A note on SAT algorithms and proof complexity

Jan Krajíček*

Faculty of Mathematics and Physics
Charles University in Prague

Abstract

We apply classical proof complexity ideas to transfer lengths-of-proofs lower bounds for a propositional proof system P into examples of hard unsatisfiable formulas for a class $\mathsf{Alg}(P)$ of SAT algorithms determined by P. The class $\mathsf{Alg}(P)$ contains those algorithms M for which P proves in polynomial size tautologies expressing the soundness of M. For example, the class $\mathsf{Alg}(F_d)$ determined by a depth d Frege system contains the commonly considered enhancements of DPLL (even for small d). Exponential lower bounds are known for all F_d. Such results can be interpreted as a form of consistency of P $\neq$ NP.

Further we show how the soundness statements can be used to find hard satisfiable instances, if they exist.

Keywords: Computational complexity.

We shall consider algorithms that take as inputs CNF formulas and always terminate. If the output $M(\varphi)$ of algorithm M on input φ is not a satisfying assignment for φ we shall call the computation **rejecting**. By a SAT algorithm we shall mean an algorithm that upon receiving a CNF formula computes a satisfying assignment, if one exists. We consider primarily deterministic algorithms but the whole construction works for algorithms whose set of rejecting computations is in NP. Such more general algorithms are a special case of propositional proof systems in the sense of Cook and Reckhow [11]. We could also allow non-uniform algorithms.

Denoting $Reject_M(w, \varphi)$ the property that w is a rejecting computation of M on φ and by $Sat(a, \varphi)$ the satisfiability relation between a truth assignment and a formula, the property that M is a SAT algorithm can be expressed by a universal statement

$$(\mathbf{M}) \qquad\qquad Sat(a, \varphi) \to \neg Reject_M(w, \varphi) \ .$$

*Supported in part by grants IAA100190902 and MSM0021620839. Also partially affiliated with the Institute of Mathematics of the Academy of Sciences and grant AV0Z10190503.

A Cook-Reckhow propositional proof system (a proof system for the set of propositional tautologies) mentioned earlier is simply any p-time relation $P(x, y)$ for which

$$\varphi \notin SAT \text{ iff } \exists y P(\neg\varphi, y)$$

(usually one allows any formula φ and not just CNF but one can also reduce to CNF via the so called limited extension, cf.[11, 14]). A SAT algorithm M can be then trivially thought of as a proof system P_M by defining

$$P_M(\tau, w) \text{ iff } Reject_M(w, \neg\tau) \ .$$

For example, if M is the simple DPLL algorithm [12, 13] then P_M is exactly the tree-like resolution (i.e. given an unsatisfiable formula, computations using different DPLL search strategies are in one-to-one correspondence with tree-like resolution refutations). Similar descriptions in terms of variants of resolution are known for several commonly considered enhancements of DPLL, cf. [3, 4, 6].

In this note we want to point out[1] that proof complexity offers another, somewhat more subtle, relation between SAT algorithms and proof systems based on interpreting the formulas (M) as soundness statements (aka reflection principles) about M. This is a specialization of a classical relation between the provability of reflection principles and simulations among proof systems, cf. [9, 17, 14]. Let us remark that proof complexity studies also other topics intimately linked with SAT algorithms, such as the existence of a p-optimal proof system or automatizability of proof search (cf.[8, 15, 19] for overviews and references). Here we shall restrict ourselves to reflection principles only.

Relevant background can be found in any of [10, 14, 19].

1 Reflection principles and the class $\mathsf{Alg}(P)$

For $k \geq n \geq 1$ and suitable $s = k^{O(1)}$ let

$$\mathsf{reject}^M_{k,n}(y, x, z)$$

with $y = (y_1, \ldots, y_k)$, $x = (x_1, \ldots, x_n)$ and $z = (z_1, \ldots, z_s)$ be **any** 3CNF formula such that for all $w \in \{0,1\}^k$ and $\varphi \in \{0,1\}^n$:

$$(\mathbf{F}) \qquad Reject_M(w, \varphi) \text{ iff } \mathsf{reject}^M_{k,n}(w, \varphi, z) \in SAT \ .$$

Such a formula exists by the NP-completeness of 3SAT as $Reject_M(w, \varphi)$ is a polynomial time relation. Similarly, for $n \geq 1$ and suitable $m = n^{O(1)}$ let

$$\mathsf{sat}_n(u, x, v)$$

with $u = (u_1, \ldots, u_n)$, $x = (x_1, \ldots, x_n)$ and $v = (v_1, \ldots, v_m)$ be a CNF formula such that for all $a, \varphi \in \{0,1\}^n$:

$$Sat(a, \varphi) \text{ iff } \mathsf{sat}_n(a, \varphi, v) \in SAT$$

[1]Actually I was pointing out these facts informally in writings or in lectures over the years but it appears that a formal presentation may be useful.

