Propositional Proof Complexity

Alexander Razborov
Institute for Advanced Study, Princeton
and
Steklov Mathematical Institute, Moscow

The framework underlying propositional proof complexity was developed in the seminal paper [Cook and Reckhow 1979].

Definition 1. Let TAUT be the set of all propositional tautologies. A propositional proof system is any poly-time computable function

\[ P : \{0, 1\}^* \rightarrow \text{TAUT}. \]

The proof complexity \( S_P(\phi) \) of a tautology \( \phi \in \text{TAUT} \) is naturally defined as

\[ S_P(\phi) \overset{\text{def}}{=} \min_{P[w] = \phi} |w|. \]

\( P \) is \( p \)-bounded if \( S_P(\phi) \) is polynomial in \( |\phi| \).

The basic task of propositional proof complexity is to learn as much as possible about the behaviour of \( S_P(\phi) \) for "interesting" proof systems \( P \) and "interesting" tautologies \( \phi \). In particular, central problems in this area ask whether certain concrete and natural proof systems are \( p \)-bounded.

The main question of the classical (non-uniform) computational complexity is which objects (like functions or languages) exist in a world where our computational abilities are limited. Likewise, for almost all proof systems of significance the basic task formulated above can be seen as studying what is provable in such worlds. In this sense proof complexity makes a nice and important complement to computational complexity. Besides this, it is also tightly connected in many ways to classical proof theory, automated theorem proving and cryptography. Unfortunately, due to lack of space we can not elaborate here on these connections. The interested reader may consult for a general overview of the area the sources [Urquhart 1995; Krajíček 1995; Razborov 1996; Beame and Pitassi 1998; Pudlák 1998] (serving various tastes).

Two most prominent proof systems that are in the focus of attention today are Frege and Extended Frege. A Frege proof system is just an ordinary textbook propositional calculus (having finitely many axiom schemes and inference rules); this definition is very robust in the sense that all possible versions of this system are polynomially equivalent in terms of the function \( S_P(\phi) \). Extended Frege extends Frege by allowing (possibly iterated) abbreviations of the form \( p_A \equiv A \), where \( A \) is a propositional formula and \( p_A \) is a new propositional variable (this definition is also robust). In the sense discussed in the previous paragraph, Frege is the proof system for the world of \( \text{NC}^1 \)-computability, and Extended Frege corresponds to poly-time computability. Whence stems their extreme importance.

A less pleasant thing stemming from the same source, however, is the wide (although not universal) belief that the question whether Frege or Extended Frege is \( p \)-bounded must be even harder to answer than \( \text{NC}^1 \neq \text{P} \) and \( \text{P} \neq \text{NP} \), respectively. Indeed, the existing evidence strongly suggests that proof complexity
lower bounds are even harder to attain than computational lower bounds for the respective complexity classes.

Quite fortunately, there is a weaker version of this question which is almost as interesting but at the same time looks by far more feasible. For these reasons it is this version that is formulated here as the main (in the author’s opinion, of course) problem of propositional proof complexity. Show that proof complexity lower bounds are at most as hard as computational lower bounds or, more specifically,

\[
\text{Prove that Frege or Extended Frege are not } p\text{-bounded modulo any reasonable hardness assumption in the purely computational world.}
\]

\textbf{Remark 1.} Reasonable here is roughly understood as “arbitrarily strong but still natural, acceptable and believable”. The most prominent candidate (at the moment) is the existence of pseudo-random generators (or one-way functions). On the other hand, } \text{NP} \neq \text{co-NP} \text{ is equivalent to the much more general fact of non-existence of any } p\text{-bounded proof system whatsoever. Therefore, purely computational refers to the demand that the assumption itself should speak only about computations and should not attempt to restrict the power of proofs even in a disguised form.}

\textbf{Remark 2.} The introduction to [Razborov 2002] contains a more or less complete and updated (up today) account of the author’s views on importance and feasibility of this conjecture, various approaches to it, as well as of its connections to the provability of central open problems in computational complexity itself.

I would like to use this opportunity and briefly emphasize another interesting direction in proof complexity, even if it is apparently lacking well-defined open problems comparable in importance with the one above. Namely, in more practical activities (like automated theorem or program verification) the question of existence of efficient proofs is only half the story. What is at least as important is how to search efficiently for such proofs.

As above, the “absolute” versions of this question which simply ask for as good performance of any proof search algorithm as possible have (in the author’s opinion) little to do with proof complexity. A proof system with polynomial search time exists if and only if } \text{P} \neq \text{NP}. A proof system optimal in this sense exists if and only if there exists an optimal deterministic algorithm for } \text{NP}\text{-complete problems etc. Be these problems as important as they obviously are, they do not fall into the scope of our brief article.

What does belong to its scope is the relative question of whether search for efficient proofs substantially adds to the inherent complexity of the statement in a specific proof system. Mathematically this is captured by the notion of automizability [Bonet et al. 2000]: a proof system } \text{P} \text{ is called automatizable if there exists a deterministic algorithm that outputs a } P\text{-proof of any tautology } \phi \text{ in time which is polynomial in } S_P(\phi) \text{ (i.e., in the size of the optimal } P\text{-proof ).}

By their nature, automatizable proof systems make a perfect base for automated theorem provers of any sort. Unfortunately, they in general seem to be few and rare
(and for many existing proof systems it is known that they are not automatizable modulo various hardness assumptions). It would be very interesting to study this issue deeper, both in terms of classifying existing proof systems (with respect to their automatizability) as well as in terms of building automatizable proof systems with prescribed properties. For example\(^1\), does there exist a proof system \( P \) for refuting CNFs such that:

- if a CNF \( \psi \) is obtained from a CNF \( \phi \) by renaming variables or clauses or by adding new clauses then \( S_P(\psi) \leq S_P(\phi)^{0(1)} \);
- \( P \) is automatizable;
- \( P \) has a polynomial size proof of the pigeonhole principle?

Its existence would capture some known computational algorithms and would most likely lead to other interesting algorithmic consequences.

REFERENCES


\(^{1}\)this problem was contributed by R. Impagliazzo