mathcal{P}$ in propositional logic, normally in Conjunctive Normal Form (CNF)

$$(l_{1,1} \vee l_{1,2} \vee \cdots) \wedge (l_{2,1} \vee l_{2,2} \vee \cdots) \wedge \cdots$$

where the $l_{i,j}$ are either p_k or $\neg p_k$ for a set of propositional variables $\{p_k\}$, either find a satisfying assignment of true/false to the p_k, or state (correctly!) that no such exists.

Theory
: 3-SAT is NP-complete

Practice
: "real-world" examples with millions of clauses are solved in competitions, and it's hard to produce hard examples [Spe15]

Eclipse
: uses a SAT solver to resolve package dependencies

BMW etc.
: use SAT solvers to configure cars for customers [KS00]: $>$2M (unknowing) SAT users/year

Many tricks
: especially lemma generation

Instead of the $\{p_k\}$ being free Booleans, they are elements of some underlying theory (for us, polynomials over $\mathbf{R}$ with $=$, $>$ etc.) with underlying variables x_i, and it is the x_i whose values we seek.
So $p_1 \wedge p_2$ is solvable for free Boolean p_k, but not when $p_1 \equiv x < 0$ and $p_2 \equiv x > 1$.
When $p_1 \equiv x > 0$ and $p_2 \equiv x < 1$ it is solvable with, say, $x = 1/2$.

Quantifiers?

Showing satisfiability of $\mathcal{P}$ is answering (positively) $\exists x_1, \ldots, x_n \mathcal{P}(x_1, \ldots, x_n)$, and unsatisfiability is answering it negatively, or answering $\forall x_1, \ldots, x_n \neg\mathcal{P}(x_1, \ldots, x_n)$ positively.

In many verification applications, $\mathcal{P}$ is "a dangerous state", and we want to answer $\forall x_1, \ldots, x_n \neg\mathcal{P}(x_1, \ldots, x_n)$ positively, i.e. "the system cannot enter a dangerous state" *preferably with a proof*

One technique [JdM12] essentially drives the CAD algorithm backwards: looking for a counter-example

$\exists x_1 \ldots \forall y_1 \ldots \mathcal{P}(x_1, \ldots, y_1, \ldots)$ is considered in [CAR$^+$15, RDK$^+$15], but seems to require more technology

This is "there exists a solution such that the system cannot enter a dangerous state" — "we haven't painted ourselves into a corner"

Why do modern SAT-solvers do so well?

- A lot of attention to constant(ish) factors.
- Even more attention to $O(n)$ factors (watched literals)
- A great deal of accumulated heuristics
- Keep reformulating the problem (re-ordering the p_i) [HH10], e.g. every 100, 100,200,100,100,200,400,... [LSZ93] deductions we restart

But keeping track of "useful" (i.e. short) lemmas learned

(which potentially invalidates the argument for [LSZ93])

Lemmas?

In principle, there's much more scope for lemma-learning in (polynomial) SMT than straight SAT: for example $x^2 + y^2 \leq 1 \Rightarrow (x \geq -1) \wedge (x \leq 1)$, but the only use currently made of this sort of reasoning in mainstream CAD is $(f = 0 \wedge g = 0) \Rightarrow \mathrm{res}_{x_n}(f, g) = 0$ (extensions in [DE16])

- Q what is a "useful" lemma in this context?
- Q Is there an equivalent of "short"?
- Q Research shows utility (75+% of the time) of $(f = 0 \wedge g > 0) \Rightarrow (\hat{g} > 0)$ where $\hat{g}$ is the Gröbner reduction of g by f

CAD is very dependent on the order of the x_k, with significant research, and effort at runtime, going into choosing the "best" order [DSS04, HEW$^+$14]

On the other hard, SAT solvers frequently [HH10] reorder the p_i, received wisdom being that there is no *a priori* best order

Q What is the relationship between the order of the p_i and the x_k?

Questions?

C.-H. Cheng, L. Astefanoaei, H. Ruess, S. Ben Rayana, and S. Bensalem.
Timed Orchestration for Component-based Systems.
https://arxiv.org/abs/1504.05513, 2015.

J.H. Davenport and M. England.
Need Polynomial Systems be Doubly-exponential?
https://arxiv.org/abs/1605.02912, 2016.

A. Dolzmann, A. Seidl, and Th. Sturm.
Efficient Projection Orders for CAD.
In J. Gutierrez, editor, *Proceedings ISSAC 2004*, pages 111–118, 2004.

Z. Huang, M. England, D. Wilson, J.H. Davenport, L.C. Paulson, and J. Bridge.
Applying machine learning to the problem of choosing a heuristic to select the variable ordering for cylindrical algebraic decomposition.
In S.M.Watt *et al.*, editor, *Proceedings CICM 2014*, pages 92–107, 2014.

S. Haim and M. Heule.
Towards Ultra Rapid Restarts.
Technical Report Universities of New South Wales and Deflt, 2010.

D. Jovanović and L. de Moura.
Solving Non-Linear Arithmetic.
In *Proceedings IJCAR 2012*, pages 339–354, 2012.

Bibliography III

W. Küchlin and C. Sinz.
Proving Consistency Assertions for Automotive Product Data Management.
J. Automated Reasoning, 24:145–163, 2000.

M. Luby, A. Sinclair, and D. Zuckerman.
Optimal Speedup of Las Vegas algorithms.
Inf. Proc. Letters, 47:173–180, 1993.

A. Reynolds, M. Deters, V. Kuncak, C. Tinelli, and C. Barrett.
On Counterexample Guided Quantifier Instantiation for Synthesis in CVC4.
`http://arxiv.org/pdf/1502.04464v3.pdf`, 2015.

I. Spence.
Weakening Cardinality Constraints Creates Harder Satisfiability Benchmarks.
J. Exp. Algorithmics Article 1.4, 20, 2015.