(we may need the extra variables v to put sat_n into CNF).

We want to stress that the choice of the formulas $\mathsf{reject}^M_{k,n}$ is completely arbitrary as long as the equivalence (F) holds. This is important as it substantially simplifies proofs of reflection principles (see below). On the other hand, the formulas sat_n need to be chosen with care, following the natural formalization of the satisfiability relation by induction on the logical complexity of φ. This is in order to be able to establish condition D4 in the next section.

For $k \geq n \geq 1$ define

$$\mathsf{ref}^M_{k,n}(x, y, z, u, v)$$

where x, y, z, u, v are tuples of variables as above to be the DNF formula

$$\mathsf{reject}^M_{k,n}(y, x, z) \to \neg \mathsf{sat}_n(u, x, v) \ .$$

Note that the size of $\mathsf{ref}^M_{k,n}$ is $k^{O(1)}$.

Definition 1.1 *For a proof system P let $\mathsf{Alg}(P)$ be the class of all algorithms M such that there are formulas $\mathsf{reject}^M_{k,n}$ satisfying the equivalence (F) and such that the corresponding formulas $\mathsf{ref}^M_{k,n}$ have polynomial size P-proofs.*

Let us consider an example. A Frege proof system is any propositional calculus operating with formulas in a complete basis, based on a finite number of axiom schemes and schematic inference rules which are sound and implicationally complete. By a theorem of Reckhow [11, 14] Frege systems polynomially simulate each other, even when having different languages. In particular, the minimal proof lengths of formulas differ at most polynomially in various Frege systems.

Let F be a Frege system in DeMorgan language with disjunction $\bigvee$ and conjunction $\bigwedge$ of unbounded arity. The depth $dp(A)$ of a formula is defined inductively: the depth of constants and variables is 0, the negation increases the depth by 1, and applying any of $\bigvee$, $\bigwedge$ to formulas A_i yields a formula of the depth $1 + \max_i dp(A_i)$. Note that in this counting of the depth DNF formulas have depth up to 3.

For $d \geq 3$ define F_d to be the subsystem of F using only formulas with the depth bounded by d. Systems F_d are collectively called by some authors AC^0 Frege systems. Note that various choices of F determine AC^0 Frege systems that p-simulate each other, although the particular depth may change in the simulation.

We claim that the class $\mathsf{Alg}(F_d)$ contains various common enhancements of DPLL, even for small d. To construct short F_d-proofs of the reflection principles (and, in fact, any other short proofs) it is best to use the general relation between first-order theories and proof systems. The relation has many facets but the one useful here can be informally described as follows. We say that proof system P **simulates** theory T if whenever T proves that a *co*NP-property $A(x)$ holds for all x then P admits polynomial size proofs of the propositional tautologies $\langle A \rangle_n$ expressing (as in the proof of the NP-completeness of SAT) that $\forall x(|x| = n)A(x)$

holds (see [14, Chpt.9] for details of the translation). First such simulation of a theory by a proof system was defined by Cook [9].

The AC^0 Frege systems simulate in this sense a theory from [7] called $V_1^0(BD)$ there, or V_1^0 in [14] or V^0 in [10]. The theory represents formulas, computations, assignments, etc. as binary string and allows to prove their properties by induction on the length of a string for any AC^0-property. It is easy to see that that suffices to prove the soundness of resolution or its various extensions corresponding to enhancements of DPLL. Namely, think that a k-tuple of clauses over variables $x_1, \dots, x_n$ is represented by a string defining a $(2n) \times k$ $0-1$ matrix where rows correspond to all $2n$ potential literals and 0 or 1 in row i and column j means that the i-th literal occurs in the j-th clause.

It is then straightforward to define, quantifying only over the matrix entries (i.e. its coordinates), that a tuple of clauses forms a refutation of another tuple of clauses. Having such a refutation, and an assignment to variables satisfying all initial clauses, prove by induction on j that the first j clauses in the refutation all are satisfied by the assignment (this is, in particular, an AC^0 property). This type of proof in V_1^0 uses induction for formulas with two quantifiers and can be simulated by short proofs in F_d for small d (in our definition of the depth $d = 5$ will suffice).

