Electronic Colloquium on Computational Complexity, Report No. 163 (2018)

ECCC

Adventures in Monotone Complexity and TFNP

Mika Göös† IAS
Pritish Kamath MIT
Robert Robere† Simons Institute
Dmitry Sokolov KTH

September 18, 2018

Abstract

Separations: We introduce a monotone variant of Xor-Sat and show it has exponential monotone circuit complexity. Since Xor-Sat is in NC^2, this improves qualitatively on the monotone vs. non-monotone separation of Tardos (1988). We also show that monotone span programs over $\mathbb{R}$ can be exponentially more powerful than over finite fields. These results can be interpreted as separating subclasses of TFNP in communication complexity.

Characterizations: We show that the communication (resp. query) analogue of PPA (subclass of TFNP) captures span programs over $\mathbb{F}_2$ (resp. Nullstellensatz degree over $\mathbb{F}_2$). Previously, it was known that communication FP captures formulas (Karchmer–Wigderson, 1988) and that communication PLS captures circuits (Razborov, 1995).

Contents

†Work done while M.G. was at Harvard University and R.R. was at University of Toronto.

ISSN 1433-8092

1 Our Results

We study the complexity of *monotone* boolean functions $f\colon \{0,1\}^n \to \{0,1\}$, that is, functions satisfying $f(x) \le f(y)$ for every pair $x \le y$ (coordinate-wise). (An excellent introduction to monotone complexity is the textbook [Juk12].) Our main results are new ***separations*** of monotone models of computation and ***characterizations*** of those models in the language of query/communication complexity. At the core of these results are two conceptual innovations.

1. We introduce a natural *monotone* encoding of the usual CSP satisfiability problem (Section 1.1). This definition unifies many other monotone functions considered in the literature.

2. We extend and make more explicit an intriguing connection between *circuit complexity* and *total* NP *search problems* (TFNP) via communication complexity. Several prior characterizations [KW88, Raz95] can be viewed in this light. This suggests a potentially useful organizational principle for circuit complexity measures; see Section 2 for our survey.

1.1 Monotone $\mathcal{C}$-Sat

The basic conceptual insight in this work is a new simple definition: a *monotone encoding* of the usual constraint satisfaction problem (CSP). For any finite set of constraints $\mathcal{C}$, we introduce a monotone function $\mathcal{C}$-Sat. A general definition is given in Section 3, but for now, consider as an example the set $\mathcal{C} = 3\textsc{Xor}$ of all ternary parity constraints

$$3\textsc{Xor} \;\coloneqq\; \big\{\, (v_1 \oplus v_2 \oplus v_3 = 0),\; (v_1 \oplus v_2 \oplus v_3 = 1) \,\big\}.$$

We define $3\textsc{Xor}\text{-}\textsc{Sat}_n\colon \{0,1\}^N \to \{0,1\}$ over $N \coloneqq |\mathcal{C}|n^3 = 2n^3$ input bits as follows. An input $x \in \{0,1\}^N$ is interpreted as (the indicator vector of) a set of 3Xor constraints over n boolean variables $v_1, \ldots, v_n$ (there are N possible constraints). We define $3\textsc{Xor}\text{-}\textsc{Sat}_n(x) \coloneqq 1$ iff the set x is *unsatisfiable*, that is, no boolean assignment to the v_i exists that satisfies all constraints in x. This is indeed a monotone function: if we flip any bit of x from 0 to 1, this means we are adding a new constraint to the instance, thereby making it even harder to satisfy.

Prior work. Our $\mathcal{C}$-Sat encoding generalizes several previously studied monotone functions.

(NL) Karchmer and Wigderson [KW88] (also [GS92, Pot17, RPRC16] and textbooks [KN97, Juk12]) studied the NL-complete *st-connectivity* problem. This is equivalent to a $\mathcal{C}$-Sat problem with $\mathcal{C}$ consisting of a binary implication $(v_1 \to v_2)$ and unit clauses (v_1) and $(\neg v_1)$.

(P) Raz and McKenzie [RM99] (also [Cha13, CP14, GP14, dRNV16, RPRC16, PR18]) studied a certain P-complete *generation* problem. In hindsight, this is simply Horn-Sat, that is, $\mathcal{C}$ consists of *Horn clauses*: clauses with at most one positive literal, such as $(\neg v_1 \vee \neg v_2 \vee v_3)$.

(NP) Göös and Pitassi [GP14] and Oliveira [Oli15, §3] (also [PR17, PR18]) studied the NP-complete (dual of) Cnf-Sat problem, where $\mathcal{C}$ consists of bounded-width clauses.

These prior works do not exhaust all interesting classes of $\mathcal{C}$, as is predicted by various classification theorems for CSPs [Sch78, FV98, Bul17, Zhu17]. In this work, we focus on *linear* constraints over finite fields $\mathbb{F}_p$ (for example, 3Xor-Sat corresponding to $\mathbb{F}_2$) and over the reals $\mathbb{R}$.

1.2 Separations

First, we show that 3Xor-Sat$_n$ cannot be computed efficiently with monotone circuits.

Theorem 1. 3Xor-Sat$_n$ *requires monotone circuits of size* $2^{n^{\Omega(1)}}$.

This theorem stands in contrast to the fact that there exist fast parallel (non-monotone) algorithms for linear algebra [Mul87]. In particular, 3Xor-Sat is in NC^2. Consequently, our result improves qualitatively on the monotone vs. non-monotone separation of Tardos [Tar88] who exhibited a monotone function in P (computed by solving a semidefinite program) with exponential monotone circuit complexity. For further comparison, another famous candidate problem to witness a monotone vs. non-monotone separation is the *perfect matching* function: it is in RNC^2 [Lov79] while it is widely conjectured to have exponential monotone circuit complexity (a quasipolynomial lower bound was proved by Razborov [Raz85a]).

Span programs. The computational easiness of 3Xor-Sat$_n$ can be stated differently: it can be computed by a linear-size monotone $\mathbb{F}_2$*-span program*. Span programs are a model of computation introduced by Karchmer and Wigderson [KW93] (see also [Juk12, §8] for exposition) with an extremely simple definition. An $\mathbb{F}$*-span program*, where $\mathbb{F}$ is a field, is a matrix $M \in \mathbb{F}^{m\times m'}$ each row of which is labeled by a literal, x_i or $\neg x_i$. We say that the program accepts an input $x \in \{0,1\}^n$ iff the rows of M whose labels are consistent with x (literals evaluating to true on x) span the all-1 row vector. The *size* of a span program is its number of rows m. A span program is *monotone* if all its literals are positive; in this case the program computes a monotone function.

A corollary of Theorem 1 is that monotone $\mathbb{F}_2$-span programs cannot be simulated by monotone circuits without exponential blow-up in size. This improves on a separation of Babai, Gál, and Wigderson [BGW99] who showed that monotone circuit complexity can be quasipolynomially larger than monotone $\mathbb{F}_2$-span program size.

Furthermore, Theorem 1 holds more generally over *any* field $\mathbb{F}$: an appropriately defined function 3Lin($\mathbb{F}$)-Sat$_n$ (ternary $\mathbb{F}$-linear constraints; see Section 3) is easy for monotone $\mathbb{F}$-span programs, but exponentially hard for monotone circuits. No such separation, even superpolynomial, was previously known for fields of characteristic other than 2.

This brings us to our second theorem.

Theorem 2. 3Lin($\mathbb{R}$)-Sat$_n$ *requires monotone* $\mathbb{F}_p$*-span programs of size* $2^{n^{\Omega(1)}}$ *for any prime* p.

In other words: monotone $\mathbb{R}$-span programs can be exponentially more powerful than monotone span programs over finite fields. This separation completes the picture for the relative powers of monotone span programs over distinct fields, since the remaining cases were exponentially separated by Pitassi and Robere [PR18].

Finally, our two results above yield a bonus result in proof complexity as a byproduct: the Nullstellensatz proof system over $\mathbb{R}$ can be exponentially more powerful than the Cutting Planes proof system (see Section 4.2).

Techniques. The new lower bounds are applications of the lifting theorems for monotone circuits [GGKS18] and monotone span programs [PR18]. We show that, generically, if some unsatisfiable formula composed of $\mathcal{C}$ constraints is hard to refute for the Resolution (resp. Nullstellensatz) proof system, then the $\mathcal{C}$-Sat problem is hard for monotone circuits (resp. span programs). Hence we can invoke (small modifications of) known Resolution and Nullstellensatz lower bounds [BR98, BW01, ABRW04]. The key conceptual innovation here is a reduction from unsatisfiable $\mathcal{C}$-CSPs (or their lifted versions) to the monotone Karchmer–Wigderson game for $\mathcal{C}$-Sat. This reduction is extremely slick, which we attribute to having finally found the "right" definition of $\mathcal{C}$-Sat.

1.3 Characterizations

There are two famous "top-down" characterizations of circuit models (both monotone and non-monotone variants) using the language of communication complexity; these characterizations are naturally related to communication analogues of subclasses of TFNP.

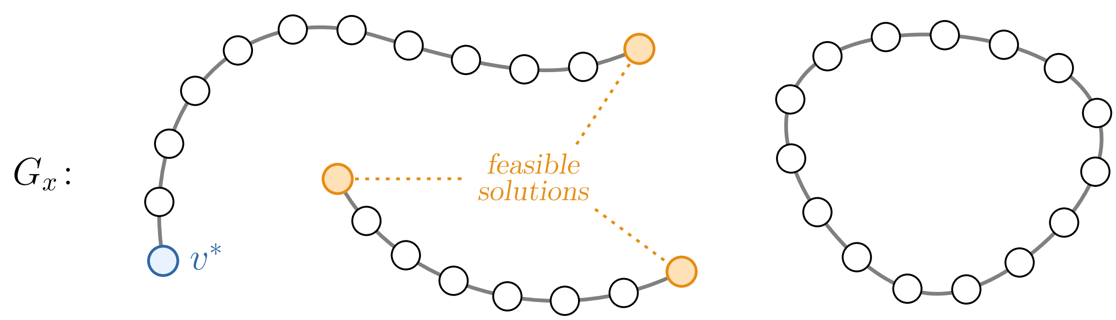
(FP) Karchmer and Wigderson [KW88] showed that the logarithm of the (monotone) formula complexity of a (monotone) function $f\colon \{0,1\}^n \to \{0,1\}$ is equal, up to constant factors, to the communication complexity of the *(monotone) Karchmer–Wigderson game*:

Search problem KW(f) **[resp. KW$^+$(f)]**	=	*input:*	a pair $(x, y) \in f^{-1}(1) \times f^{-1}(0)$
		output:	an $i \in [n]$ with $x_i \neq y_i$ [resp. $x_i > y_i$]

We summarize this by saying that *the communication analogue of* FP *captures formulas.* Here FP $\subseteq$ TFNP is the classical (Turing machine) class of total NP search problems efficiently solved by deterministic algorithms [MP91].

