SIGACT News Complexity Theory Column 91

Lane A. Hemaspaandra
Dept. of Computer Science
University of Rochester
Rochester, NY 14627, USA

Introduction to Complexity Theory Column 91

This issue's column is a wonderful (and perspective-expanding) article by Alexander Razborov, on Proof Complexity and Beyond. Thank you so very much, Sasha!

And please stay tuned also for the Complexity Theory Columns in the coming issues, which will feature articles by Swastik Kopparty and Shubhangi Saraf (tentative title: "Local Testing and Decoding of High Rate Error Correcting Codes") and Neeraj Kayal and Chandan Saha (tentative topic: arithmetic circuit lower bounds).

David Johnson's book, coauthored with Michael Garey, *Computers and Intractability*, is a true treasure that has made NP-completeness and the great variety of NP-complete problems known to and clear to decades of enraptured readers. David of course also was endlessly generous in service, a stellar researcher, and a passionate advocate for theoretical computer science. He has passed away at the age of 70, and will be deeply missed. (Please see the in-memoriam in this issue.)

Guest Column: Proof Complexity and Beyond[1]

Alexander Razborov[2]

Abstract

This essay is a highly personal and biased account of some main concepts and several research directions in the modern propositional proof complexity. A special attention will be paid to connections with other disciplines.

1 Introduction

Everyone loves (or at least ought to love) proving theorems. If this statement appears exaggerated, let me remark that the notion of a "theorem" is perhaps way broader than it is commonly thought of. It is undoubtedly true that the the first connotation this word invokes goes back to Euclid's "Elements", a rigorous and convincing argument deducing an abstract mathematical statement from agreed-upon axioms. But if you run your favorite SAT solver on a CNF τ, and it determines that τ is unsatisfiable, it is also a theorem, the log of this run being its proof. The fact that a piece of hardware or software meets its specifications is a theorem as well, and, with a bit of oversimplification, one could say that the whole industry of soft/hardware verification is devoted to proving (well, more often disproving in fact) theorems of this sort. When we apply an approximation algorithm to a combinatorial optimization problem, and it tells us there is no solution within the integrality gap, this is also a theorem. Its proof in this context is called an analysis of the algorithm.

Proof complexity is the discipline that studies *efficient*, often called *feasible*, provability in formal proof systems of all kinds of statements, without discriminating them on the base of their origin. The only requirement is that the statements we are interested in can be written in the language of the system. The emphasis on *efficiency*, as opposed to mere *existence* is what makes proof complexity different from its parent discipline, classical Proof Theory.

Proof systems that are studied in the area can be roughly sub-divided into two large groups, according to how they treat quantifiers or, sometimes, even more sophisticated logical tools. One possible approach is to keep them while introducing complexity elements by severely restricting the power of axioms and bounding the scope of quantifiers. This approach is collectively known as *Bounded Arithmetic*, it was introduced in Sam Buss's thesis [19]. It will not be considered in our essay; besides [19], we refer the interested reader to [32, 21].

[1]© A. Razborov, 2016.

[2]University of Chicago, `razborov@cs.uchicago.edu`, Steklov Mathematical Institute, Moscow, Russia, and Toyota Technological Institute, Chicago.

Another approach, and this is the one that we adopt here, is quite radical: ban quantifiers whatsoever and replace first-order logic with something simpler or different, like propositional calculus. This leads to the sub-field of *propositional proof complexity* which is the main hero of our story. But before saying our final farewell to quantifiers, let me remark that there exist extremely tight connections between propositional proof systems and theories of Bounded Arithmetic, closely mirroring connections between circuit and Turing complexities; see e.g. [32, 35].

For technical reasons it is almost always more convenient to negate the statement, and, instead of proofs, consider *refutations*, or proofs of unsatisfiability of specific unsatisfiable instances of the constraint satisfaction problem, aka *contradictions*. The latter can be often thought of as a Boolean CNF expression. All proof systems considered will be sound and complete, which is one more important deviation from the classical proof theory. Propositional proof complexity is oriented toward understanding various complexity measures of proofs, much in the same way it happens in the circuit, arithmetic, communication and any other non-uniform complexity theory.

Disclaimer. This essay is *not* a full-fledged survey in proof complexity, and I have tried to suppress the temptation of turning it into a surrogate and lousy one. The choice of topics is highly biased towards my personal tastes, and even on those included, the list of results is exemplary rather than comprehensive. My main intention is to highlight some general points and ideas of modern proof complexity, sometimes with considerable digressions.

I will try to compensate for all these deficiencies by amply suggesting further readings on the subject in appropriate places, as I already started above.

2 Theorems (and conjectures) of interest

In this section I will elaborate on the list of sources of interesting tautologies we briefly mentioned in Introduction. One thing I do not touch here are propositional tautologies coming from industrial and engineering applications. A very good sample of these can be found e.g. in the application track of the SAT solving competition `http://www.satcompetition.org/`.

I will try my best to concentrate on tautologies themselves deferring the question of their (efficient) provability in *concrete* proof systems to Section 5. But sometimes it is unavoidable to use the words like "Frege", "Extended Frege" or even "semi-algebraic proof systems" in surrounding discussions, and the reader unfamiliar with these terms should consult Section 3 as needed.

2.1 Random k-CNFs

Perhaps, one of the most shocking discoveries awaiting everyone who, like myself, drifted to propositional proof complexity from circuit complexity or a similar endeavor is that

Counting arguments do not count any more.

The "Shannon-Lupanov effect" (see e.g. [30, Chapter 1.4.1] or [7, Chapter 6.5]) in circuit complexity tells us that almost all Boolean functions are complex, and in various forms, this idea propagates into almost any adjacent area. For example, in algebraic complexity it turns into a simple dimension argument showing that almost all (in Zariski's sense) polynomials are hard etc. As we know, this leads to a peculiar notion of an "explicit" problem.

Not so in propositional proof complexity. Here the space of all problems, that is contradictions of a given size, is roughly of the same magnitude as the space of all solutions, that is proofs. Thus, at least a priori it is not clear why, given a proof system, showing lower bounds for an explicit tautology should be any more difficult than a non-constructive result merely stating that a hard tautology exists. A posteriori, this becomes even more resonating: to the best of my knowledge, we currently do not know of a *single* natural proof system for which an existential proof that something is hard for it would not be witnessed by an example of a hard tautology explicit in the good common sense. This empirical observation might be taken as an interesting counterpart to the theory of Natural Proofs in circuit complexity [53], except that here we remove all disclaimers related to counting... well, because it does not count any more.

Another interesting side of (arguably) the same phenomenon is that for really strong proof systems it is awfully difficult to come up even with *candidate* tautologies to be hard – there is no such thing as tons of NP-complete problems at our disposal! The only effort to systematically address this I am aware of goes back to the paper [40]. However, echoing the sentiment expressed by the authors themselves, I would not call candidates from that list "robust" in any reasonable sense. Indeed, many of them have been since that broken in ways that I would call "quite expected". It would be perhaps fair to say that there are only two classes of explicit candidate tautologies for which there are sustainable reasons to believe in their hardness (let me stress that we are now talking only about very strong proof systems like Frege or Extended Frege). One will be addressed in Section 2.3, and another would be propositional translations of a feasible version of Göedel's incompleteness theorem, see e.g. [19, Chapter 7].

But if we do not insist on explicitness, this is where random k-CNFs come into the picture. While they fail to do the job of providing hard instances *provably*, they still do it *probably.* The belief that no proof system whatsoever can refute, say, random 3-CNFs a.e. is widely spread in the proof complexity community, and outside of it this conjecture is commonly referred to as *Feige's Hypothesis* [25]. This unique position taken by random CNFs, even if it is mostly due to the absence of other good choices, make them one of the most favorite benchmarks both in propositional proof complexity and in the practical SAT solving where they make their own track `http://www.satcompetition.org/`.

Let me know discuss the choice of parameters. It turns out that in this department proof complexity is also pretty robust, and in many cases all reasonable choices of a random model will bring the same results.

