The (Un)Reasonable (In)Effectiveness of Theory for Satisfiability Solvers

Sam Buss

Casa Matemática Oaxaca
August 2018

Reasonable Effectiveness of Theory

Theory: Resolution can require exponential size proofs.

- Some clause sets require exponential size proofs, e.g., the pigeonhole principle [Haken'85]
- Some satisfiable clause set are conjectured to have hard-to-find satisfying assignments;
  e.g., Expressing an integer is composite, or inverting a hard one-way function.
- Therefore, it is impossible find resolution proofs feasibly in general. They can be too long!
- And, is expected to be hard to find satisfying assignments in many cases. For *any* feasible search procedure!

UnReasonable InEffectiveness of Theory

Nonetheless: Practical SAT Solvers are able to almost routinely solve instances of SAT with 100,000's or 1,000,000's of variables. Including instances that are of interest for industrial applications.

We have no good understanding of *why* CDCL Sat Solvers can be so effective.

[Un?]Reasonable Ineffectiveness of Theory

Theory: Frege and extended Frege are more efficient than resolution.

- Some clauses have much shorter Frege or extended Frege proofs than resolution proofs.
  E.g., the pigeonhole principle [Cook-Reckhow'79; B'87]
- But, SAT Solvers have so far been unable to make find strategies for introducing new variables via the extension rule which perform better than searching for resolution proofs (without new variables).

Reasonable [In?]Effectiveness of Theory

Theory: Cutting Planes are more Efficient than Resolution.

- Some clauses have much shorter cutting planes proofs than resolution proofs.
  E.g., the pigeonhole principle.
- But, SAT Solvers have so far been unable to use cutting planes more effectively than CDCL resolution proofs.
- Partial theoretical explanation by [Hooker'88,'92], [Vinyals-Elffers-GiráldezCru-Gocht-Nordström'18]:

Cutting planes provers as currently implemented for CNF inputs are polynomially simulated by CDCL.

Suggestion [VEGCGN'18]: Fully implement division in Sat Solvers.

[Un?]Reasonable Ineffectiveness of Theory

There is yet no good theory of effective/feasible proof search — at least that is applicable to SAT Solvers.

TFNP: Total NP Search Functions

- Very rich theory of TFNP, [Papadimitriou'91], many others.
- [Beckmann-B.'17; Papadimitriou-Goldberg'18] All common TFNP problems reducible to searching for an error in a Frege or stronger proof. ("Consistency search", "Wrong proof").
- TFNP has so far no contact with practical proof search problems.

More applicable to the theory of SAT Solvers is "Automatizability" (next slide...)

[Un?]Reasonable Ineffectiveness of Theory, p. 2

Automatizability: Hardness of finding polysize proofs

- Finding a shortest resolution proof is NP-hard. [Iwama'97]
- Cannot find resolution proofs of length within $2^{\log^{1-\epsilon} n}$ factor of optimal is unless $P = NP$.
  [Alekhnovich-B-Moran-Pitassi'98, Dinur-Safra'99]
- Cannot find proof in (tree-like) resolution of polynomially optimal length is hard
  (i.e., resolution is not polynomially automatizable)
  unless $W[P]$ is fixed-parameter tractable with a randomized algorithm. [Alekhnovich-Razborov'01]
- Frege, and TC^0-Frege, is not automatizable, unless there is a polynomial time algorithm for factoring Blum integers. [Bonet-Pitassi-Raz'2000]

This called an "ineffectiveness of theory" since the hardness results do not apply to industrial-style instances of SAT.

Challenge for Theory (or Practice)

Challenge: Develop a theory of proof search that can explain the relative efficiency of resolution proof search and the relative inefficiency of proof search in stronger systems.

or

Challenge: Develop better proofs search methods for systems stronger than resolution.

There is a Missing Link between Theory and Practice

Theory: Has identified many propositional proof systems for resolution based proof systems: Including: Tree-like resolution, Regular resolution, Input resolution, Linear resolution, Regular resolution, Pool resolution and Regwrti, General resolution, and Extended resolution.

Practice: A large number of effective resolution-based proof search methods, including Unit propagation, Davis-Putnam procedure, DPLL, Conflict directed clause learning (CDCL), Pure literals, Variable elimination (and addition), Maxsat, Restarts, etc.

But: There is almost no correspondence between theory's systems and practical systems.
Some exceptions are: (a) unit propagation and input (trivial) resolution. (b) DPLL and tree-like resolution.

Missing Link? (p. 2)

[Van Gelder'05] and [B-Hoffmann-Johannsen'08] identified two systems, *Pool resolution* and *Regwrti*, which are very closely related to the power of (non-greedy) CDCL proof systems.