(PLS) Razborov [Raz95] (see also [Pud10, Sok17]) showed that the logarithm of the (monotone) circuit complexity of a function $f\colon \{0,1\}^n \to \{0,1\}$ is equal, up to constant factors, to the least cost of a PLS-*protocol* solving the KW(f) (or KW$^+$(f)) search problem. Here a PLS-*protocol* (Definition 4 in Appendix A) is a natural communication analogue of PLS $\subseteq$ TFNP [JPY88]. We summarize this by saying that *the communication analogue of* PLS *captures circuits.*

We contribute a third characterization of this type: *the communication analogue of* PPA *captures* $\mathbb{F}_2$*-span programs.* The class PPA [Pap94] is a well-known subclass of TFNP embodying the combinatorial principle "every graph with an odd degree vertex has another". Informally, a search problem is in PPA if for every n-bit input x we may describe implicitly an undirected graph $G_x = (V, E)$ (typically of size exponential in n; the edge relation is computed by a polynomial-size circuit) such that G has degree at most 2, there is a distinguished degree-1 vertex $v^* \in V$, and every other degree-1 vertex $v \in V$ is associated with a feasible solution to the instance x (that is, the solution can be efficiently computed from v).



Communication PPA. The communication analogue of PPA is defined canonically by letting the edge relation be computed by a (deterministic) communication protocol. Specifically, first fix a two-party search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$, that is, Alice gets $x \in \mathcal{X}$, Bob gets $y \in \mathcal{Y}$, and their goal is to find a *feasible solution* in $S(x, y) := \{o \in \mathcal{O} : (x, y, o) \in S\}$. A PPA-*protocol* Π solving S consists of a vertex set V, a distinguished vertex $v^* \in V$, and for each vertex $v \in V$ there is an associated solution $o_v \in \mathcal{O}$ and a protocol Π_v (taking inputs from $\mathcal{X} \times \mathcal{Y}$). Given an input (x, y), the protocols Π_v implicitly describe a graph $G = G_{x,y}$ on the vertex set V as follows. The output of protocol Π_v on input (x, y) is interpreted as a subset $\Pi_v(x, y) \subseteq V$ of size at most 2. We define $\{u, v\} \in E(G)$ iff $u \in \Pi_v(x, y)$ and $v \in \Pi_u(x, y)$. The correctness requirements are:

(C1) if $\deg(v^*) \neq 1$, then $o_{v^*} \in S(x, y)$.
(C2) if $\deg(v) \neq 2$ for $v \neq v^*$, then $o_v \in S(x, y)$.

The *cost* of Π is defined as $\log |V| + \max_v |\Pi_v|$ where $|\Pi_v|$ is the communication cost of Π_v. Finally, define $\mathsf{PPA}^{\mathsf{cc}}(S)$ as the least cost of a PPA-protocol that solves S.

For a (monotone) function f, define $\mathrm{SP}_{\mathbb{F}}(f)$ (resp. $\mathrm{mSP}_{\mathbb{F}}(f)$) as the least size of a (monotone) $\mathbb{F}$-span program computing f. Our characterization is in terms of $S := \mathrm{KW}(f)$.

Theorem 3. *For any boolean function f, we have $\log \mathrm{SP}_{\mathbb{F}_2}(f) = \Theta(\mathsf{PPA}^{\mathsf{cc}}(\mathrm{KW}(f)))$. Furthermore, if f is monotone, we have $\log \mathrm{mSP}_{\mathbb{F}_2}(f) = \Theta(\mathsf{PPA}^{\mathsf{cc}}(\mathrm{KW}^+(f)))$.*

Query PPA. Our second characterization concerns the *Nullstellensatz* proof system; see Section 3 for the standard definition. Span programs and Nullstellensatz are known to be connected via interpolation [PS98] and lifting [PR18]. Given our first characterization (Theorem 3), it is no surprise that a companion result should hold in query complexity: *the query complexity analogue of* PPA *captures the degree of Nullstellensatz refutations over* $\mathbb{F}_2$.

The query analogue of PPA is defined in the same way as the communication analogue, except we replace protocols by (deterministic) decision trees. In fact, query PPA was already studied by Beame et al. [BCE+98] who separated query analogues of different subclasses of TFNP. To define it, first fix a search problem $S \subseteq \{0,1\}^n \times \mathcal{O}$, that is, on input $x \in \{0,1\}^n$ the goal is to find a *feasible solution* in $S(x) := \{o \in \mathcal{O} : (x, o) \in S\}$. A PPA*–decision tree* $\mathcal{T}$ solving S consists of a vertex set V, a distinguished vertex $v^* \in V$, and for each vertex $v \in V$ there is an associated solution $o_v \in \mathcal{O}$ and a decision tree $\mathcal{T}_v$ (querying bits of an n-bit input). Given an input $x \in \{0,1\}^n$, the decision trees $\mathcal{T}_v$ implicitly describe a graph $G = G_x$ on the vertex set V as follows. The output of $\mathcal{T}_v$ on input x is interpreted as a subset $\mathcal{T}_v(x) \subseteq V$ of size at most 2. We then define $\{u, v\} \in E(G)$ iff $u \in \mathcal{T}_v(x)$ and $v \in \mathcal{T}_u(x)$. The correctness requirements are the same as before, *(C1)* and *(C2)*. The *cost* of $\mathcal{T}$ is defined as the maximum over all $v \in V$ and all inputs x of the number of queries made by $\mathcal{T}_v$ on input x. Finally, define $\mathsf{PPA}^{\mathsf{dt}}(S)$ as the least cost of a PPA–decision tree that solves S.

With any unsatisfiable n-variate boolean CSP F one can associate a canonical search problem:

CSP search problem $S(F)$ $=$	*input:*	an n-variate truth assignment $x \in \{0,1\}^n$
	output:	constraint C of F falsified by x (i.e., $C(x) = 0$)

Theorem 4. *The $\mathbb{F}_2$-Nullstellensatz degree of an k-CNF formula F equals $\Theta(\mathsf{PPA}^{\mathsf{dt}}(S(F)))$.*

The easy direction of this characterization is that Nullstellensatz degree lower bounds $\mathsf{PPA}^{\mathsf{dt}}$. This fact was already observed and exploited by Beame et al. [BCE+98] to prove lower bounds for $\mathsf{PPA}^{\mathsf{dt}}$. Our contribution is to show the other (less trivial) direction.

Let us finally mention a related result in Turing machine complexity due to Belovs et al. [BIQ+17]: a circuit-encoded version of Nullstellensatz is PPA-complete. Their proof is highly nontrivial whereas our characterizations admit relatively short proofs, owing partly to us working with simple nonuniform models of computation.

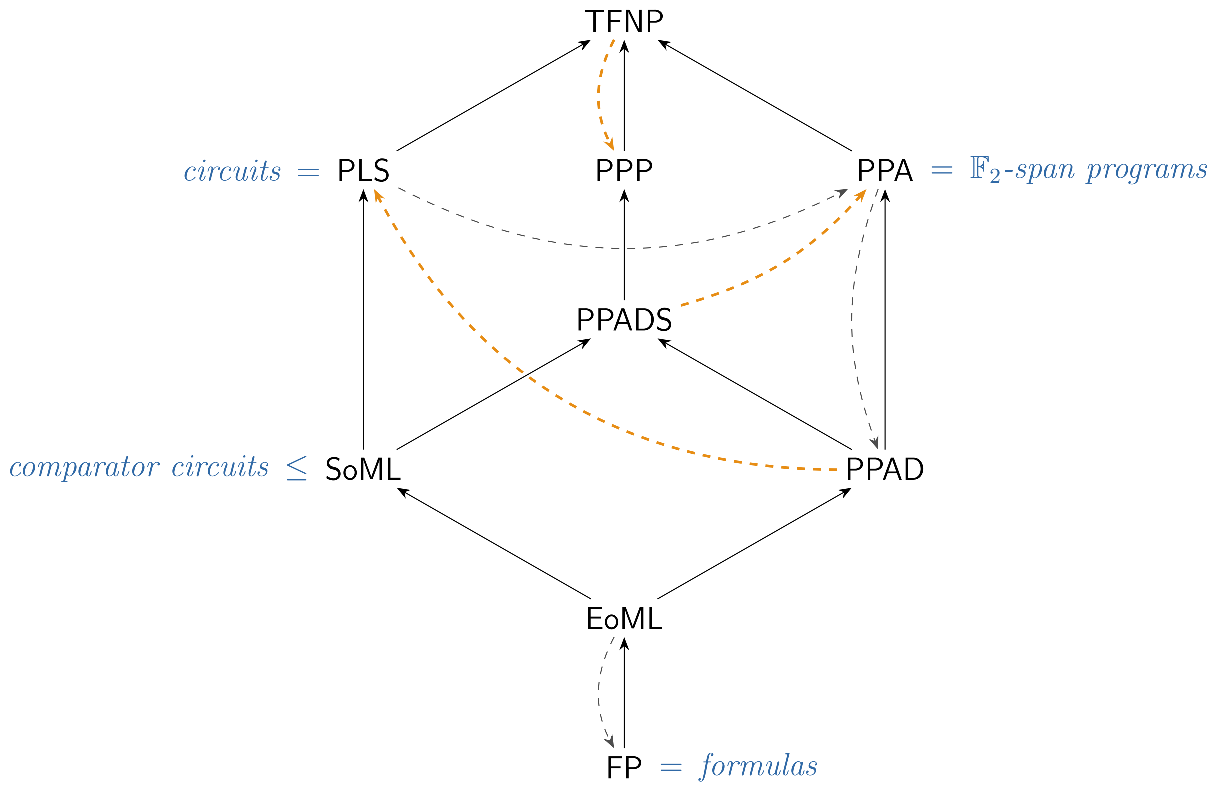


Figure 1: The landscape of communication search problem classes (uncluttered by the usual 'cc' superscripts). A solid arrow $\mathsf{C}_1 \to \mathsf{C}_2$ denotes $\mathsf{C}_1 \subseteq \mathsf{C}_2$, and a dashed arrow $\mathsf{C}_1 \dashrightarrow \mathsf{C}_2$ denotes $\mathsf{C}_1 \not\subseteq \mathsf{C}_2$ (in fact, an exponential separation). The yellow arrows are new separations. Some classes can characterize other models of computation (printed in blue). See Appendix A for definitions.