Definition 1 (Random k-CNFs) Let $k, \Delta > 0$ be fixed constants. The random k-CNF $\boldsymbol{\tau_k^{n,\Delta}}$ is defined by choosing uniformly at random $m = \Delta n$ clauses of width k in n variables.

When $\Delta = \Delta(k)$ increases, at some point called the *satisfiability threshold*, $\boldsymbol{\tau_k^{n,\Delta}}$ turns from a.e. satisfiable to a.e. unsatisfiable; the recent breakthrough result [23] determines the exact value of this point for sufficiently large k. But none of these subtleties is relevant for propositional proof complexity. Most known lower bounds work already for $k = 3$, and even when the original techniques require larger values of k, they are often eventually improved to $k = 3$. None of the concrete results, to the best of my knowledge, depends on the choice of the parameter Δ, and most of them extend well into the super-constant region. In particular, passing the phase transition point Δ_k goes absolutely unnoticed by typical methods in proof complexity – differentiating whether efficient refutations of a formula do not exist due to a deep argument or simply because it is satisfiable (and you can not prove false facts) is about the last thing they are likely to try. This is yet another reminder (we will see more below) that while the landscape of proof complexity looks mostly familiar to computational complexity folks, now and then we spot some oddities that remind us: we are not completely at home, and relations between bare truth and its provability are more subtle than they may appear at the first sight.

2.2 Combinatorial principles

But even given these differences, proof complexity is still a full-fledged member of, as it were, Complexity Commonwealth and, in order to conform to its spirit and standards, it must have its own stock of totally useless problems that it studies anyway. I do not think I should defend this thesis to a typical reader of this column, nor should I go into lengthy explanations as to why we conduct our studies this way. But just in case these pages are visited by someone out of the guild, let me point out that quite similar phenomenon exists in many more practically oriented communities, where it is called "benchmarks". In particular, the third (and the last) track of the SAT solving competition we already referred to is precisely like that: combinatorial.

Proof complexity has borrowed from combinatorics, algebra etc. many valid statements that are simple enough to fit into our framework that is quantifier-free, and developed quite a few of its own. All of them are "obviously true", this is why I allowed myself the epithet "useless" above, and the question is what is the exact power of a proof system that can prove them efficiently. Here I will confine myself only to two principles and their derivatives that, along with random k-CNFs, are probably the most frequently used benchmarks in the field.

Definition 2 (Pigeonhole Principle) It is exactly what is says. Let $m > n$, and let us introduce mn propositional variables x_{ij} $(i \in [m], j \in [n])$. In the form of a contradiction, PHP_n^m consists of the following axioms (clauses):

$$Q_i \stackrel{\mathrm{def}}{=} \bigvee_{j=1}^{n} x_{ij} \ (i \in [m]);$$

$$Q_{i_1,i_2;j} \stackrel{\mathrm{def}}{=} (\bar{x}_{i_1 j} \vee \bar{x}_{i_2 j}) \ (i_1 \neq i_2 \in [m],\ j \in [n]).$$

In this formulation, pigeons i and holes j are asymmetric. To circumvent this, we may add *functionality axioms*

$$Q_{i;j_1,j_2} \stackrel{\mathrm{def}}{=} (\bar{x}_{ij_1} \vee \bar{x}_{ij_2}) \ (i \in [m],\ j_1 \neq j_2 \in [n])$$

saying that every pigeon flies to *exactly* one hole and/or *surjectivity, or onto axioms*

$$Q_j \stackrel{\mathrm{def}}{=} \bigvee_{i=1}^{m} x_{ij} \ (j \in [n])$$

saying that every hole is occupied. This gives us four different versions of the pigeon-hole principle, besides the fact that the number of pigeons m can be also an arbitrary function of n.

Either version contains very wide clauses that is rather inconvenient and unnatural in many situations; perhaps, it is a good place to mention that proof complexity strongly favors statements expressible as CSPs in which all constraints are of bounded arity, and preferably (cf. the previous section 2.1) of arity 3. It is quite obvious how to circumvent this for PHP_n^m: we simply set most of the variables to zero, keeping alive only Δn of them, but as far as I remember, to a good use this simple idea was for the first time put in the seminal paper [14] by Ben-Sasson and Wigderson. Let $G - \mathrm{PHP}_n^m$ be the resulting principle, where $G \subseteq [m] \times [n]$ is the bipartite graph representing the set of remaining variables. This reduction gives the best results when G is a constant-rate expander or boundary expander, and it is one of the most commonly recurring themes in proof complexity that expansion implies complexity, see e.g. [12].

Upon a moment's reflection, the functional onto version $onto - G - \mathrm{FPHP}^m_n$ of the latter principle simply says that the graph G does not contain a perfect matching, and we could equally well to say this explicitly and more generally. Namely, for any simple graph $G = (V, E)$, not necessarily bipartite, the *perfect matching principle* $\mathrm{PM}(G)$ expresses that G does not[3] possess a perfect matching. That is, we introduce propositional variables x_e for all edges $e \in E$ and consider the axioms

$$\bigvee_{e \ni v} x_e \ (v \in [V]);$$
$$\bar{x}_e \wedge \bar{x}_f \ (e, f \in E \text{ share a vertex}).$$

Strictly speaking, $\mathrm{PM}(G)$ is not a "principle" since it does not pass the "identically" test – it is true for some graphs G and false for others. Thus, in full generality it rather belongs to the class of tautologies that will be discussed in Section 2.4 below. But we have Tutte's theorem, and while the pigeonhole principle captures one reason why G may not have a perfect matching, the complementary one is captured simply by saying that the number of vertices is odd.

Let us in particular look at the principle $\mathrm{PM}(K_n)$ for an odd n. After a moment's thinking, we see that it is basically the *counting principle* Count_2 stating that an universe V of odd size can not be partitioned into sets of size 2. It can be obviously generalized to an arbitrary integer r in place of 2 (or, if you prefer, to perfect matching principles in r-graphs).

Definition 3 (Tseitin tautologies) Let again $G = (V, E)$ be a simple undirected graph, and let $\sigma : V \longrightarrow \{0, 1\}$ satisfy $\bigoplus_{v \in V} \sigma(v) = 1$. The Tseitin tautology $T(G, \sigma)$ in the same variables x_e $(e \in E)$ has the set of axioms $\bigoplus_{e \ni v} x_e = \sigma_v$.

Tseitin tautologies are very useful because of the structure and rich symmetries provided by the XOR operator. Their obvious drawback is that they are not CNFs. In most cases this is circumvented by requiring that the maximum degree $\Delta(G)$ of G is bounded, which allows us to re-write the Tseitin tautology $T(G, \sigma)$, if desired, as a CNF with size increase by at most a factor of $2^{\Delta(G)}$. Also, the conclusions about its hardness typically follow from the expansion properties of G.

When $\sigma \equiv 1$ and hence n is odd, $T(G, \sigma)$ is a stronger (meaning it has less restrictive axioms) statement than $PM(G)$. Thus, lower bounds for $PM(G)$ imply lower bounds for Tseitin tautologies, although I am not aware of any useful applications of this reduction. On the other hand, $T(G, \sigma)$ is just an unsatisfiable system of $\mathbb{F}_2$-linear equations that happens to have a special form, and it is often useful to get rid of their structure and consider random k-XOR formulas, defined analogously to Definition 1. These formulas are weaker than their Boolean counterparts $\tau^{\mathbf{\Delta}}_{\mathbf{k}}$, and the corresponding reduction *has* turned quite useful: many lower bounds for random k-CNFs are actually obtained by relaxing them first to random k-XOR formulas.

It can be only added that Tseitin tautologies can be easily generalized to fields of arbitrary characteristics, including zero, and in the latter case they naturally become *flow tautologies.* For that we need to consider oriented graphs; we refer to [18, 5] for details.

[3]Yes, in proof complexity we tend to blur the difference between "identically true" and "identically false", it is the adverb "identical" that matters!