Remark: It may be quite cumbersome to transform a rejecting run of some algorithm, a DPLL enhancement, into a resolution refutation, as the references [3, 4, 6] mentioned earlier show. However, because we have a free hand in choosing the formulas $\mathsf{reject}_{k,n}^M$ we can incorporate such a refutation (and the transformation of the particular computation into it as well) into the non-deterministic witness (i.e. into the tuples of variables z in the formulas $\mathsf{reject}_{k,n}^M$). Then V_1^0 may prove such a reflection principle (and hence some F_d may prove shortly their propositional translations) even though it may not be able to formalize the particular transformation of computations into refutations or prove its properties.

The reader can find details of the simulation of V_1^0 by the AC^0 Frege systems (going really back to [18]) in [14].

2 Hard examples for SAT algorithms

To state the theorem without excessive technical assumptions let us call a proof system P **decent** iff the following tasks can be performed by polynomial time algorithms:

D1 From a P-proof π of formula $\psi(x)$ and a truth assignment a to variables x construct a P-proof of $\psi(a)$.

D2 Given a true sentence ψ (i.e. no variables) construct its P-proof.

D3 Given P-proofs π_1 of ψ and π_2 of $\psi \to \eta$ construct a proof of η.

D4 Given a formula $\varphi(u_1, \ldots, u_n)$ and a P-proof of $\neg\mathsf{sat}_n(u, \neg\varphi, v)$ construct a P-proof of φ.

Conditions D1-3 are fairly obvious to verify for the usual proof systems[2], including Frege systems and their constant depth subsystems F_d. The algorithm for condition D4 is defined by induction on the number of connectives in φ, cf.[14, Chpt.9]. Note that the decency of a proof system has not much to do with its strength.

Theorem 2.1 *Let P be a decent proof system and M a SAT algorithm. Assume that P admits polynomial size proof of all formulas*

$$\mathsf{ref}^M_{k,n}$$

corresponding to some formulas $\mathsf{reject}^M_{k,n}$ obeying the equivalence (F). Assume further that for some sequence of DNF tautologies τ_n of size $n \leq |\tau_n| \leq n^{O(1)}$ and the function

$$\ell(n) \ := \ \text{the minimal size of a } P\text{-proof of } \tau_n$$

it holds that $\ell(n)$ is super-polynomial (i.e. the formulas τ_n witness a super-polynomial lower bound for P).

Then M needs time $\ell(n)^{\Omega(1)}$ to reject $\neg\tau_n$ as unsatisfiable.

Proof :

For a tautology τ let k_τ be the time M needs to reject $\neg\tau$. The statement follows from

Claim: *Any tautology τ has a P-proof of size $k_\tau^{O(1)}$.*

Put $\varphi := \neg\tau$ and let $k := k_\tau$. Assume that the variables of φ are $u = (u_1, \ldots, u_n)$. Let $w \in \{0,1\}^k$ and $e \in \{0,1\}^s$ be such strings that the sentence $\mathsf{reject}^M_{k,n}(w, \varphi, e)$ is true.

By the existence of polynomial size P-proofs of $\mathsf{ref}^M_{k,n}$ and by the decency condition D1 we have that formulas

$$\mathsf{reject}^M_{k,n}(w, \varphi, e) \rightarrow \neg\mathsf{sat}_n(u, \varphi)$$

have p-size P-proofs. By D2 also sentences

$$\mathsf{reject}^M_{k,n}(w, \varphi, e)$$

have p-size P-proofs and thus by D3 the formulas

$$\neg\mathsf{sat}_n(u, \varphi)$$

have size $k^{O(1)}$ P-proofs. Applying D4 yields size $k^{O(1)}$ P-proof of τ.

Note the the final P-proof of τ is constructed by a p-time algorithm once given w and e.

[2]P. Beame has pointed out that not all standard SAT algorithms when considered as proof systems are decent. In particular, [4] have shown that resolution with clause learning does not satisfy D1 unless it simulates general resolution (which appears unlikely).

q.e.d.

Remark: The assumption that the hard tautologies τ_n are DNF is not essential. If τ_n is not a DNF one uses in the argument a CNF representing $\neg\tau_n$ via limited extension mentioned earlier. In particular, the decency condition D4 holds for the usual proof systems for these more complex formulas too.