2 Survey: Communication TFNP

Given the results in Section 1, it is natural to examine other communication analogues of subclasses of TFNP. The goal in this section is to explain the current state of knowledge as summarized in Figure 1. The formal definitions of the communication classes appear in Appendix A.

TFNP. As is customary in structural communication complexity [BFS86, HR90, GPW16] we formally define $\mathsf{TFNP}^{\mathsf{cc}}$ (resp. $\mathsf{PLS}^{\mathsf{cc}}$, $\mathsf{PPA}^{\mathsf{cc}}$, etc.) as the class of all two-party n-bit search problems that admit a nondeterministic protocol (resp. PLS-protocol, PPA-protocol, etc.) of communication cost polylog(n). For example, Karchmer–Wigderson games $\mathrm{KW}(f)$ and $\mathrm{KW}^+(f)$, for an n-bit boolean function f, have efficient nondeterministic protocols: guess a $\log n$-bit coordinate $i \in [n]$ and check that $x_i \neq y_i$ or $x_i > y_i$. Hence these problems are in $\mathsf{TFNP}^{\mathsf{cc}}$. In fact, a converse holds: any total two-party search problem with nondeterministic complexity c can be reduced to $\mathrm{KW}^+(f)$ for some 2^c-bit *partial* monotone function f, see [Gál01, Lemma 2.3]. In summary, the study of total NP search problems in communication complexity is equivalent to the study of monotone Karchmer–Wigderson games for *partial* monotone functions.

Sometimes a *partial* function f can be canonically extended into a *total* one f' without increasing the complexity of $\mathrm{KW}(f)$ (or $\mathrm{KW}^+(f)$). This is possible whenever $\mathrm{KW}(f)$ lies in a communication class that captures some associated model of computation. For example, if $\mathrm{KW}(f)$ is solved by a deterministic protocol (resp. PLS-protocol, PPA-protocol) then the Karchmer–Wigderson connection can build us a corresponding formula (resp. circuit, $\mathbb{F}_2$-span program) that computes some total extension f' of f. Consequently, separating two communication classes that capture two monotone models is *equivalent* to separating the monotone models themselves.

FP. Raz and McKenzie [RM99] showed an exponential separation between monotone formula size and monotone circuit size. This can be rephrased as $\mathsf{PLS}^{\mathsf{cc}} \not\subseteq \mathsf{FP}^{\mathsf{cc}}$. Their technique is much more general: they develop a query-to-communication lifting theorem for deterministic protocols (see also [GPW15] for exposition). By plugging in known query complexity lower bounds against the class EoML (combinatorial subclass of CLS [DP11] introduced by [HY17, FGMS18]), one can obtain a stronger separation $\mathsf{EoML}^{\mathsf{cc}} \not\subseteq \mathsf{FP}^{\mathsf{cc}}$.

A related question is whether *randomization* helps in solving $\mathsf{TFNP}^{\mathsf{cc}}$ problems. Lower bounds against randomized protocols have applications in proof complexity [IPU94, BPS07, HN12, GP14] and algorithmic game theory [RW16, BR17, GR18, Rou18, GS18, BDN18]. In particular, some of these works (for finding Nash equilibria) have introduced a communication analogue of the PPAD-complete End-of-Line problem, which we will continue to study in Section 4.2.

PLS. Razborov's [Raz85b] famous monotone circuit lower bound for the *clique/coloring* problem (which is in $\mathsf{PPP}^{\mathsf{cc}}$) can be interpreted as an exponential separation $\mathsf{PPP}^{\mathsf{cc}} \not\subseteq \mathsf{PLS}^{\mathsf{cc}}$. We show a stronger separation $\mathsf{PPAD}^{\mathsf{cc}} \not\subseteq \mathsf{PLS}^{\mathsf{cc}}$ using the End-of-Line problem in Section 4.2. Note that this is even slightly stronger than Theorem 1, which only implies $\mathsf{PPA}^{\mathsf{cc}} \not\subseteq \mathsf{PLS}^{\mathsf{cc}}$.

PPA(D). In light of our characterization of $\mathsf{PPA}^{\mathsf{cc}}$, we may interpret the inability of monotone $\mathbb{F}_2$-span program to efficiently simulate monotone circuits [PR18] as a separation $\mathsf{PLS}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$. We show an incomparable separation $\mathsf{PPADS}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$ in Section 4.3.

In the other direction, prior work implies $\mathsf{PPA}^{\mathsf{cc}} \not\subseteq \mathsf{PPAD}^{\mathsf{cc}}$ as follows. Pitassi and Robere [PR18] exhibit a monotone f (in hindsight, one can take $f := 3\textsc{Xor-Sat}_n$) computable with a small monotone $\mathbb{F}_2$-span program (hence $\mathrm{KW}^+(f) \in \mathsf{PPA}^{\mathsf{cc}}$) and such that $\mathrm{KW}^+(f)$ has an exponentially large $\mathbb{R}$-*partition number* (see Section 3 for a definition); however, we observe that all problems in $\mathsf{PPAD}^{\mathsf{cc}}$ have a small $\mathbb{R}$-partition number (see Remark 4.2).

PPP. There are no lower bounds against $\mathsf{PPP}^{\mathsf{cc}}$ for an *explicit* problem in $\mathsf{TFNP}^{\mathsf{cc}}$. However, we can show non-constructively the existence of $\mathrm{KW}(f) \in \mathsf{TFNP}^{\mathsf{cc}}$ such that $\mathrm{KW}(f) \notin \mathsf{PPP}^{\mathsf{cc}}$, which implies $\mathsf{PPP}^{\mathsf{cc}} \neq \mathsf{TFNP}^{\mathsf{cc}}$. Indeed, we argue in Remark 4.1 that every S reduces to $\mathrm{KW}^+(3\textsc{Cnf-Sat}_N)$ over $N := \exp(O(\mathsf{PPP}^{\mathsf{cc}}(S)))$ variables. Applying this to $S := \mathrm{KW}(f)$ for an n-bit f, we conclude that f is a (non-monotone) projection of $3\textsc{Cnf-Sat}_N$ for $N := \exp(O(\mathsf{PPP}^{\mathsf{cc}}(\mathrm{KW}(f))))$. In particular, if $\mathrm{KW}(f) \in \mathsf{PPP}^{\mathsf{cc}}$ (i.e., $\mathsf{PPP}^{\mathsf{cc}}(\mathrm{KW}(f)) \leq \mathrm{polylog}(n)$), then f is in non-uniform quasipoly-size NP. Therefore $\mathrm{KW}(f) \notin \mathsf{PPP}^{\mathsf{cc}}$ for a random f.

EoML, SoML, and comparator circuits. One prominent circuit model that currently lacks a characterization via a $\mathsf{TFNP}^{\mathsf{cc}}$ subclass is *comparator circuits* [MS92, CFL14]. These circuits are composed only of *comparator gates* (taking two input bits and outputting them in sorted order) and input literals (positive literals in the monotone case).

We can show an upper bound better than $\mathsf{PLS}^{\mathsf{cc}}$ for comparator circuits. Indeed, we introduce a new class SoML generalizing EoML [HY17, FGMS18] as follows. Recall that EoML is the class of problems reducible to End-of-Metered-Line: we are given a directed graph of in/out-degree at most 1 with a distinguished source vertex v^* (in-degree 0), and moreover, each vertex is labeled with an integer "meter" that is strictly decreasing along directed paths; a solution is any sink or source distinct from v^*. The complete problem defining SoML is SINK-OF-METERED-LINE, which is the same as END-OF-METERED-LINE except only sinks count as solutions. It is not hard (left as an exercise) to adapt the characterization of circuits via $\mathsf{PLS}^{\mathsf{cc}}$ [Raz95, Pud10, Sok17] to show that $\mathrm{KW}(f)$ is in $\mathsf{SoML}^{\mathsf{cc}}$ if f is computed by a small comparator circuit. However, we suspect that the converse (SoML-protocol for $\mathrm{KW}(f)$ implies a comparator circuit) is false.

2.1 Open problems

In query complexity, the relative complexities of TFNP subclasses are almost completely understood [BCE+98, BM04, Mor05]. In communication complexity, by contrast, there are huge gaps in our understanding as can be gleaned from Figure 1. For example:

(1) There are no lower bounds against classes $\mathsf{PPADS}^{\mathsf{cc}}$ and $\mathsf{PPP}^{\mathsf{cc}}$ for an explicit problem in $\mathsf{TFNP}^{\mathsf{cc}}$. For starters, show $\mathsf{PLS}^{\mathsf{cc}} \not\subseteq \mathsf{PPADS}^{\mathsf{cc}}$ or $\mathsf{PPA}^{\mathsf{cc}} \not\subseteq \mathsf{PPADS}^{\mathsf{cc}}$.