2.3 $\mathsf{NP} \overset{?}{\not\subseteq} \mathsf{P}/poly$ and Proof Complexity

The set of ideas reviewed in this section was developed in connection and in analogy with the theory of Natural Proofs [53], and some familiarity with the latter would be helpful for understanding the message. Our goal is the same as in Natural Proofs: capture a decent class of arguments that is powerful enough to carry all "normal" lower bound *proofs* (as opposed to approaches) in non-uniform circuit complexity for *explicit* functions and try to say something intelligent about proving lower bounds for general circuits by arguments from this class. The hope is also similar: to help focusing research in "promising" directions and, optimistically, shed at least some light on the nature of difficulties mysteriously surrounding our burning questions.

The first major difference from Natural Proofs is in the choice of the framework. Usefulness, Largeness and Constructivity conditions in Natural Proofs are arguably very natural things (I personally love them), but, like it or not, the fact is that: they were cooked up for the purpose. Does there exist a mainstream logical theory T that would have precisely the same separation property: existing lower bounds can be "straightforwardly" formalized in T, with an efficiency clause if T is propositional, while there are good reasons to believe it is not the case for lower bounds against general circuits?

As in Natural Proofs, the first major decision to be made is this: are we going to declare the input size to be n or 2^n, where n is the number of variables of the function in question? Attempts to go with the first option inevitably lead to quantifiers – we will need them even to express the value of a function in NP.

If, on the other hand, we agree to allow tautologies and proofs to have size polynomial in 2^n (which is exactly what happens in Natural Proofs), the translation works out very smoothly and leads to the 3-CNF formula $\mathrm{Circuit}_t(f_n)$ of size $2^{O(n)}$ whose unsatisfiability precisely means that the Boolean function f_n in n variables does not possess circuits of size $\leq t$. For an exact definition we refer the reader e.g. to [51, Chapter 8], but, given the luxury afforded by having $2^{O(n)}$ variables, the encoding is pretty straightforward. For every gate v in a prospective circuit and every $a \in \{0,1\}^n$ we introduce a special Boolean variable y_{av} etc. The same definition works for any restricted circuit class Λ, of course.

The thesis that all concrete lower bound proofs against a circuit class Λ are not only naturalizable but at the same time lead to $2^{O(n)}$-sized Extended Frege refutations of the corresponding formula $\Lambda - \mathrm{Circuit}_t(f_n)$ was explicitly put forward in [48, Appendix]. In a sense, it works even better than in Natural Proofs: there is no reason now to exclude lower bounds for monotone circuits. In fact, the method of approximations is about the only class of arguments that seems to require the full power of Extended Frege, most others seem to be contend with ordinary Frege or even below [48, Appendix E].

So, can we extend the main (and the only) theorem of Natural proofs [53, Theorem 4.1] to the current context and show that under some standard or, for that matter, any reasonable complexity/cryptographic assumption Extended Frege does not possess $2^{O(n)}$-sized refutations of, say $\mathrm{Circuit}_{n^2}(f_n)$ for a function f_n of our choice?

The second main difference is that for some reasons beyond my comprehension, in proof complexity this question, very akin to the one answered in Natural Proofs, turns into an extremely difficult open problem to which we currently do not seem to have even viable approaches. Per our discussion in Section 2.1, we can even strip off the particular context and ask if we can prove conditional lower bounds on (Extended) Frege proofs of an *arbitrary* contradiction, not necessarily $\mathrm{Circuit}_t(f_n)$ or even explicit. The result will still be the same: this currently appears to be completely out of reach of the current methods.

We will re-visit this issue in Section 5, and for now let us briefly review a relevant class of candidate tautologies introduced in [33] and [4]; for a comprehensive account see [51, Section 1] or [35, Chapters 29, 30]. On a conceptual level, recall that a poly-time computable mapping $G_n : \{0,1\}^n \longrightarrow \{0,1\}^m$ $(m > n)$ is a pseudo-random generator if no circuit of size $m^{O(1)}$ can distinguish between $\mathbf{y} \in_R \{0,1\}^m$ and $G_n(\mathbf{x})$, where $\mathbf{x} \in_R \{0,1\}^n$. Depending on the relation between m and n, it can be viewed as a pseudo-random number generator or a function generator. [33, 4] proposed to call G_n a pseudo-random generator that is *hard for a propositional proof system* P if P can not efficiently refute the fact $b \notin \mathrm{im}(G_n)$ for *any* string $b \in \{0,1\}^m$.

By the same argument as in [53, Theorem 4.1], if P has a hard pseudo-random generator with parameters $G_n : \{0,1\}^{n^k} \longrightarrow \{0,1\}^{2^n}$ then it can not efficiently refute $\mathrm{Circuit}_{n^C}(f_n)$ for any f_n, where n^C is the size of a circuit computing G_n. The hope in introducing this concept was precisely that by the token of being better structured, it may turn out to be useful for understanding efficient provability of the formulas $\mathrm{Circuit}_{n^C}(f_n)$ itself. I refer the reader to [4, 51] for an extended account of these principles and their relations to others, including those already discussed in this essay.

2.4 Tautologies from Combinatorial Optimization

This is actually not as much about concrete examples as about quite a general paradigm relating propositional proof complexity with algorithms in combinatorial optimization and particularly those based on LP/SDP relaxations. This connection was gradually worked out in a series of papers; as some of the landmarks I can mention [17, 2], and I also recommend the later survey [57] fully exploring it in one important scenario.

The underlying idea is simple, and I will illustrate it by an equally simple example. Let us take our favorite optimization problem, like VERTEX COVER, and encode its instance in the style of Section 2.2:

$$x_u \vee x_v \ ((u,v) \in E(G)). \tag{1}$$

Take also an approximation algorithm for VERTEX COVER with relative performance guarantee (say) 2, and run it on your favorite graph G; let k be the value it produces. Then the correctness of our algorithm implies that the Boolean formula obtained from (1) by adding the constraint

$$\sum_{v \in V(G)} x_v < k/2 \tag{2}$$

is unsatisfiable and, moreover, any proof system in which we can "argue" about the correctness should also be the one in which the combined system of constraints (1), (2) must have an efficient refutation. What gives this unsophisticated observation its enormous power is the fact that the proof systems obtained in this way are usually not something artificial, but actually were independently studied in proof complexity for a long time; we will review a few key systems in Section 3.2. Thus, this connection really allows to bring together two different communities working essentially on the same problems, even if sometimes from different perspectives.

Two proof-theoretical remarks about the constraint (2) are in place here. The first is actually a concern: as we repeatedly said before, in proof complexity we highly prefer simply looking contradictions, ideally expressible as constant-width CNFs. This concern is legitimate, but one good answer to it is this: proof systems *actually* resulting from existing approximation algorithms already allow linear inequalities in their language, so this concern is not a practical problem at the moment. But even if in the future[4] we would need to study efficient provability of contradictions including linear inequalities in a poorer language that forbids them, the remedy is well-known in proof complexity, and it is called *extension variables*. I will talk a bit about this, albeit in a slightly different context, in Section 3.1.

The second remark is actually a speculative question. What approximation algorithms based on LP/SDP relaxations really do in proof-complexity terms is *not* refuting the combined system (1), (2). Instead, they *deduce* the negation $\sum_{v\in V(G)} x_v \geq k/2$ of (2) from the axioms (1). Now, in proof systems that are strong enough to allow the deduction theorem, this difference is of course immaterial. But I do not see any obvious reasons (perhaps, I am overlooking something simple) why it should hold in relatively weak semi-algebraic proof systems that we will review in Section 3.2.

Let me perhaps re-phrase the same thing algorithmically. Assume that we have a (say) minimization problem $\mathcal{P}$ with the goal function g that we attempt to solve using a systematic LP/SDP hierarchy or perhaps even a stand-alone LP/SDP program. By the trivial binary search we can assume w.l.o.g. that we are solving a *decision* problem $g(x) \overset{?}{>} k$. What will happen if we plug into the solver both the original constraints *and* the constraint $g(x) \leq k$, expecting to get a contradiction if it is unsatisfiable? Can this pre-processing step help to improve performance of the original solver passively waiting for the estimate on $g(x)$ to come out? I do not see an immediate general answer to this question, but, again, I may be overlooking something simple.