There is quite a variety a proof systems for which even exponential lengths-of-proofs lower bounds (for DNFs) are known. These include the systems F_d introduced earlier, but also proof system operating with linear inequalities (e.g. cutting planes) or with polynomials over fields (e.g. polynomial calculus) and their various combinations, or even with OBDDs. On the other hand super-polynomial lower bounds are not known for (depth unrestricted) Frege systems; see [15] or [8] for an overview and references (and [16] for the OBDD proof system).

In principle it may happen[3] that some algorithm M, when considered as a proof system, may be included in a proof system P while not being in $Alg(P)$. Indeed, Atserias and Bonet [2] have shown that resolution proof system R does not prove its own soundness, i.e. no SAT algorithm whose rejecting runs correspond to general resolution proofs will be in $Alg(R)$. However, it seems plausible that any such (natural) algorithm will be in $Alg(F_d)$ for some $d \geq 3$.

The theorem implies, in particular, that any super-polynomial lengths-of-proofs lower bound for a decent proof system (for any sequence of tautologies) yields the following form of consistency for P $\neq$ NP. This is because the formulas $\mathsf{ref}^M_{k,n}$ are the propositional translations of the statement (**M**), just written contrapositively.

Corollary 2.2 *Let T be a first-order theory and P a decent proof system that simulates T in the sense described earlier.*

If P is not polynomially bounded (i.e. there is a sequence of tautologies requiring super-polynomial size P-proofs) then the set of the negations of all statements (M) for all clocked polynomial time algorithms M is consistent with T.

In particular, P $\neq$ NP is in this sense consistent with V^0_1 (in fact, even with a bit stronger theory with various combinatorial principles - like pigeonhole principle - added as axioms). Note that any super-polynomial lower bounds for Extended Resolution (which is equivalent to Extended Frege system, cf. [11]) would imply such a consistency with theory V^1_1 whose principal axiom is induction on the length of strings for any NP-property. This theory defines all polynomial time algorithms and proves many of their properties (and, in fact, proves many significant complexity-theoretic results) and it is difficult to imagine a *natural* SAT algorithm whose soundness would not be provable there, cf.[7, 10, 14].

[3]As K. Ghasemloo pointed out.

Let us note that using some more proof complexity ([9, 17, 14]) one can get a similar statement about the consistency of NP $\neq$ *co*NP.

3 Finding hard satisfiable formulas

Unless P = NP, for any SAT algorithm M there have to be satisfiable hard formulas, formulas that need super-polynomial time: if M would always find a satisfying assignment for any satisfiable formula in time n^c we could equip M with a clock that would stop any computation going over the time and reject the input. This new algorithm M^c would be sound and would run in polynomial time.

In the next statement we observe that the reflection formulas $\mathsf{ref}^{M^c}_{k,n}$ can be used to find such hard instances efficiently.

Theorem 3.1 *Assume $P \neq NP$ and let M be any SAT algorithm. Let $c \geq 1$ be arbitrary.*

Then there is an algorithm that for infinitely many $n \geq 1$ constructs in time $n^{O(c^2)}$ from $1^{(n)}$ a satisfiable formula α_n of size at least n such that M needs more time on α_n than $|\alpha_n|^c$.

Proof :

Fix M and $c \geq 1$. From P $\neq$ NP it follows that for infinitely many $n \geq 1$ M needs more time than n^c on some satisfiable formula of size n.

Take any such $n \geq 1$ and put $k := n^c$; hence $\neg\mathsf{ref}^{M^c}_{k,n}$ is satisfiable. Note that the size of $\neg\mathsf{ref}^{M^c}_{k,n}$ is $n^{O(c)}$.

We let M run on $\neg\mathsf{ref}^{M^c}_{k,n}$ for time $|\neg\mathsf{ref}^{M^c}_{k,n}|^c \leq n^{O(c^2)}$. If it finds a satisfying assignment we read α_n of size n from it. Otherwise we take for α_n the formula $\neg\mathsf{ref}^{M^c}_{k,n}$ itself.

q.e.d.

Note, as Albert Atserias has pointed out, that we have apparently no way to tell which length n is good (i.e. the algorithm succeeds in finding the hard formula) unless the first case happens for the particular n. This drawback can be removed by replacing the assumption P $\neq$ NP by a presumably stronger one that every polynomial time algorithm fails to compute SAT on any sufficiently large length n. Then our algorithm succeeds to find hard satisfiable instances of every sufficiently large length (of a specific form and other lengths can be treated by a suitable padding). Bogdanov et.al.[5] manage to construct hard witnessed satisfiable instances (for infinitely many n, assuming also P $\neq$ NP).