(2) Find computational models captured by $\mathsf{EoML}^{\mathsf{cc}}$, $\mathsf{SoML}^{\mathsf{cc}}$, $\mathsf{PPAD}^{\mathsf{cc}}$, $\mathsf{PPADS}^{\mathsf{cc}}$, $\mathsf{PPP}^{\mathsf{cc}}$.

(3) Query-to-communication lifting theorems are known for FP [RM99], PLS [GGKS18], PPA [PR18]. Prove more. (This is one way to attack Question (1) if proved for PPADS.)

(4) Prove more separations. For example, can our result $\mathsf{PPADS}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$ be strengthened to $\mathsf{SoML}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$? This is closely related to whether monotone comparator circuits can be more powerful than monotone $\mathbb{F}_2$-span programs (no separation is currently known).

3 Preliminaries

$\mathcal{C}$-SAT. Fix an alphabet Σ (potentially infinite, e.g., $\Sigma = \mathbb{R}$). Let $\mathcal{C}$ be a finite set of k-ary predicates over Σ, that is, each $C \in \mathcal{C}$ is a function $C\colon \Sigma^k \to \{0,1\}$. We define a monotone function $\mathcal{C}\text{-}\mathrm{SAT}_n\colon \{0,1\}^N \to \{0,1\}$ over $N = |\mathcal{C}|n^k$ input bits as follows. An input $x \in \{0,1\}^N$ is interpreted as a $\mathcal{C}$-CSP instance, that is, x is (the indicator vector of) a set of $\mathcal{C}$-constraints, each applied to a k-tuple of variables from $v_1, \dots, v_n$. We define $\mathcal{C}\text{-}\mathrm{SAT}_n(x) \coloneqq 1$ iff the $\mathcal{C}$-CSP x is *unsatisfiable*: no assignment $v \in \Sigma^n$ exists such that $C(v) = 1$ for all $C \in x$.

For a field $\mathbb{F}$, we define $k\mathrm{LIN}(\mathbb{F})$ as the set of all $\mathbb{F}$-linear equations of the form

$$\textstyle\sum_{i\in[k]} a_i v_i \;=\; a_0, \qquad \text{where } a_i \in \{0, \pm 1\}.$$

In particular, we recover $3\mathrm{XOR}\text{-}\mathrm{SAT}_n$ defined in Section 1 essentially as $3\mathrm{LIN}(\mathbb{F}_2)\text{-}\mathrm{SAT}_n$. We could have allowed the a_i to range over $\mathbb{F}$ when $\mathbb{F}$ is finite, but we stick with the above convention as it ensures that the set $k\mathrm{LIN}(\mathbb{R})$ is always finite.

Boolean alphabets. We assume henceforth that all alphabets Σ contain distinguished elements 0 and 1. We define $\mathcal{C}_{\mathrm{bool}}$ to be the constraint set obtained from $\mathcal{C}$ by restricting each $C \in \mathcal{C}$ to the boolean domain $\{0,1\}^k \subseteq \Sigma^k$. Moreover, if F is a $\mathcal{C}$-CSP, we write F_{bool} for the $\mathcal{C}_{\mathrm{bool}}$-CSP obtained by restricting the constraints of F to boolean domains. Consequently, any $S(F_{\mathrm{bool}})$ associated with a $\mathcal{C}$-CSP F is a *boolean* search problem.

Algebraic partitions. We say that a subset $A \subseteq \mathcal{X} \times \mathcal{Y}$ is *monochromatic* for a two-party search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ if there is some $o \in \mathcal{O}$ such that $o \in S(x,y)$ for all $(x,y) \in A$. Moreover, if $M \in \mathbb{F}^{\mathcal{X}\times\mathcal{Y}}$ is a matrix, we say M is *monochromatic* if the support of M is monochromatic. For any field $\mathbb{F}$, an $\mathbb{F}$-*partition* of a search problem S is a set $\mathcal{M}$ of rank-1 matrices $M \in \mathbb{F}^{\mathcal{X}\times\mathcal{Y}}$ such that $\sum_{M\in\mathcal{M}} M = \mathbb{1}$ and each $M \in \mathcal{M}$ is monochromatic for S. The *size* of the partition is $|\mathcal{M}|$. The $\mathbb{F}$-*partition number* $\chi_{\mathbb{F}}(S)$ is the least size of an $\mathbb{F}$-partition of S. In the following characterization, recall that we use $\mathrm{SP}_{\mathbb{F}}$ and $\mathrm{mSP}_{\mathbb{F}}$ to denote (monotone) span program complexity.

Theorem 5 ([Gál01])**.** *For any boolean function f and any field $\mathbb{F}$, $\mathrm{SP}_{\mathbb{F}}(f) = \chi_{\mathbb{F}}(\mathrm{KW}(f))$. Furthermore, if f is monotone then $\mathrm{mSP}_{\mathbb{F}}(f) = \chi_{\mathbb{F}}(\mathrm{KW}^+(f))$.*

Nullstellensatz. Let $P \coloneqq \{p_1 = 0, p_2 = 0, \ldots, p_m = 0\}$ be an unsatisfiable system of polynomial equations in $\mathbb{F}[z_1, z_2, \ldots, z_n]$ for a field $\mathbb{F}$. An $\mathbb{F}$-*Nullstellensatz refutation* of P is a sequence of polynomials $q_1, q_2, \ldots, q_m \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ such that $\sum_{i=1}^m q_i p_i = 1$ where the equality is syntactic. The *degree* of the refutation is $\max_i \deg(q_i p_i)$. The $\mathbb{F}$-*Nullstellensatz degree* of P, denoted $\mathrm{NS}_{\mathbb{F}}(P)$, is the least degree of an $\mathbb{F}$-Nullstellensatz refutation of P.

Moreover, if F is a k-CNF formula (or a boolean k-CSP), we often tacitly think of it as a polynomial system P_F by using the standard encoding (e.g., $(z_1 \vee \neg z_2) \rightsquigarrow (1 - z_1)z_2 = 0$) and also including the *boolean axioms* $z_i^2 - z_i = 0$ in P_F if we are working over $\mathbb{F} \neq \mathbb{F}_2$.

Lifting theorems. Let $S \subseteq \{0,1\}^n \times \mathcal{O}$ be a boolean search problem and $g\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ a two-party function, usually called a *gadget*. The composed search problem $S \circ g^n \subseteq \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{O}$ is defined as follows: Alice holds $x \in \mathcal{X}^n$, Bob holds $y \in \mathcal{Y}^n$, and their goal is to find an $o \in S(z)$ where $z \coloneqq g^n(x,y) = (g(x_1,y_1), \ldots, g(x_n,y_n))$. We focus on the usual *index* gadget $\textsc{Ind}_m\colon [m] \times \{0,1\}^m \to \{0,1\}$ given by $\textsc{Ind}_m(x,y) \coloneqq y_x$.

The main results of [GGKS18, PR18] can be summarized as follows (we define more terms below).

Theorem 6. *Let $k \geq 1$ be a constant and let $m = m(n) \coloneqq n^C$ for a large enough constant $C \geq 1$. Then for any an unsatisfiable boolean n-variate k-CSP F,*

$$\begin{aligned}
\text{[GGKS18]:}\quad & \mathsf{PLS}^{\mathsf{cc}}(S(F) \circ \textsc{Ind}_m^n) = \mathsf{PLS}^{\mathsf{dt}}(S(F)) \cdot \Theta(\log n),\\
\text{[PR18]:}\quad & \mathsf{PPA}^{\mathsf{cc}}(S(F) \circ \textsc{Ind}_m^n) = \mathsf{PPA}^{\mathsf{dt}}(S(F)) \cdot \Theta(\log n),\\
\text{[PR18]:}\quad & \log \chi_{\mathbb{F}}(S(F) \circ \textsc{Ind}_m^n) = \mathrm{NS}_{\mathbb{F}}(F) \cdot \Theta(\log n), \qquad \forall \mathbb{F} \in \{\mathbb{F}_p, \mathbb{R}\}.
\end{aligned}$$

For aesthetic reasons, we have used $\mathsf{PLS}^{\mathsf{dt}}(S(F))$ here to denote the *Resolution width* of F (introduced in [BW01]), which is how the result of [GGKS18] was originally stated. (But one can check that the query analogue of PLS, obtained by replacing protocols with decision trees in Definition 4, is indeed equivalent to Resolution width.) We also could not resist incorporating our new characterizations of $\mathsf{PPA}^{\mathsf{cc}}$ and $\mathsf{PPA}^{\mathsf{dt}}$ to interpret the result of [PR18] specialized to $\mathbb{F}_2$.

4 Proofs of Separations

In this section, we show lower bounds for $\mathcal{C}$-$\textsc{Sat}$ against monotone circuits (Theorem 1) and monotone span programs (Theorem 2), plus some bonus results ($\mathsf{PPAD}^{\mathsf{cc}} \not\subseteq \mathsf{PLS}^{\mathsf{cc}}$, $\mathsf{PPADS}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$, Nullstellensatz degree over $\mathbb{R}$ vs. Cutting Planes).

4.1 Reduction

The key to our lower bounds is a new reduction. We show that a lifted version of $S(F_{\text{bool}})$, where F is an unsatisfiable $\mathcal{C}$-CSP, reduces to the monotone Karchmer–Wigderson game for $\mathcal{C}$-Sat. Note that we require F to be unsatisfiable over its original alphabet Σ, but the reduction is from the booleanized (and hence easier-to-refute) version of F.