As I said, this scheme works for an arbitrary combinatorial optimization problem in very much the same way, so I will skip further examples with an exception for the *Small-Set Expansion Hypothesis* [47] that is of extreme importance due to its relevance to the Unique Games Conjecture. We follow the excellent exposition in [9], although, of course, we prefer to view all things through the proof-complexity lens.

Definition 4 (SSEH tautologies in the weighted form) Let G be a symmetric stochastic $n \times n$ matrix, and $\delta, \epsilon > 0$ be constants. The formula $\mathrm{SSEH}_{G,\delta,\epsilon}$ has Boolean variables $x_1, \ldots, x_n$ and the axioms

$$\begin{aligned} \sum_{i=1}^n x_i &= \delta n \\ (\bar{x}_1, \ldots, \bar{x}_n)\, G \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} &\leq \epsilon\delta n. \end{aligned} \tag{3}$$

Thus, unsatisfiability of $\mathrm{SSEH}_{G,\delta,\epsilon}$ means that for every set S of δn vertices in the weighted regular graph G has edge-expansion at least ϵ, after an appropriate normalization.

Since G is stochastic, the second constraint (3) can be equivalently represented as $(x_1, \ldots, x_n)\, G \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \geq (1-\epsilon)\delta n$. Either way, it is *quadratic*; for the reasons we discussed earlier, we are not perplexed by this fact.

[4]I should remark that I am not aware of any coherent effort of a proof-complexity analysis of those approximation algorithms that are *not* based on LP/SDP relaxations, that would be very interesting to see how this works.

3 Proof systems and complexity measures

The most general definition of a proof system was given in the seminal paper by Cook and Reckhow [22]. Let TAUT be the set of propositional tautologies.

Definition 5 A *propositional proof system* P is any surjective poly-time computable function $P : \{0,1\}^* \longrightarrow \text{TAUT}$.

The idea here is simple: proofs are represented by binary strings w, and $P(w)$ does two things:

1. checks that w represents a legal proof in the intended proof system; if it does not, $P(w)$ outputs something like $x \vee \bar{x}$.

2. outputs the last line in this proof, i.e., its conclusion.

In its full generality, Cook-Reckhow's definition can be studied in the spirit of structural complexity. For example, the assumption that there exists an optimal, in the obvious way, propositional proof system is known to be implied by $\mathsf{EXP} = \mathsf{NEXP}$ [36], but the opposite is not known to be true.

Speaking of concrete proof systems P, they can, sometimes superficially, be sub-divided into two large groups: genuinely logical systems, as conceived by Cook-Reckhow and many other researchers since them, and systems based on totally different principles. Before reviewing each category, let me remark that the most general Definition 5 supplies one universal complexity measure of proofs, which is their *size*, that is, the number of bits needed to write the proof down. As usual, for $\phi \in \text{TAUT}$, its minimum proof size $S_P(\phi)$ in a proof system P is defined as $\min\{|w| | P(w) = \phi\}$. Two proof systems P and Q are *polynomially equivalent* if $S_P(\phi)$ and $S_Q(\phi)$ are so.

3.1 Logic-based proof systems

A *Frege proof system* is simply the polynomial calculus as described in any textbook on mathematical logic. It operates with Boolean expressions built from variables via a set of connectives like $\{\neg, \wedge, \vee, \rightarrow\}$, has a finite number of axiom schemes like $\phi \rightarrow (\phi \vee \psi)$ and a finite number of inference rules like the *modus ponens* rule $\frac{\phi \quad \phi \rightarrow \psi}{\psi}$. The most important fact to remember about Frege proof systems is that Spira's theorem works perfectly well in this setting and all (sound and complete) Frege proof systems are polynomially equivalent [54]. In a sense that can be made quite precise, the Frege proof system captures the class of arguments that can be carried over exclusively using NC^1-concepts.

From the pair (Frege, NC^1) we can go either up or down. Going upward, we arrive at (Extended Frege, P). The *Extended Frege* proof system is extremely robust and can be defined in many equivalent ways. To make the connection to circuit complexity clearer, let us introduce an *extension variable* y_C for any Boolean *circuit* $C(x_1, \ldots, x_n)$ in the original variables $x_1, \ldots, x_n$ and write down the obvious *extension axioms* like $y_C \equiv y_{C'} \vee y_{C''}$ if the final gate in C is an OR-gate and C', C'' are its descendants etc. This enhancement turn out to be so powerful that it does not really matter what proof system is applied on top of it. It can be a Frege proof system or resolution (or width-4 resolution), all resulting systems will be polynomially equivalent. The general idea of extension variables is a recurrent theme in proof complexity (cf. the discussion in Section 2.4), but normally they are used in a controlled environment when the class of admissible circuits C is severely restricted.

Descending from (Frege, NC^1), the next big station is (bounded-depth Frege, AC^0). The term "bounded-depth Frege" is self-explaining: these are Frege proofs in the de Morgan language $\{\neg, \vee, \wedge\}$ (and the connectives can have unbounded arity, of course).

Further down, working with tautologies becomes awkward, and we resolutely switch to the language UNSAT, that is refuting unsatisfiable CNFs. By far the most cherished, popular and well-studied system in proof complexity is that of *resolution*, 1-depth Frege as it were. It operates with clauses and has only one inference rule called *resolution rule*:

$$\frac{C \vee x \qquad D \vee \bar{x}}{C \vee D}.$$

As the bit size of a clause is negligible, the universal size measure $S_R(\tau)$ that, as a reminder that we are dealing with refutations, is traditionally denoted by $S_R(\tau \vdash 0)$, can be safely identified with the minimum number of *lines* in a resolution refutation of τ. Another extremely important related measure is *width*. $w_R(\tau \vdash 0)$ is the minimum w for which there exists a resolution refutation in which all clauses have width (the number of literals) $\leq w$.

For a few results it turns out to be natural to consider a finer gradation between resolution and bounded-depth Frege. Of the corresponding systems I would like to mention only the system $\mathrm{Res}(k)$ operating with k-DNFs instead of clauses. Above bounded-depth Frege virtually any reasonable circuit class leads to an associated proof system: the rule of thumb is always the same, we try to capture the kind of arguments "expressible" by objects efficiently computable within this class. As a good example, in the pair (bounded-depth Frege with MOD_m gates, $\mathsf{ACC}^0[m]$) the proof complexity term is obtained by appending to the language of bounded-Frege gates new connectives $\mathrm{MOD}_{m,a}$ of unbounded areas along with obvious axioms that can be found e.g. in [32, Section 12.6].

3.2 Algebraic and semi-algebraic proofs

In this section I will review only a few of these systems; for further reading I refer to the excellent survey [28].

0 and 1 stand for FALSE and TRUE only in mathematical logic and related disciplines. Elsewhere, they are much better recognized as distinguished real numbers or, more generally, elements of a ring that we will assume to be commutative. Proof systems that we mention here heavily exploit this duality by translating logical clauses as either polynomial equations ($x_1 \vee \bar{x}_2 \vee x_3 \mapsto (1-x_1)x_2(1-x_3) = 0$) or inequalities, linear ($x_1 \vee \bar{x}_2 \vee x_3 \mapsto x_1 + x_3 \geq x_2$) or sometimes even of higher degree as in (3). After that we argue about the resulting system of polynomial equations or inequalities using algebraic or geometric tools. In most cases, the original logical content is lost very quickly (see, though, the end of Section 5) that, of course, is reminiscent of various algebrization techniques in computational complexity. In fact, both sets of concepts were worked out around the same time, and in some cases even by the same people.