But: common CDCL systems surpass Pool and Regwrti refutations in several ways

- Restarts. These are crucial to effectiveness of CDCL !!!
- Pure literals, variable elimination.
- Self-subsuming resolution inferences (potentially).
- RAT (Resolution Asymmetric Tautology introduction).

Challenge for theory: Separate the pool/Regwrti systems from resolution, or show polynomial simulations.

Missing Link? (p. 3)

[Pipatsrisawat-Derwiche'09], see also [Atersias-Fichte-Thurley'11], showed an exact correspondence between CDCL **with restarts** and general resolution.

But:

- It is clear that the reason CDCL proof search is so successful is not because of this correspondence with general resolution.
- RAT (Resolution Asymmetric Tautology) inferences are still beyond the power of CDCL with restarts and even beyond resolution.

So the "missing link" is still missing. We lack a good theoretical model for the proof system(s) implicitly used by Sat Solvers.

Verified refutations, (D)RAT inferences

Verification: The verification of Sat Solver results has become increasingly important.

- Verification of satisfiability is straightforward: the verification is the explicit satisfying assignment.
- Verification of unsatisfiability is much harder. The run of a *correct* program provides such a verification. But:
- Sat Solvers may contain bugs.
- Sat Solvers now incorporate many sophisticated reasoning steps based on disparate principles. These may interact in subtle ways to create incorrect proofs of unsatisfiability.

A major success: Verified solution of the Pathagorean Triples Problem based on verification of a 200TB DRAT proof. [Heule-Kullmann-Marek'16]

An Unexpected Effectivity of Theory

Verification of Unsatisfiability = Proof System

[Wetzler-Heule-Hunt'14, HHW'13] (and others) identified the "DRAT" system as a proof system (a verification system) that is

(a) Strong enough to encompass current CDCL SAT Solvers.

(b) Straightforward to generate and verify proof traces.

DRAT incorporates RAT inferences — which preserve (un)satisfiability, but not logical equivalence — and Deletion (D) which cannot create new unsatisfiability.

This is an extension of blocked clause inference [Kullmann'99].

Def'n: A RAT inference of clause C from Γ is permitted if

$$\Gamma_{|\alpha} \vDash_1 \Gamma_{|\tau}$$

where α is the partial assignment falsifying all literals in C and τ differs from α in exactly one value. This means that for each $D \in \Gamma$, there is a unit propagation refutation of $\Gamma_{|\alpha} \cup \overline{D_{|\tau}}$.

Theorem: DRAT is equivalent to (p-simulates) extended resolution.

Challenge to practice: Settle on one verification system (or a small number of systems) that serve as verification systems for unsatisfiability.

Write independent, and verified, implementations.

Question: Should these verification systems be as strong as extended resolution?

Currently there is a discrepancy between published descriptions of DRAT checkers and implemented DRAT checkers [RebolaPardo-Biere'18+]:

- To take advantage of 2-watched literal code, implemented DRAT checkers do not "undo" unit propagation after deletion operations.
- Implemented DRAT systems use multisets of clauses.

My opinion: The second is very reasonable; the first is questionable.

Question: Can "practice" settle on one or two stable verification systems?

Already there are other stronger verification systems that could be used.

- [Heule-Kiesl-Biere'17] introduce various "propagation redundancy" (PR) systems that provide additional flexibility over DRAT.
- [B-Thapen, i.p.] A strong "substitution redundancy" (SR) inference can also be used: (where it holds $\Gamma_{|\alpha} \vDash_1 (\Gamma \cup \{C\})_{|\tau}$ for τ an arbitrary substitution which is allowed to map variables to literals).
- Also possible to track a subset of Γ_0 of Γ such that $\Gamma_0 \vDash \Gamma$ and use instead $\Gamma_{|\alpha} \vDash_1 (\Gamma_0 \cup \{C\})_{|\tau}$ — this works for all of Blocked Clause, DRAT or propagation/substitution redundancy.

  (Γ_0 should not be confused with irredundant clauses.)

Question for practice: Is there a sufficiently useful verification system that does not depend on introducing new variables? E.g. propagation or substitution redundancy, PR or SR.
Question for theory and practice: Will this system be a missing link?
Question for theory: Do PR or SR without new variables p-simulate extended resolution?
All currently known hard examples for propositional proof systems can be handled by polynomial size PR (and SPR and SR) proofs that do not introduce new variables. This includes PHP, Parity, Clique-Coloring, Tseitin on expander graphs, xorification, orification. [Heule-Kiesl-Biere'17, B-Thapen'i.p.]

Conversely:

Question for practice: Can extension adding new variables be used *profitably* in SAT solvers?

A final challenge:

Moshe asked yesterday: Why does CDCL work so well on real-world benchmarks?

Counter-challenge for theory and practice: Can we find ways to tune or modify CDCL to work even better on real-world benchmarks?

Thank You!

Feedback/comments appreciated!!!!