Lemma 7. *Let F be an unsatisfiable $\mathcal{C}$-CSP. Then $S(F_{\text{bool}}) \circ \textsc{Ind}_m^n$ reduces to $\mathrm{KW}^+(\mathcal{C}\text{-}\textsc{Sat}_{nm})$.*

Proof. Suppose the $\mathcal{C}$-CSP F consists of constraints $C_1, \dots, C_t$ applied to variables $z_1, \dots, z_n$. We reduce $S(F_{\text{bool}}) \circ \textsc{Ind}_m^n \subseteq [m]^n \times (\{0,1\}^m)^n \times [t]$ to the problem $\mathrm{KW}^+(f) \subseteq f^{-1}(1) \times f^{-1}(0) \times [N]$ where $f \coloneqq \mathcal{C}\text{-}\textsc{Sat}_{mn}$ over $N \coloneqq |\mathcal{C}|(mn)^k$ input bits. The two parties compute locally as follows.

Alice: Given $(x_1, \dots, x_n) \in [m]^n$, Alice constructs a $\mathcal{C}$-CSP over variables $\{v_{i,j} : (i,j) \in [n] \times [m]\}$ that is obtained from F by renaming its variables $z_1, \dots, z_n$ to $v_{1,x_1}, \dots, v_{n,x_n}$ (in this order). Since F was unsatisfiable, so is Alice's variable-renamed version of it. Thus, when interpreted as an indicator vector of constraints, Alice has constructed a 1-input of $\mathcal{C}\text{-}\textsc{Sat}_{mn}$.

Bob: Given $y \in (\{0,1\}^m)^n$, Bob constructs a $\mathcal{C}$-CSP over variables $\{v_{i,j} : (i,j) \in [n] \times [m]\}$ as follows. We view y naturally as a boolean assignment to the variables $v_{i,j}$. Bob includes in his $\mathcal{C}$-CSP instance all possible $\mathcal{C}$-constraints C applied to the $v_{i,j}$ such that C is satisfied under the assignment y (i.e., $C(y) = 1$). This is clearly a satisfiable $\mathcal{C}$-CSP instance, as the assignment y satisfies all Bob's constraints. Thus, when interpreted as an indicator vector of constraints, Bob has constructed a 0-input of $\mathcal{C}\text{-}\textsc{Sat}_{mn}$.

It remains to argue that any solution to $\mathrm{KW}^+(\mathcal{C}\text{-}\textsc{Sat}_{mn})$ gives rise to a solution to $S(F_{\text{bool}}) \circ \textsc{Ind}_m^n$. Indeed, a solution to $\mathrm{KW}^+(\mathcal{C}\text{-}\textsc{Sat}_{mn})$ corresponds to a $\mathcal{C}$-constraint C that is present in Alice's $\mathcal{C}$-CSP but not in Bob's. By Bob's construction, such a C must be violated by the assignment y (i.e., $C(y) = 0$). Since all Alice's constraints involve only variables $v_{1,x_1}, \dots, v_{n,x_n}$, the constraint C must in fact be violated by the partial assignment to the said variables, which is $z = \textsc{Ind}_m^n(x, y)$. Thus the constraint of F from which C was obtained via renaming is a solution to $S(F_{\text{bool}}) \circ \textsc{Ind}_m^n$. ☐

Remark 4.1 (Generic reduction to Cnf-Sat)**.** We claim that any problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ that lies in one of the known subclasses of $\mathsf{TFNP}^{\mathsf{cc}}$ (as listed in Section 2) reduces efficiently to $\mathrm{KW}^+(k\textsc{Cnf-Sat}_n)$ for constant k (one can even take $k = 3$ by standard reductions). Let us sketch the argument for $S \in \mathsf{PPP}^{\mathsf{cc}}$; after all, better reductions are known for $\mathsf{PLS}^{\mathsf{cc}}$ and $\mathsf{PPA}^{\mathsf{cc}}$, namely to Horn-Sat and 3Xor-Sat (see Lemma 10).

Proof sketch. Let $\Pi \coloneqq (V, v^*, o_v, \Pi_v)$ be a PPP-protocol solving S of cost $c \coloneqq \mathsf{PPP}^{\mathsf{cc}}(S)$. We may assume wlog that all the Π_v have constant communication cost $k \le O(1)$ by embedding the protocol trees of the Π_v as part of the implicitly described bipartite graph. In particular, we view each Π_v as a function $\mathcal{X} \times \mathcal{Y} \to \{0,1\}^k$ where the output is interpreted according to some fixed map $\{0,1\}^k \to V$. Consider a set of $n \coloneqq k|V|$ ($|V| \le 2^c$) boolean variables $\{z_{v,i} : (v,i) \in V \times [k]\}$ with the intuitive interpretation that $z_{v,i}$ is the i-th output bit of Π_v. We may encode the correctness conditions for Π as an unsatisfiable $2k$-CNF formula F over the $z_{v,i}$ that has, for each $\{v,u\} \in \binom{V}{2}$, clauses requiring that the outputs of Π_v and Π_u (as encoded by the $z_{v,i}$) should point to distinct vertices. Finally, we note that computing the i-th output bit $(\Pi_v)_i \colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ reduces to a large enough constant-size index gadget $\textsc{Ind}_{O(1)}$ (which embeds any two-party function of communication complexity $k \le O(1)$). Therefore S naturally reduces to $S(F) \circ \textsc{Ind}_{O(1)}^n$, which by Lemma 7 reduces to $\mathrm{KW}^+(2k\textsc{Cnf-Sat}_{O(n)})$, as desired.

4.2 Monotone circuit lower bounds

Xor-Sat. The easiest result to prove is Theorem 1: an exponential monotone circuit lower bound for 3Xor-Sat_n. By the characterization of [Raz95] it suffices to show

$$\mathsf{PLS}^{\mathsf{cc}}(\mathrm{KW}^+(3\text{Xor-Sat}_n)) \;\geq\; n^{\Omega(1)}. \tag{1}$$

Urquhart [Urq87] exhibited unsatisfiable n-variate 3Xor-CSPs F (aka *Tseitin formulas*) requiring linear Resolution width, that is, $\mathsf{PLS}^{\mathsf{dt}}(S(F)) \geq \Omega(n)$ in our notation. Hence Theorem 6 implies that $\mathsf{PLS}^{\mathsf{cc}}(S(F) \circ \text{Ind}^n_m) \geq \Omega(n)$ for some $m = n^{O(1)}$. By the reduction in Lemma 7, we get that $\mathsf{PLS}^{\mathsf{cc}}(\mathrm{KW}^+(3\text{Xor-Sat}_{nm})) \geq \Omega(n)$. (Note that 3Xor has a boolean alphabet, so $F = F_{\text{bool}}$.) This yields the claim (1) by reparameterizing the number of variables.

Lin(F)-Sat. More generally, we can prove a similar lower bound over any field $\mathbb{F} \in \{\mathbb{F}_p, \mathbb{R}\}$:

$$\mathsf{PLS}^{\mathsf{cc}}(\mathrm{KW}^+(3\text{Lin}(\mathbb{F})\text{-Sat}_n)) \;\geq\; n^{\Omega(1)}. \tag{2}$$

Fix such an $\mathbb{F}$ henceforth. This time we start with a $k\text{Lin}(\mathbb{F})$-CSP introduced in [BGIP01] for $\mathbb{F} = \mathbb{F}_p$ (aka *mod-p Tseitin formulas*), but the definition generalizes to any field. The CSP is constructed based on a given directed graph $G = (V, E)$ that is *regular*: in-deg(v) = out-deg$(v) = k/2$ for all $v \in V$. Fix also a distinguished vertex $v^* \in V$. Then $F = F_{G,\mathbb{F}}$ is defined as the following $k\text{Lin}(\mathbb{F})$-CSP over variables $\{z_e : e \in E\}$:

$$\forall v \in V: \quad \sum_{(v,u)\in E} z_{(v,u)} \;-\; \sum_{(u,v)\in E} z_{(u,v)} \;=\; \mathbb{1}_{v^*}(v), \tag{$F_{G,\mathbb{F}}$}$$

where $\mathbb{1}_{v^*}(v^*) = 1$ and $\mathbb{1}_{v^*}(v) = 0$ for $v \neq v^*$. This system is unsatisfiable because the sum over $v \in V$ of the RHS equals 1 whereas the sum of the LHS equals 0 (each variable appears once with a positive sign, once with a negative sign).

We claim that the booleanized k-CSP F_{bool} (more precisely, its natural k-CNF encoding) has linear Resolution width, that is, $\mathsf{PLS}^{\mathsf{dt}}(S(F_{\text{bool}})) \geq \Omega(n)$ in our notation. Indeed, the constraints of F_{bool} are $k/2$-*robust* in the sense that if a partial assignment $\rho \in \{0, 1, *\}^k$ fixes the value of a constraint of F_{bool}, then ρ must set more than $k/2$ variables. Alekhnovich et al. [ABRW04, Theorem 3.1] show that if k is a large enough constant, there exist regular expander graphs G such that F_{bool} (or any k-CSP with $\Omega(k)$-robust constraints) has Resolution width $\Omega(n)$, as desired.

Combining the above with the lifting theorem in Theorem 6 and the reduction in Lemma 7 yields $\mathsf{PLS}^{\mathsf{cc}}(k\text{Lin}(\mathbb{F})\text{-Sat}_n) \geq n^{\Omega(1)}$ for large enough k. Finally, we can reduce the arity from k to 3 by a standard trick. For example, given the linear constraint $a_1v_1 + a_2v_2 + a_3v_3 + a_4v_4 = a_0$ we can introduce a new auxiliary variable u and two equations $a_1v_1 + a_2v_2 + u = 0$ and $-u + a_3v_3 + a_4v_4 = a_0$. In general, we replace each equation on $k > 3$ variables with a collection of $k-2$ equations by introducing $k-3$ auxiliary variables to create an equisatisfiable instance. This shows that $k\text{Lin}(\mathbb{F})\text{-Sat}_n$ reduces to (i.e., is a monotone projection of) $3\text{Lin}(\mathbb{F})\text{-Sat}_{kn}$, which concludes the proof of (2).