The sub-division into algebraic and semi-algebraic proof systems is straightforward, it is determined by whether we work with equations or inequalities. A more subtle distinction that, as far as I can see, does not have any useful analogue in logic-based proof complexity is between *dynamic* and *static* proof systems. Dynamic proof systems may operate with fancy statements but they still have the time-sanctioned, line-by-line, Hilbert-style form: deduce from axioms a bunch of intermediate statements called "lemmas" until you arrive at the final conclusion, possibly just a contradiction, called the "theorem". On the contrary, static proof systems present a proof as a single and supposedly incomprehensible line; ironically, these systems are directly inspired by the **Hilbert** Nullstellensatz theorem.

To recall this theorem, a system of polynomial equations $f_1(x_1, \ldots, x_n) = 0, \ldots f_m(x_1, \ldots, x_n) = 0$ over a field k is unsatisfiable if and only if the ideal in the ring $k[x_1, \ldots, x_n]$ generated by $f_1, \ldots, f_m$ contains 1 or, in other words, there exist polynomials $Q_1, \ldots, Q_m$ such that

$$f_1Q_1 + \ldots + f_mQ_m = 1. \tag{4}$$

In our context, we also have the integrality constraints $x_i \in \{0, 1\}$ that can be incorporated either by adding $x_i^2 - x_i = 0$ to the list of axioms $\{f_1, \ldots, f_m\}$ or just by factoring them out immediately thus working in the ring $\Lambda_n \approx k[x_1, \ldots, x_n]/(x_1^2 - x_1, \ldots, x_n^2 - x_n)$ of multi-linear polynomials. Either way, $Q_1, \ldots, Q_m$ can be assumed to be multi-linear and hence $\deg(Q_1), \ldots, \deg(Q_m) \leq n$.

Thus, a proof in the *Nullstellensatz proof system* is simply a tuple of polynomials $Q_1, \ldots, Q_m$, and the verification procedure consists in checking the validity of (4) by expanding all polynomials and performing cancellations. While the universal size measure is perfectly legal for Nullstellensatz proofs, it is obviously rather unnatural. The complexity of Nullstellensatz proofs is usually measured by their *degree*, defined as $\max\{\deg(Q_1), \ldots, \deg(Q_m)\}$.

The *Polynomial Calculus* introduced in [20] under the name "Gröbner basis algorithms" is just a dynamic version of Nullstellensatz that attempts to break down the proof of $1 \in (f_1, \ldots, f_m)$ into intermediate steps. It has two obvious inference rules

$$\frac{f = 0 \qquad\quad g = 0}{\alpha f + \beta g = 0}\ (\alpha, \beta \in k) \qquad \frac{f = 0}{fg = 0}.$$

Studying size or line complexity in this system already makes better sense than for Nullstellensatz, particularly if we switch to its "symmetrized" version PCR ("Polynomial Calculus with Resolution"). But, still, the measure that is predominantly used for Polynomial Calculus is *degree*, that is the maximum degree of a polynomial appearing in a proof. Polynomial Calculus can be more efficient than Nullstellensatz due to degree-reducing cancellations that may occur at intermediate steps.

Let us turn now to semi-algebraic proof systems, and this time I will start with dynamic ones. Most, if not all of them, were inspired by or even directly borrowed from the work on integer programming.

The *Cutting Planes* proof system (introduced in that context by Gomory and Chvátal, for a comprehensive account see [30, Chapter 19]) operates with *linear* inequalities of the form $c^T x \geq t$, where $c \in \mathbb{Z}^n$ and $t \in \mathbb{Z}$. It has one obvious *convex combination* rule and one non-obvious *Gomory-Chvátal cut* rule:

$$\frac{f \geq 0 \qquad\quad g \geq 0}{\alpha f + \beta g \geq 0}\ (\alpha, \beta \geq 0) \qquad \frac{dc_1x_1 + \ldots + dc_nx_n \geq t}{c_1x_1 + \ldots + c_nx_n \geq \lceil t/d \rceil}\ (d > 0)$$

(note rounding up, not down, all constants here are integers); we also include the obvious axioms $x_i \geq 0,\ x_i \leq 1$, of course. Cutting Planes easily simulates resolution, in particular it is sound and complete which might not be immediately obvious. Like in the theory of threshold circuits, there are two different versions of Cutting Planes: in the first version CP, the coefficients can be arbitrary integers that, although, just as in circuit complexity, they can be assumed to be bounded by a single exponent. In the second version, usually denoted by CP*, they are required to be bounded by *poly*(n) in the *absolute value*.

The universal size measure is of extreme interest for this systems, we will talk about it later. Another complexity measure that has been studied for a long time is proof *rank*. Proof-complexity speaking, it can be defined as the maximum number of applications of the Gomory-Chvátal rules along a path in the proof tree. By Caratheodory's theorem, however, if we look at the overall *depth* instead, i.e., count applications of the convex combination rule as well, this will change the rank by at most a factor of $O(\log n)$. The recent paper [52] initiated the study of *width* in Cutting Planes proofs. For this study, it is absolutely paramount that the convex combination rule is allowed to have unbounded fan-in, that is, it looks way more natural in the dual framework of polytopes in $[0,1]^n$ and integer programming.

The *Lovaśz-Schrijver system* LS_+ operates with *quadratic* constraints $q(x) \geq 0$. It shares the convex combination rule with Cutting Planes, has additional *positive semi-definiteness* axioms $\ell^2 \geq 0$ (that are responsible for the subscript "+") and the inference rules

$$\frac{\ell \geq 0}{\ell x_i \geq 0} \qquad \frac{\ell \geq 0}{\ell(1-x_i) \geq 0}. \tag{5}$$

Here ℓ is an arbitrary *linear* (affine) form so that we stay quadratic after applying them. Size, rank and width are defined in exactly the same way as for Cutting Planes.

The rules (5) clearly look as special cases of one more general rule $\frac{\ell_1 \geq 0 \qquad \ell_2 \geq 0}{\ell_1 \ell_2 \geq 0}$, and the resulting proof system is called $\mathrm{LS}_{+,*}$. There has not been much work done on it, though, probably because it turns out to be *way* harder than LS_+.

Of static semi-algebraic proof systems, I will mention only the one that has been risen by the tide of recent events to practically the King's powers in the realm of semi-algebraic proof systems and, arguably, in the field of propositional proof complexity in general. This is the *Positivstellensatz* proof system.

In full analogy with (4), a system of polynomial inequalities $f_1 \geq 0, \ldots, f_m \geq 0$ is unsatisfiable if and only if there exists a system of polynomials $Q_0, Q_1, \ldots, Q_m$ which are *sums of squares* and such that

$$Q_0 + f_1 Q_1 + \ldots + f_m Q_m = -1;$$

the integrality constraints $x_i^2 - x_i = 0$ can be handled in an arbitrary reasonable way. Like with Nullstellensatz, the primary measure of interest is also degree.

As is usual with many members of the royal family, this system has several different names: Positivstellensatz, Lassierre hierarchy, Sum-of-Squares. We will stick to the latter: it is as informative as "Positivstellensatz" but is much easier to pronounce.

4 Automatizability and Feasible Interpolation

The theory of NP-completeness is often popularly viewed as the difference between "creative search" for a solution and its routine verification. In our case, this translates into the difference between the existence of an efficient propositional proof of a statement and our algorithmic ability to actually find it. One important difference from the computational context is that a short proof may simply not exist at all. The general notion of automatizability approaches this issue in the "relativistic" way.

Definition 6 A proof system P (in the sense of Definition 5) is *automatizable* if there exists an algorithm $\mathbb{A}$ that for every $\phi \in$ TAUT (or UNSAT) satisfies $P(\mathbb{A}(\phi)) = \phi$ and runs *in time* $S_P(\phi)^{O(1)}$.

This certainly looks like a great definition, but one small problem with it is that useful automatizable systems do not seem to exist. [6] proved this for resolution modulo a standard assumption from the parameterized complexity, and [26] extended this to Polynomial Calculus (it should be clear that Definition 6 is not obviously anti-monotone in the strength of the system P). Conditional non-automatizability results for stronger proof systems automatically (seems to be a good word) follow from analogous results for Feasible Interpolation, to be reviewed shortly.