In fact, it would be interesting to remove the hypothesis P $\neq$ NP from the theorem altogether for some broad class of SAT algorithms. Alekhnovich et.al.[1] do this for two subclasses of DPLL algorithms, the so called generalized myopic and drunk algorithms, but their algorithm finding hard satisfiable instances succeeds only with a high probability.

Acknowledgements: I thank Paul Beame, Luke Friedman, Kaveh Ghasemloo, Leszek Kolodziejczyk and Neil Thapen for comments. I thank in particular to Albert Atserias for his remarks on Theorem 3.1 and for pointing out [5].

References

[1] M. Alekhnovich, E. A. Hirsch, and D. Itsykson, Exponential lower bounds for the running time of DPLL algorithms on satisfiable formulas, in: Proc. of ICALP 2004, LN Computer Science, **3142**, (2004), pp.84-96.

[2] A. Atserias and M. L. Bonet, On the Automatizability of Resolution and Related Propositional Proof Systems, *Information and Computation*, **189(2)**, (2004), pp.182-201.

[3] A. Atserias, J. K. Fichte, and M. Thurley, Clause-learning algorithms with many restarts and bounded-width resolution, *J. of Artificial Intelligence*, **40**, (2011), pp.353-373.

[4] P. Beame, H. Kautz, and A. Sabharwal, Towards understanding and harnessing the potential of clause learning, *J.of Artificial Intelligence Research*, **22**, (2004), pp.319-351.

[5] A. Bogdanov, K. Talwar, and A. Wan: Hard instances for satisfiability and quasi-one-way functions, in: *Proc. of the First Symposium on Innovations in Computer Science*, (2010), pp.290-300.

[6] M. L. Bonet and S. R. Buss, An Improved Separation of Regular Resolution from Pool Resolution and Clause Learning. Preliminary manuscript, 2011.

[7] S. R. Buss, *Bounded Arithmetic.* Naples, Bibliopolis, (1986).

[8] S. R. Buss, Towards NP-P via Proof Complexity and Search. To appear, 2011.

[9] S. A.Cook, Feasibly constructive proofs and the propositional calculus, in: *Proc. 7th Annual ACM Symp. on Theory of Computing*, (1975), pp. 83-97. ACM Press.

[10] S. A. Cook, and P. Nguyen, *Logical foundations of proof complexity*, Cambridge University Press, 2010).

[11] S. A. Cook, and Reckhow, The relative efficiency of propositional proof systems, *J. Symbolic Logic*,**44(1)**, (1979), pp.36-50.

[12] M.Davis, G. Logemann, and D. Loveland, A Machine Program for Theorem Proving, *Communications of the ACM*, **5(7)**, (1962), pp.394397.

[13] M.Davis and H.Putnam, A Computing Procedure for Quantification Theory, *J.of the ACM.* **7(3)**, (1960), pp.201215.

[14] J. Krajíček, *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications, Vol. **60**, Cambridge University Press, (1995).

[15] J. Krajíček, Proof complexity, in: Laptev, A. (ed.), European congress of mathematics (ECM), Stockholm, Sweden, June 27–July 2, 2004. Zurich: European Mathematical Society, (2005), pp.221-231

[16] J. Krajíček, An exponential lower bound for a constraint propagation proof system based on ordered binary decision diagrams, *J. of Symbolic Logic*, **73(1)**, (2008), pp. 227-237.

[17] J. Krajíček and P. Pudlák, Propositional Proof Systems, the Consistency of First Order Theories and the Complexity of Computations, *J. Symbolic Logic*, **54(3)**, (1989), pp. 1063-1079.

[18] J. Paris and A. J. Wilkie, Counting problems in bounded arithmetic, in: *Methods in Mathematical Logic*, Ed. C.A.DiPrisco, LNM 1130, (1985), pp.317-340. Springer.

[19] P. Pudlák, The lengths of proofs, in: Handbook of Proof Theory, S.R. Buss ed., Elsevier, (1998), pp.547-637.

Mailing address:
Department of Algebra
Faculty of Mathematics and Physics
Charles University
Sokolovská 83, Prague 8, CZ - 186 75
The Czech Republic
`krajicek@karlin.mff.cuni.cz`