$\mathsf{PPAD}^{\mathsf{cc}} \not\subseteq \mathsf{PLS}^{\mathsf{cc}}$ via End-of-Line. Consider the $\mathbb{R}$-linear system $F = F_{G,\mathbb{R}}$ defined above. We observe that $S(F_{\text{bool}})$ is in fact equivalent to (a query version of) the PPAD-complete End-of-Line problem. In the End-of-Line problem, we are given a directed graph of in/out-degree at most 1 and a distinguished source vertex v^* (in-degree 0); the goal is to find a sink or a source distinct from v^* (cf. Definition 5). On the other hand, in $S(F_{\text{bool}})$ we are given a boolean assignment $z \in \{0, 1\}^E$, which can be interpreted as (the indicator vector of) a subset of edges defining a (spanning) subgraph G_z of G; the goal is to find a vertex $v \in V$ such that either

(1) $v = v^*$ and out-deg$(v) \neq$ in-deg$(v) + 1$ in G_z; or
(2) $v \neq v^*$ and out-deg$(v) \neq$ in-deg(v) in G_z.

The only essential difference between $S(F_{\text{bool}})$ and End-of-Line is that the graph G_z can have in/out-degree a large constant $k/2$ rather than 1. But there is a standard reduction between the two problems [Pap94]: we may locally interpret a vertex $v \in V(G_z)$ with out-deg(v) = in-deg$(v) = \ell$ as ℓ distinct vertices of in/out-degree 1. This reduction also shows that the lifted problem $S(F_{\text{bool}}) \circ \textsc{Ind}_m$ for $m = n^{O(1)}$ admits a $O(\log n)$-cost PPAD-protocol, and is thus in $\mathsf{PPAD}^{\mathsf{cc}}$. By contrast, we proved above that this problem is not in $\mathsf{PLS}^{\mathsf{cc}}$ (for appropriate G).

Remark 4.2 (Algebraic partitions for $\mathsf{PPAD}^{\mathsf{cc}}$)**.** We claim that every problem $S \in \mathsf{PPAD}^{\mathsf{cc}}$ admits a small $\mathbb{Z}$-partition, and hence a small $\mathbb{F}$-partition over any field $\mathbb{F}$. More precisely, we argue that $\log \chi_{\mathbb{Z}}(S) \leq O(\mathsf{PPAD}^{\mathsf{cc}}(S))$. Indeed, let $\Pi \coloneqq (V, v^*, o_v, \Pi_v)$ be an optimal PPAD-protocol for S. We define a $\mathbb{Z}$-partition $\mathcal{M}$ by describing it as a nondeterministic protocol for S whose accepting computations output weights in $\mathbb{Z}$ (interpreted as values of the entries of an $M \in \mathcal{M}$): On input (x, y), guess a vertex $v \in V$; if v is a sink in $G_{x,y}$, accept with weight 1; if v is a source distinct from v^*, accept with weight -1; otherwise reject (i.e., weight 0). This protocol accepts with overall weight $\#(\text{sinks}) - \#(\text{non-distinguished sources}) = 1$ on every input (x, y), as desired.

A similar argument yields an analogous query complexity bound $\mathrm{NS}_{\mathbb{Z}}(F) \leq O(\mathsf{PPAD}^{\mathsf{dt}}(S(F)))$ where $\mathsf{PPAD}^{\mathsf{dt}}(S)$ is the least cost of a PPAD–*decision tree* (Definition 5) solving S.

$\mathbb{R}$-Nullstellensatz vs. Cutting Planes. By the above remark, F_{bool} for $F = F_{G,\mathbb{R}}$ admits a low-degree—in fact, constant-degree—Nullstellensatz refutation over $\mathbb{R}$. Nullstellensatz degree behaves well with respect to compositions: if we compose F_{bool} with a gadget $\textsc{Ind}^n_m$, $m = n^{O(1)}$ (see, e.g., [GGKS18, §8] how this can be done), the Nullstellensatz degree can only increase by the query complexity of the gadget, which is $O(\log n)$ for $\textsc{Ind}^n_m$. This gives us an $n^{O(1)}$-variate boolean k-CSP $F' \coloneqq F_{\text{bool}} \circ \textsc{Ind}^n_m$ (where k is constant [GGKS18, §8]) such that $\mathrm{NS}_{\mathbb{R}}(F') \leq O(\log n)$. On the other hand, we can invoke the strong version of the main result of [GGKS18]: if F has Resolution width w, then $F \circ \textsc{Ind}^n_m$ requires Cutting Planes refutations of length $n^{\Omega(w)}$. In summary, F' witnesses that $\mathbb{R}$-Nullstellensatz can be exponentially more powerful than log of Cutting Planes length.

4.3 Monotone span program lower bounds

Let us prove Theorem 2: $3\textsc{Lin}(\mathbb{R})\text{-}\textsc{Sat}_n$ requires exponential-size monotone $\mathbb{F}_p$-span programs, i.e.,

$$\chi_{\mathbb{F}_p}(\mathrm{KW}^+(3\textsc{Lin}(\mathbb{R})\text{-}\textsc{Sat}_n)) \;\geq\; n^{\Omega(1)}. \tag{3}$$

Using Theorem 6 and Lemma 7 similarly as in Section 4.2, it suffices to show that $\mathrm{NS}_{\mathbb{F}_p}(F_{\text{bool}}) \geq n^{\Omega(1)}$, for some unsatisfiable $k\textsc{Lin}(\mathbb{R})$-CSP F where k is a constant. To this end, we consider an $\mathbb{R}$-linear system $F = F_{G,U,\mathbb{R}}$ that generalizes $F_{G,\mathbb{R}}$ defined above:

$$\forall v \in V: \quad \sum_{(v,u)\in E} z_{(v,u)} - \sum_{(u,v)\in E} z_{(u,v)} \;=\; \mathbb{1}_U(v), \tag{$F_{G,U,\mathbb{R}}$}$$

where $\mathbb{1}_U\colon V \to \{0,1\}$ is the indicator function for $U \subseteq V$. This is unsatisfiable as long as $U \neq \emptyset$. Combinatorially, the boolean search problem $S(F_{\text{bool}})$ can be interpreted as an End-of-ℓ-Lines problem for $\ell \coloneqq |U|$: given a graph with distinguished source vertices U, find a sink or a source not in U. It is important to have many distinguished sources, $|U| \geq n^{\Omega(n)}$, as otherwise $S(F_{\text{bool}})$ is in $\mathsf{PPAD}^{\mathsf{dt}}$ [HG18] and hence F_{bool} has too low an $\mathbb{F}_p$-Nullstellensatz degree (by Remark 4.2).

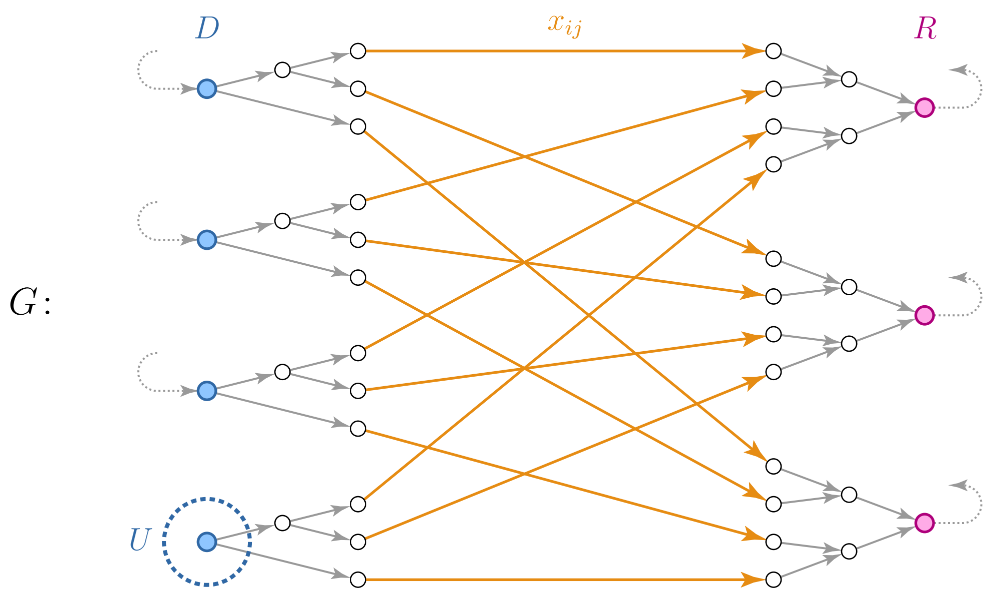


Figure 2: Graph $G = (V, E)$, a bounded-degree version of the biclique $D \times R$.

Nullstellensatz lower bound. To show $\mathrm{NS}_{\mathbb{F}_p}(F_{\mathrm{bool}}) \geq n^{\Omega(1)}$ for an appropriate $F = F_{G,U,\mathbb{R}}$, we adapt a result of Beame and Riis [BR98]. They proved a Nullstellensatz lower bound for a related *bijective pigeonhole* principle P_n whose underlying graph has *unbounded* degree; we obtain a bounded-degree version of their result by a reduction.

Lemma 8 ([BR98, §8])**.** *Fix a prime p. The following system of polynomial equations over variables $\{x_{ij} : (i,j) \in D \times R\}$, where $|D| = n$ and $|R| = n - n^{\Omega(1)}$, requires $\mathbb{F}_p$-Nullstellensatz degree $n^{\Omega(1)}$:*

$$
\begin{array}{rrrcll}
\text{(i)} & \forall i \in D: & \sum_{j \in R} x_{ij} & = & 1 & \text{“each pigeon occupies a hole”,} \\
\text{(ii)} & \forall j \in R: & \sum_{i \in D} x_{ij} & = & 1 & \text{“each hole houses a pigeon”,} \\
\text{(iii)} & \forall i \in D, \{j, j'\} \in \binom{R}{2}: & x_{ij}x_{ij'} & = & 0 & \text{“no pigeon occupies two holes”,} \\
\text{(iv)} & \forall j \in R, \{i, i'\} \in \binom{D}{2}: & x_{ij}x_{i'j} & = & 0 & \text{“no hole houses two pigeons”.}
\end{array}
\qquad (P_n)
$$

We construct a natural bounded-degree version G of the complete bipartite graph $D \times R$ and show that each constraint of F_{bool} for $F = F_{G,U,\mathbb{R}}$ is a low-degree $\mathbb{F}_p$-Nullstellensatz consequence of P_n. Hence, if F_{bool} admits a low-degree $\mathbb{F}_p$-Nullstellensatz proof, so does P_n (see, e.g., [BGIP01, Lemma 1] for composing proofs), which contradicts Lemma 8.