But in the *spirit* of Definition 6, much more can be said positively (well, I probably should clarify here that I mean "algorithmically"). Resolution proof systems of width $\leq w$ can be efficiently found in time $n^{O(w)}$, simply by exhaustively generating all provable clauses of width $\leq w$. In other words, resolution is *width-automatizable.* By the same token, Nullstellensatz and Polynomial Calculus are *degree-automatizable*, that is, refutations of degree $\leq d$ can be found in time $n^{O(d)}$, and Cutting Planes and LS$_+$ are *width-automatizable* as well. Of absolutely uttermost importance is the fact that Sum-of-Squares is degree-automatizable: if refutations of degree d (say, $d = 8$) exist, they can be found in time $n^{O(d)}$ using positive semi-definite programming.

On the first sight, Feasible Interpolation seems to be a somewhat different notion. Assume that we have an unsatisfiable set of constraints that is of the form $\vec{\phi}(\vec{x}, \vec{y}), \vec{\psi}(\vec{x}, \vec{z})$, where all tuples of variables are disjoint. Then for every fixed assignment $\vec{a}$ to the variables $\vec{x}$, at least one of the two sets $\vec{\phi}(\vec{a}, \vec{x})$ and $\vec{\psi}(\vec{a}, \vec{z})$ must be unsatisfiable, and we would like to be able to tell algorithmically which one. Without any further restrictions, this is just a problem in structural complexity: given a pair of disjoint NP-sets, separate them by a set in P. But let us bring in proof-complexity elements.

Definition 7 A proof system P has the *Feasible Interpolation property* if there exists a poly-time algorithm $\mathbb{A}$ that, given a refutation of an unsatisfiable formula of the form $\phi(\vec{x}, \vec{y}) \wedge \vec{\psi}(\vec{x}, \vec{z})$ and an assignment $\vec{a}$ to the variables $\vec{x}$, tells us which of the two formulas $\vec{\phi}(\vec{a}, \vec{y})$, $\vec{\psi}(\vec{a}, \vec{z})$ is unsatisfiable. If both of them are unsatisfiable, the answer can be arbitrary.

A simple observation [20] says that automatizability implies Feasible Interpolation. Applied to previous remarks, it immediately gives that Polynomial Calculus, Nullstellensatz and Sum-of-Squares admit Feasible Interpolation for constant-degree proofs. But now the situation is not so grave even with the standard size measure. Building upon a long chain of works, Pudlák showed Feasible Interpolation for Cutting Planes [45] and Lovász-Schrijver [46]; either result implies Feasible Interpolation for resolution. Whether Feasible Interpolation holds for the Polynomial Calculus is open.

On the other hand, Frege and Extended Frege proof systems do not possess Feasible Interpolation provided RSA is secure [37], and some evidence of a similar nature is also known for bounded-depth Frege. Results of this kind can be re-cast in the following way. Let (U, V) be a pair of disjoint NP-sets that we believe to be non-separable by a set in P; for example, in the work [37] this pair is constructed in such a way that this assumption follows from the security of RSA. Hence, if we can efficiently prove in our system P that $U \cap V = \emptyset$ then P does not possess Feasible Interpolation.

This reasoning can be reversed: if P is known to possess Feasible Interpolation, *then* it can not efficiently prove $U \cap V = \emptyset$ for any P-inseparable pair (U, V) of NP-sets. In other words, we automatically get lower bounds for P, conditioned by the assumption of existence of inseparable disjoint NP-pairs.

Let us moreover assume that all occurrence of the variables in $\vec{x}$ are positive in $\vec{\phi}$ and negative in $\vec{\psi}$. Then, under this assumption, the *Monotone Feasible Interpolation* additionally requires that the algorithm $\mathbb{A}$ is actually a (uniform) *monotone* circuit. Since "monotonically inseparable" pairs are well-known from circuit complexity (like the CLIQUE-COLORING pair), this gives *unconditional* exponential lower bounds on the proof size in any system admitting Monotone Feasible Interpolation. This property is known to hold for Cutting Planes [45] and, hence, also for resolution.

5 State-of-the-art and some directions for future research

In this section, it is particularly important to bear in mind the disclaimer I made on page 3.

In the department of logic-based proof systems, one guiding empirical principle is that it is at least as difficult, and usually way more difficult, to prove lower bound for a propositional proof system than for the corresponding circuit class (for an explanation of this correspondence, see Section 3.1). By this token, lower bounds for *strong* proof systems, like Frege or Extended Frege, seem to be completely out of reach for the moment. What might be potentially more realistic is the ability to prove such bounds modulo some standard complexity or cryptographic assumptions. One possible approach (and motivation!) to this task was outlined in Section 2.3, there might be others as well.

We do not have the least clue how to do it, and, what is even more embarrassing, we do not understand why it is so difficult. This is particularly frustrating since this is precisely what Feasible Interpolation does for weaker systems: are there any inherent reasons for this genuinely philosophical argument to stop at resolution, and why some version of super-interpolation has not taken us any further? I view it as the most important and challenging question in Proof Complexity.

Weaker logic-based proof systems are understood much better, but still there is a steady flow of important results and open problems there.

For PHP_n^{n+1}, exponential lower bounds for bounded-depth Frege proofs were established in the seminal papers [43, 38]. They were extended in [13] to Tseitin tautologies via a very clear reduction to PHP_n^{n+1}, and this is perhaps a good place to mention another striking difference between propositional proof complexity and computational complexity. In the former, the stock of "useful" reductions between different problems is rather limited, and examples are far and few between. [13] is certainly one of them, and just as was the case of PHP_n^{n+1} in [43, 38], the proof works up to depth $\epsilon \log\log n$.

The very recent breakthrough result [44] manages to extend the latter bound (that is, for bounded-depth Frege and Tseitin tautologies) up to depth $\epsilon\sqrt{\log n}$. The significance of this work may not be entirely obvious from the description, so let me say a few words why we should be excited about it. One of our most beloved techniques in both circuit and proof complexities is the method of random restrictions. It is known to be notoriously non-robust: even small deviations from genuine independence leads to catastrophic results. It looks like [44] has started developing techniques that allow for more flexibility and can work with random restrictions in situations where pure independence is hard to get by (that in fact accounts for most situations in proof complexity, as truly independent variable assignments are detrimental to axioms). It can be of course fully expected that their bound will be pushed to exponential, but, perhaps more importantly, it can be hoped that it will allow us to tackle principles like $\boldsymbol{\tau_k^{n,\Delta}}$ and $\mathrm{Circuit}_t(f_n)$. In both cases, the development stopped at Res(k), see [1] and [51], respectively.

Speaking of the main result in [51], it is based on pseudo-random generator principles we discussed in Section 2.3. One of important ingredients in it is the "iterability principle" from the paper [34]: under certain conditions, pseudo-random generators hard in the sense of Section 4 can be iterated (yes, what is easy in computational complexity, can become quite tricky in proof complexity). The hope that structural results like that may be possible and helpful was one of our main motivations behind introducing this concept in [4], and it would be very interesting to see more things of this sort coming out.

Speaking of the pigeonhole principle PHP_n^m, it displays *quite* an interesting behavior when m varies as a function of n and it takes different forms reviewed in Section 2.4. For a large part, this behavior is recorded in my old survey [50]; while some open problems asked there have been solved, there are still quite a few of them that remain open.

Size lower bounds for resolution are comparatively well understood, and in particular we know them for all principles mentioned in Sections 2.1, 2.2, 2.3. Much of the landscape in the area is dominated by the celebrated *width-size relation* by Ben-Sasson and Wigderson [14]:

$$w(\tau_n \vdash 0) \leq O(n \cdot \log S(\tau_n \vdash 0))^{1/2} + w(\tau_n),$$

where τ_n is an arbitrary contradiction in n variables. But still there are some important problems left here. Of those, I would only like to mention the *small clique problem* asked by several different groups of researchers. Let k be a fixed constant and G be a graph with $\chi(G) = k$ and $\omega(G) \leq k-1$. Does resolution always have a refutation of the latter fact that would have size $f(k) \cdot n^C$, where C does not depend on k? As should be clear from the statement, the question is motivated by connections with parameterized proof complexity, see e.g. [15].

Connections with practical SAT solving have been flourishing over several last years. As it turns out, the propositional proof system underlying most state-of-the-art SAT solvers is exactly the resolution proof system, so, theoretically speaking (and oversimplifying), what much of the field is doing is trying to automatize resolution, not being worried by theoretical limitations to this project from [6]. So far, they have been remarkably successful in this endeavor (that of course should be reminiscent of other parts of complexity), but there seems to exist quite a considerably interest on both sides to understand how it can happen, i.e. understand the interplay between theory and practice. For further details, we refer the reader to the survey [42] written for a sister SIGLOG column.

Another quite active topic in which resolution plays a leading role is that of *space complexity* started in [24, 3]. In recent years, it has seen very spectacular advances, like the quadratic lower bound for the so-called *total space* in resolution [16]. For further details I would like to refer the reader to the comprehensive survey [41].

We started this section with reinforcing the duality between propositional proof complexity and circuit classes. In most cases, lower bounds for proof systems follow lower bounds for the corresponding circuit classes, sometimes with a bit of effort. One glaring exception is bounded-depth Frege with MOD_p gates, p a prime. Matching the circuit lower bound here well might be the most intriguing open problem in the department of "concrete" size lower bounds: so far, this is known only for bounded-depth Frege augmented with the Count_p principles as additional *axioms* [11].

Let us move on to algebraic proof systems. Polynomial calculus and Nullstellensatz are at the moment understood fairly well, and lower bounds have been proven for all (that is, when they make sense) principles in Sections 2.1-2.3, including $\mathrm{Circuit}_t(f_n)$ [49, 18, 5]. Given the degree-automatizability of Polynomial calculus, it is of course quite tempting to use it for practical SAT solving, as was the hope in the paper [20] in which it was introduced. Although algebra-based SAT solvers do not seem to be at the moment competitive with the dominating conflict-driven clause learning (resolution-based) technique, it still seems to be a promising direction. We refer the reader to [42, Section 4.5] for further readings on the subject.

Another very interesting direction in the algebraic proof complexity was recently started in [29], where they introduced the so-called *Ideal Proof System.* This is a prominent system tightly connected to algebraic circuit complexity and, arguably, lower bounds in this system imply lower bounds in the latter. Note that this "transfer principle" goes in the direction *opposite* to the one we were discussing above.

By far the most important open problem for dynamic semi-algebraic proof systems (in my view) is understanding their *size*. Recall from Section 4 that, based on Feasible Interpolation, we do have conditional lower bounds for LS_+ and unconditional lower bounds for Cutting Planes. But even the latter can be at the moment achieved only for quite artificial contradictions; say, lower bounds for "normal" principles from Section 2 remain completely elusive. We do need combinatorial and/or geometric techniques to handle them, and, hopefully, the concept of width proposed in [52] (and methods for its analysis, like *w-obstructing polytopes*) might be helpful here. One reason why I think it is very important is because of tight connections with integer programming. Rank measures correspond to the quite wasteful scenario in which we keep *all* new constraints that can be inferred, without even attempting to sort them out into "useful" and "useless". This is in sharp contrast with, say, conflict-driven clause-learning that is extremely picky about which clauses to derive and keep. This "intelligent" behaviour is proof-theoretically captured by size/length measures, and we currently have only very limited tools for analyzing them in the context of semi-algebraic proof systems.

The first degree lower bounds for SOS (Sum-of-Squares) were proven by Grigoriev [27] (using previous ideas from [18]) and "re-discovered"[5] in [55, 56]. They work for random k-CNFs and optimization-based principles in the spirit of Section 2.4.

Then came the "SOS revolution". It has rapidly becomes an extremely dynamic area (even if it is about a static proof system) in which things are happening in front of our eyes at an amazing speed, and that would be fairly pointless to try for a "screenshot" or make predictions as to what will happen next. Let me instead recall, in a telegraphic mode, several well-known facts; we again refer to the survey [9] for additional details.

Even if the contradictions $\mathrm{SSEH}_{G,\delta,\epsilon}$ from Definition 4 look like yet another optimization problem, de facto it is not so, they rather capture a huge and overly important class of *general* methods for proving hardness encapsulated in the Unique Game Conjecture (UGC). Until several years ago, the most promising candidates for hardness of UGC were the *noisy Hypercube* ($G_{xy} = p^{\rho(x,y)}(1-p)^{n-\rho(x,y)}$, $x, y \in \{0,1\}^n$, ρ the Hamming distance) and its modifications and generalizations. All of these were broken by SOS refutations of degree 8 or 10. After that, there has not been a single hardness candidate for UGC that would be universally recognized so by the community.

Quite an interesting sociological question here, I guess, is this: what *exactly* has changed recently? After all, as we remarked in Section 2.4, SOS or another semi-algebraic proof system

[5]With all due respect to the modesty of their authors, there is certainly much more to those papers.

quietly works "behind the hood" in any approximation algorithm based on convex optimization, which immediately gives (usually undocumented) upper bounds for many tautologies of the kind we discussed in Section 2.4.

This question, in my view, has several good answers. The obvious one was already alluded to above: going from the analysis of specific optimization problems to entirely changing the UGC-dominated landscape (that is all about general methods) is a huge qualitative leap that simply brings things to a new level. Then, of course, the whole proof complexity community should be grateful to the main players in the field for *explicitly* highlighting and popularizing the role of semi-algebraic proof systems, opening the hood, as it were. But there is something more interesting to it.

In Section 2.4 I remarked that static proof systems tend to produce proofs as a single incomprehensible line. In many respects, this new line of research on SOS is challenging this state of affairs. Instead, many of them look like a normal Hilbert-style, comprehensive and discursive mathematical argument that simply happens to be formalizable in constant-degree SOS, thus deferring algorithmic application to the mere statement of automatizability. As a model example we can mention one of the crucial papers on the hypercontractivity in SOS [31] that presents such an argument quite explicitly. While some results of this sort were spotted in proof complexity before (see e.g. [28, Section 6]), this time it is about statements that are of great independent interest, mathematical theorems in the strictest sense of this word.

For a brief summary, from the proof complexity point of view, SOS is just the system of "right size". Unlike Frege or Extended Frege, the situation with lower bounds is not hopeless even now, but, on the other hand, we are certainly quite short of industrial methods of proving them like in resolution. And it has established itself as a proof system that *can* do efficiently quite a bit of exciting things well transcending the boundaries of proof complexity. For all these reasons, further bridging the gap between what we know for SOS and what we would like to learn is an extremely exciting and active project, let me refer to the very recent word on the subject in the context of the closely related *Planted Clique Problem* (and SOS, of course) [8].

Last, but not the least, SOS has recently found another remarkable application in a long time of outstanding research on the so-called *extended formulations*. [39] has given a *general* method of converting any true inequality $f \geq 0$ that is hard for SOS w.r.t. a simple (size-like) complexity measure into a matrix of high PSD rank and hence a polytope of high extension complexity. This is loosely reminiscent of proof complexity results about *hardness amplification* [10]... but my space is completely up so I would better stop here.

References

[1] M. Alekhnovich. Lower bounds for k-DNF resolution on random 3-CNFs. *Computational Complexity*, 20(4):597–614, 2011.

[2] M. Alekhnovich, S. Arora, and I. Tourlakis. Toward strong nonapproximability results in the Lovász-Schrijver hierarchy. *Computational Complexity*, 20(4):615–648, 2011.

[3] M. Alekhnovich, E. Ben-Sasson, A. Razborov, and A. Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002.