The directed graph $G = (V, E)$ is obtained from the complete bipartite graph $D \times R$ as illustrated in Figure 2 (for $|D| = 4$ and $|R| = 3$). Specifically, each vertex of degree d in $D \times R$ is replaced with a binary tree of height $\log d$. The result is a layered graph with the first and last layers identified with D and R, respectively. We also add a “feedback” edge from each vertex in R to a vertex in D according to some arbitrary injection $R \to D$ (dashed edges in Figure 2). The vertices in D not incident to feedback edges will form the set U (singleton in Figure 2).

This defines a boolean 3-CSP F_{bool} for $F = F_{G,U,\mathbb{R}}$ over variables $\{z_e : e \in E\}$. In order to reduce P_n to F_{bool}, we define an affine map between the variables x_{ij} of P_n and z_e of F_{bool}. Namely, for a feedback edge e we set $z_e \coloneqq 1$, and for every other $e = (v, u)$ we set

$$z_{(v,u)} \coloneqq \sum_{i \in D_v \ j \in R_u} x_{ij},$$
$$\text{where} \quad D_v \coloneqq \{i \in D : v \text{ is reachable from } i \text{ without using feedback edges}\},$$
$$R_u \coloneqq \{j \in R : j \text{ is reachable from } u \text{ without using feedback edges}\}.$$

Note in particular that this map naturally identifies the edge-variables z_e in the middle of G (yellow edges) with the variables x_{ij} of P_n. The other variables z_e are simply affinely dependent on the middle edge-layer. We then show that from the equations of P_n we can derive each constraint of F_{bool}. Recall that the constraint for $v \in V$ requires that the *out-flow* $\sum_{(v,u)\in E} z_{(v,u)}$ equals the *in-flow* $\sum_{(u,v)\in E} z_{(u,v)}$ (plus 1 iff $v \in U$).

$v \notin D \cup R$: Suppose v is on the left side of G (right side is handled similarly) so that $z_{(v,u)} = \sum_{j \in R_u} x_{ij}$ for some fixed $i \in D$. The out-flow is

$$\textstyle\sum_{(v,u)\in E} z_{(v,u)} = \sum_{(v,u)\in E} \sum_{j\in R_u} x_{ij} = \sum_{j \in R_v} x_{ij}. \tag{4}$$

On the other hand, v has a unique incoming edge (u^*, v) so the in-flow is $\sum_{(u,v)\in E} z_{(u,v)} = z_{(u^*,v)} = \sum_{j\in R_v} x_{ij}$, which equals (4).

$v \in D$: (Case $v \in R$ is handled similarly). The in-flow equals 1 (either $v \in U$ so that we have the $+1$ term from $\mathbb{1}_U(v)$; or $v \notin U$ and the value of a feedback-edge variable gives $+1$). The out-flow equals $\sum_{j\in R_v} x_{ij} = \sum_{j \in R} x_{ij} = 1$ by (4), $R_v = R$, and (ii).

Finally, we can verify the boolean axioms $z_e^2 = z_e$. This holds trivially for feedback edges e. Let $e = (v, u)$ be an edge in the left side of G (right side is similar) so that $z_e = \sum_{j\in R_u} x_{ij}$ for some fixed $i \in D$. We have $z_e^2 = (\sum_{j \in R_u} x_{ij})^2 = \sum_{j\in R_u} x_{ij}^2 = \sum_{j \in R_u} x_{ij} = z_e$ by (iii) and the boolean axioms for P_n.

This concludes the reduction and hence the proof of (3).

$\mathsf{PPADS}^{\mathsf{cc}} \not\subseteq \mathsf{PPA}^{\mathsf{cc}}$ via End-of-ℓ-Lines. It is straightforward to check that F_{bool} for $F = F_{G,U,\mathbb{R}}$ is in the query class $\mathsf{PPADS}^{\mathsf{dt}}$ (Definition 6). In particular, in the PPADS–decision tree, we can define the distinguished vertex v^* as being associated with any vertex from U. Similarly, the lifted problem $S' \coloneqq S(F_{\text{bool}}) \circ \textsc{Ind}_n^m$ for $m = n^{O(1)}$ is in the communication class $\mathsf{PPADS}^{\mathsf{cc}}$. By contrast, we just proved that $\chi_{\mathbb{F}_2}(S') \geq n^{\Omega(1)}$, which implies that $S' \notin \mathsf{PPA}^{\mathsf{cc}}$.

5 Proofs of Characterizations

In this section, we prove our characterizations for $\mathsf{PPA}^{\mathsf{cc}}$ (Theorem 3) and $\mathsf{PPA}^{\mathsf{dt}}$ (Theorem 4).

5.1 Communication PPA = span programs

We first show that communication PPA captures $\mathbb{F}_2$-span program size. Constructing a span program from a PPA-protocol is almost immediate from Gál's [Gál01] characterization of span program size (Theorem 5). The other direction is more involved and proceeds in two steps: (1) we show that $3\textsc{Xor-Sat}_n$ is complete for (monotone) span programs under (monotone) projections, and then (2) give a PPA-protocol for $3\textsc{Xor-Sat}_n$.

Span programs from PPA-protocols. To show $\log \mathrm{SP}_{\mathbb{F}_2}(f) \leq O(\mathsf{PPA}^{\mathsf{cc}}(\mathrm{KW}(f)))$ for a boolean function f, we apply the below lemma with $S := \mathrm{KW}(f)$ and use the characterization $\mathrm{SP}_{\mathbb{F}_2}(f) = \chi_{\mathbb{F}_2}(\mathrm{KW}(f))$ in Theorem 5. The monotone case is similar.

Lemma 9. *For any search problem $S \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ we have $\log \chi_{\mathbb{F}_2}(S) \leq O(\mathsf{PPA}^{\mathsf{cc}}(S))$.*

Proof. From a PPA-protocol $\Pi := (V, v^*, o_v, \Pi_v)$ we can obtain canonically a nondeterministic protocol Γ for S. The protocol Γ computes as follows on input (x, y): guess a vertex $v \in V$; if $v = v^*$ and $\deg(v) \neq 1$ in $G_{x,y}$, then accept (with solution o_v); if $v \neq v^*$ and $\deg(v) = 1$ in $G_{x,y}$, then accept (with solution o_v); otherwise reject. In particular, Γ runs $\Pi_v(x, y)$ and then $\Pi_u(x, y)$ for each of the two potential neighbors $u \in \Pi_v(x, y)$. The communication cost is thus at most thrice that of Π. Since we started with a PPA-protocol, it follows that Γ accepts each input (x, y) an odd number of times. This implicitly defines an $\mathbb{F}_2$-partition for S of log-size $O(\mathsf{PPA}^{\mathsf{cc}}(S))$. ☐

PPA-protocols from span programs. As mentioned above, the converse is more involved. We begin by showing that 3Xor-Sat$_n$ is complete for $\mathbb{F}_2$-span programs under projections.

Lemma 10. *Let f be a (monotone) boolean function computable by a (monotone) $\mathbb{F}_2$-span program of size s. Then f can be written as a (monotone) projection of* 3Xor-Sat$_{s^2}$.

Proof. Let M be an $\mathbb{F}_2$-span program for f. We may assume wlog that it is an $s \times s$ matrix with $0, 1$ entries and with each row labeled by an input literal, x_i or $\neg x_i$ (or just x_i in the monotone case). By a change of basis we may assume that, instead of the all-1 row vector, the target is to span the row vector $(0, 0, \ldots, 0, 1)$. Let us thus write $M = [A\ b]$ where A is an $s \times (s-1)$ matrix and b is an $s \times 1$ vector. This suggests the following alternative interpretation of the span program M: given an input $x \in \{0, 1\}^n$, accept if and only if the corresponding system of linear equations $A_{(x)} w = b_{(x)}$ consistent with x is *unsatisfiable*; observe that this is witnessed by some linear combination of rows yielding the vector $(0, 0, \ldots, 0, 1)$. This is nearly a projection of 3Xor-Sat, except, the number of variables occurring in each linear equation in $Aw = b$ may be greater than 3. This is straightforward to fix by a standard reduction (already described in Section 4.2): we replace each equation on $k > 3$ variables with a collection of $k - 2$ equations by introducing $k - 3$ auxiliary variables to create an equisatisfiable instance. The final instance has at most s^2 variables and s^2 equations. ☐

The following lemma completes the proof that any span program implies a PPA-protocol. We prove the lemma only for the monotone game $\mathrm{KW}^+(f)$ as it implies the same bound for $\mathrm{KW}(f)$.

Lemma 11. $\mathsf{PPA}^{\mathsf{cc}}(\mathrm{KW}^+(\text{3Xor-Sat}_n)) \leq O(\log n)$.

Proof. Write $Az = b$ for the list of all $N := 2n^3$ many 3Xor-equations over n variables $z_1, \ldots, z_n$. In the game $\mathrm{KW}^+(\text{3Xor-Sat}_n)$ Alice holds a subset $x \subseteq [N]$ of the rows of $Az = b$ defining an unsatisfiable system $A_x z = b_x$, and Bob holds a subset $y \subseteq [N]$ defining a satisfiable system $A_y z = b_y$. Their goal is to find an equation which is included in Alice's system but not in Bob's. We fix henceforth some satisfying assignment $w \in \mathbb{F}_2^n$ to Bob's system. It suffices to find an equation in Alice's system that w does not satisfy.