[4] M. Alekhnovich, E. Ben-Sasson, A. Razborov, and A. Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004.

[5] M. Alekhnovich and A. Razborov. Lower bounds for the polynomial calculus: non-binomial case. *Proceedings of the Steklov Institute of Mathematics*, 242:18–35, 2003.

[6] M. Alekhnovich and A. Razborov. Resolution is not automatizable unless $W[P]$ is tractable. *SIAM Journal on Computing*, 38(4):1347–1363, 2008.

[7] S. Arora and B. Barak. *Computational Complexity: a Modern Approach.* Cambridge University Press, 2009.

[8] B. Barak, S. Hopkins, J. Kelner, P. Kothari, A. Moitra, and A. Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. Technical Report TR16-058, Electronic Colloquium on Computational Complexity, 2016.

[9] B. Barak and D. Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. In *Proceedings of International Congress of Mathematicians (ICM)*, volume IV, pages 509–533, 2014.

[10] P. Beame, T. Huynh, and T. Pitassi. Hardness amplification in proof complexity. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing,*, pages 87–96, 2010.

[11] P. Beame and S. Riis. More on the relative strength of counting principles. In P. Beame and S. Buss, editors, *Proof Complexity and Feasible Arithmetics: DIMACS workshop, April* 21-24, 1996*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol.* 39, pages 13–35. American Math. Soc., 1997.

[12] E. Ben-Sasson. *Expansion in Proof Complexity.* PhD thesis, Hebrew University, 2001.

[13] E. Ben-Sasson. Hard examples for bounded depth Frege. In *Proceedings of the 34th ACM Symposium on the Theory of Computing*, pages 563–572, 2002.

[14] E. Ben-Sasson and A. Wigderson. Short proofs are narrow - resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

[15] O. Beyersdorff, N. Galesi, M. Lauria, and A. Razborov. Parameterized bounded-depth Frege is not optimal. *ACM Transactions on Computation Theory*, 7:article 7, 2012.

[16] I. Bonacina, N. Galesi, and N. Thapen. Total space in resolution. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science*, pages 641–650, 2014.

[17] J. Buresh-Oppenheim, N. Galesi, S. Hoory, A. Magen, and T. Pitassi. Rank bounds and integrality gaps for cutting planes procedures. *Theory of Computing*, 2:65–90, 2006.

[18] S. Buss, D. Grigoriev, R. Impagliazzo, and T. Pitassi. Linear gaps between degrees for the Polynomial Calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62:267–289, 2001.

[19] S. R. Buss. *Bounded Arithmetic.* Bibliopolis, Napoli, 1986.

[20] M. Clegg, J. Edmonds, and R. Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th ACM STOC*, pages 174–183, 1996.

[21] S. Cook and P. Nguyen. *Logical Foundations of Proof Complexity.* Cambridge University Press, 2014.

[22] S. A. Cook and A. R. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.

[23] J. Ding, A. Sly, and N. Sun. Proof of the satisfiability conjecture for large k. Technical Report 1411.0650[math.PR], arXiv e-print, 2014.

[24] J. L. Esteban and J. Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001.

[25] U. Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the 34th ACM Symposium on the Theory of Computing*, pages 534–543, 2002.

[26] N. Galesi and M. Lauria. On the automatizability of polynomial calculus. *Theory of Computing Systems*, 47(2):491–506, 2010.

[27] D. Grigoriev. Linear lower bounds on degrees of Postivestellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259:613–622, 2001.

[28] D. Yu. Grigoriev, E. A. Hirsch, and D. V. Pasechnik. Complexity of semi-algebraic proofs. *Moscow Mathematical Journal*, 2(4):647–679, 2002.

[29] J. Grochow and T. Pitassi. Circuit complexity, proof complexity, and polynomial identity testing. In *Proceedings of the 55th IEEE Symposium on Foundations of Computer Science*, pages 110–119, 2014.

[30] S. Jukna. *Boolean Function Complexity: Advances and frontiers.* Springer-Verlag, 2012.

[31] M. Kauers, R. O'Donnell, Li-Yang Tan, and Y. Zhou. Hypercontractive inequalities via SOS, and the Frankl-Rödl graph. Technical Report arXiv:1212.5324v3 [cs.CC], arXiv e-print, 2012. To appear in *Discrete Analysis.*

[32] J. Krajíček. *Bounded arithmetic, propositional logic and complexity theory.* Cambridge University Press, 1995.

[33] J. Krajíček. Tautologies from pseudo-random generators. *Bulletin of Symbolic Logic*, 7(2):197–212, 2001.

[34] J. Krajíček. Dual weak pigeonhole principle, pseudo-surjective functions, and provability of circuit lower bounds. *Journal of Symbolic Logic*, 69(1):265–286, 2004.

[35] J. Krajíček. *Forcing with Random Variables and Proof Complexity, LMS Lecture Notes Series 382.* Cambridge University Press, 2011.

[36] J. Krajíček and P. Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. *Journal of Symbolic Logic*, 54(3):1063–1079, 1989.

[37] J. Krajíček and P. Pudlák. Some consequences of cryptographical conjectures for S_2^1 and EF. *Information and Computation*, 142:82–94, 1998.

[38] J. Krajíček, P. Pudlák, and A. R. Woods. Exponential lower bounds to the size of bounded depth Frege proofs of the pigeonhole principle. *Random Structures and Algorithms*, 7(1):15–39, 1995.

[39] J. R. Lee, P. Raghavendra, and D. Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, pages 567–576, 2015.

[40] S. Buss M. Bonet and T. Pitassi. Are there hard examples for Frege systems. In *Feasible Mathematics, Volume II*, pages 30–56. Birkhauser, 1994.

[41] J. Nordström. Pebble games, proof complexity and time-space trade-offs. *Logical Methods in Computer Science*, 9:1–63, 2013.

[42] J. Nordström. On the interplay between proof complexity and SAT solving. *ACM SIGLOG News*, 2(3):19–44, 2015.

[43] T. Pitassi, P. Beame, and R. Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational Complexity*, 3:97–140, 1993.

[44] T. Pitassi, B. Rossman, R. A. Servedio, and Li-Yang Tan. Poly-logarithmic Frege depth lower bounds via an expander switching lemma. To appear in STOC 2016, 2016.

[45] P. Pudlák. Lower bounds for resolution and cutting planes proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, 1997.

[46] P. Pudlák. On the complexity of propositional calculus. In *Sets and Proofs, Invited papers from Logic Colloquium'97*, pages 197–218. Cambridge University Press, 1999.

[47] P. Raghavendra and D. Steurer. Graph expansion and the Unique Games Conjecture. In *Proceedings of the 42nd ACM Symposium on the Theory of Computing*, pages 755–764, 2010.

[48] A. Razborov. Bounded Arithmetic and lower bounds in Boolean complexity. In P. Clote and J. Remmel, editors, *Feasible Mathematics II. Progress in Computer Science and Applied Logic, vol.* 13, pages 344–386. Birkhaüser, 1995.

[49] A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7:291–324, 1998.

[50] A. Razborov. Proof complexity of pigeonhole principles. In *Proceedings of the 5th DLT, Lecture Notes in Computer Science,* 2295, pages 100–116, New York/Berlin, 2002. Springer-Verlag.

[51] A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, 181(3):415–472, 2015.

[52] A. Razborov. On the width of semi-algebraic proofs and algorithms. Technical Report TR16-010, Electronic Colloquium on Computational Complexity, 2016.

[53] A. Razborov and S. Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

[54] R. A. Reckhow. On the lengths of proofs in the propositional calculus. Technical Report 87, University of Toronto, 1976.

[55] G. Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proceedings of the* 49*th IEEE Symposium on Foundations of Computer Science*, pages 593–602, 2008.

[56] M. Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proceedings of the* 41*st ACM Symposium on the Theory of Computing*, pages 303–312, 2009.

[57] M. Tulsiani. Lovász-Schrijver reformulation. In *Wiley Encyclopedia of Operations Research & Management Science*. 2011.