For convenience, we assume that the graph $G_{x,y}$ implicitly described by the soon-to-be-defined PPA-protocol $\Pi = (V, v^*, o_v, \Pi_v)$ can have maximum degree $O(1)$ instead of 2. The modified correctness conditions are:

(C1') if $\deg(v^*)$ is even, then o_{v^*} is a feasible solution,
(C2') if $\deg(v)$ is odd for $v \neq v^*$, then o_v is a feasible solution.

This assumption can be made wlog due to a standard reduction [Pap94] already discussed in Section 4.2 (e.g., a degree-$2k$ vertex can be locally split into k separate degree-2 vertices).

For further simplicity, we first describe a PPA-protocol when Alice's system $A_x z = b_x$ satisfies:

$(*)$
1. The vector b_x has only a single 1 entry, say, in position j.
2. Every variable in $A_x z = b_x$ appears in at most two equations.
3. Every equation contains at most four variables (i.e., we relax the 3XOR assumption).

The PPA-protocol is defined as follows. The vertex set V will contain, for each $i \in [N]$, a vertex v_i corresponding to the i-th equation $a_i z = b_i$ of $Az = b$ (with the label o_{v_i} naturally naming that equation) and a separate distinguished vertex v^* (whose label o_{v^*} is arbitrary). For $v \in V$ the protocol Π_v computes as follows.

- If $v = v^*$, Alice outputs v_j as the sole neighbor.
- If $v = v_i$, Alice checks if the i-th equation is in her input x.
  - If not, the protocol outputs the empty set (resulting in $\deg(v) = 0$ in $G_{x,y}$).
  - If yes, Bob tells Alice the set of variables Z that appear in $a_i z = b_i$ and that are set to 1 under w. Then Alice outputs all vertices that correspond to equations in x containing variables from Z and, if $i = j$, the vertex v^*.

Note that Π_v communicates $O(\log n)$ bits, as each equation contains at most four variables. Since every variable appears in at most two equations, $G_{x,y}$ will have maximum degree at most 5 (where 4 comes from arity of equations, and 1 from v^*). Let us check the correctness requirements.

$v = v^*$: Observe that v^* has degree 1 (with neighbor v_j), so *(C1')* is trivially satisfied.

$v = v_j$: Suppose v_j has odd degree. Recall that v_j is associated with equation $a_j z = b_j$ that uniquely has $b_j = 1$. By construction, Alice holds the j-th equation and an even number of its variables are set to true in Bob's w (since v_j has v^* as an additional neighbor), meaning $a_i z = b_i$ is violated by w. Hence the j-th equation is a feasible solution, as required by *(C2')*.

$v = v_i$: Suppose v_i ($\neq v_j$) has odd degree. Recall that v_i is associated with equation $a_i z = b_i$ where $b_i = 0$. By construction, Alice holds the i-th equation and an odd number of its variables are set to true in Bob's w, meaning $a_i z = b_i$ is violated by w. Hence the i-th equation is a feasible solution, as required by *(C2')*.

This concludes the proof under the simplifying assumptions $(*)$. It remains to show how to re-interpret Alice's input to satisfy the assumptions $(*)$.

Starting with any unsatisfiable 3XOR system $A_x z = b_x$, Alice can choose a minimal subset $x' \subseteq x$ such that, viewing x' as an indicator vector, $x' \cdot [A\ b] = (0, 0, \ldots, 0, 1)$. The subsystem $A_{x'} z = b_{x'}$ is still unsatisfiable, but now we ensure that $b_{x'}$ contains an odd number of 1s and each variable z_i occurs in an even number of equations of $A_{x'} z = b_{x'}$.

Alice re-interprets her input as follows. First, to eliminate all 1s in $b_{x'}$ except for one, Alice chooses a matching of the 1s of $b_{x'}$ except for one; this induces a partial matching of the rows of $Az = b$. For each pair of equations $a_i z = b_i$ and $a_{i'} z = b_{i'}$ that are matched, Alice changes b_i and $b_{i'}$ to 0 and adds to the sums $a_i z$ and $a_{i'} z$ a new variable that will always take value 1 in the PPA-protocol. (Note that this increases the number of variables per equation from 3 to 4). Next, consider a variable z_i and suppose it occurs in $2k$ of Alice's equations. Alice creates k copies $z_i^1, \ldots, z_i^k$ of z_i, each z_i^j replacing two occurrences of z_i. In the PPA-protocol, when Alice needs the value of a z_i^j, she asks Bob for the value of z_i. $\square$

5.2 Query PPA = Nullstellensatz

We now show that query PPA captures $\mathbb{F}_2$-Nullstellensatz degree. Showing that $\mathbb{F}_2$-Nullstellensatz degree lower bounds $\mathsf{PPA}^{\mathsf{dt}}$ complexity was already proven by Beame et al. [BCE⁺98], but we include the simple argument for completeness. Our contribution is to show the (less trivial) converse.

NS refutations from PPA–decision trees. The following is a query analogue of Lemma 9.

Lemma 12. $\mathrm{NS}_{\mathbb{F}_2}(F) \le O(\mathsf{PPA}^{\mathsf{dt}}(S(F)))$ *for any unsatisfiable k-CNF formula F.*

Proof. Suppose $F := \bigwedge_{i\in[m]} C_i$, and let p_i be the natural polynomial encoding of C_i (see Section 3) so that $p_i(x) = 0$ iff $C_i(x) = 1$. Fix a PPA–decision tree $\mathcal{T} := (V, v^*, o_v, \mathcal{T}_v)$ of cost $d := \mathsf{PPA}^{\mathsf{dt}}(S(F))$. For each $v \in V$, we can define a depth-$3d$ decision tree $\mathcal{S}_v$ such that $\mathcal{S}_v(x) = 1$ iff (1) $v = v^*$ and $\deg(v^*) \ne 1$ in G_x, or (2) $v \ne v^*$ and $\deg(v) = 1$ in G_x. (First run $\mathcal{T}_v(x)$ and then $\mathcal{T}_u(x)$ for the two potential neighbors $u \in \mathcal{T}_v(x)$.) We can then convert each $\mathcal{S}_v$ into a degree-$3d$ $\mathbb{F}_2$-polynomial s_v in the standard way (s_v is the sum over all accepting paths of $\mathcal{S}_v$ of the product of the literals, x_i or $(1 - x_i)$, recording the query outcomes on that path). Since the s_v came from a PPA–decision tree, where each x is accepted by an odd number of the $\mathcal{S}_v$, we have that $\sum_{v\in V} s_v(x) = 1$ for all x. Moreover, we have that $p_{i_v}(x) = 0 \Rightarrow s_v(x) = 0$ where i_v is such that $o_v = C_{i_v}$; this is because s_v is only supported on inputs x for which $o_v = C_{i_v}$ is feasible (i.e., $C_{i_v}(x) = 0$ and $p_{i_v}(x) = 1$). Thus we may factor each s_v as $q_v p_{i_v}$ for some q_v. Hence we have our refutation, $\sum_{v\in V} q_v p_{i_v} = 1$. □

PPA–decision trees from NS refutations. Here is the converse.

Lemma 13. $\mathsf{PPA}^{\mathsf{dt}}(S(F)) \le O(\mathrm{NS}_{\mathbb{F}_2}(F))$ *for any unsatisfiable k-CNF formula F.*

Proof. Suppose $F := \bigwedge_{i\in[m]} C_i$, and let p_i be the natural polynomial encoding C_i. Let $\sum_{i\in[m]} q_i p_i = 1$ be a degree-d $\mathbb{F}_2$-Nullstellensatz refutation of F for $d := \mathrm{NS}_{\mathbb{F}_2}(F)$.

We define a cost-d PPA–decision tree $\mathcal{T} := (V, v^*, o_v, \mathcal{T}_v)$ solving $S(F)$. The vertices V will be grouped into $m+1$ groups $V^*, V_1, V_2, \ldots, V_m$. The group V^* will contain only the distinguished vertex v^*, which we think of as associated with the constant-1 term on the RHS of the refutation (the label o_{v^*} is arbitrary). For group V_i, consider expanding the polynomial $q_i p_i$ into a sum of monomials (which we may assume are pair-wise distinct and multilinear). The group V_i will contain one vertex v_m associated with each monomial m appearing in the expansion of $q_i p_i$. Moreover, each $v \in V_i$ will have $o_v := C_i$ as its associated solution.

Let us describe the edges of G_x and how to compute them. Each vertex will have at most one neighbor outside its group and at most one inside its group.

- *Out-group.* Since the polynomials $q_i p_i$ come from an $\mathbb{F}_2$-Nullstellensatz refutation, it follows that each monomial will occur an even number of times globally in the construction of V. Thus we can fix some global perfect matching M of V where only vertices which correspond to the same monomial are matched. For instance, if a monomial m occurs in the expansions of $q_i p_i$ and $q_j p_j$, the vertices $u_m \in V_i$ and $v_m \in V_j$ corresponding to m are allowed to be matched. In particular, the constant-1 term of $v^* \in V^*$ will be matched with some other constant-1 term in another group. For each edge $e \in M$ corresponding to an m, we add e to G_x iff $m(x) = 1$.
- *In-group.* Consider group V_i. If $p_i(x) = 1$ (i.e., $C_i(x) = 0$), then we will not add any edges inside V_i. If $p_i(x) = 0$, we will add many edges: First let $\rho := x \restriction \mathrm{vars}(p_i) \in \{0, 1, *\}^n$ be the partial assignment obtained by restricting x to the variables of p_i (at most k many). Consider the multiset of non-zero monomials T_ρ obtained by applying ρ to each monomial m in V_i and

including the resulting monomial $m' \coloneqq m(\rho)$ in T_ρ iff $m' \neq 0$. (This is truly a multiset, e.g., monomials x_1x_2 and x_1x_3 both reduce to x_1 under the partial assignment $x_2 = x_3 = 1$.) Since $p_i(\rho) = 0$, we of course have $q_ip_i(\rho) = 0$, and so it must be the case that each $m' \in T_\rho$ occurs an even number of times in T_ρ. With this in mind, we can fix a matching M_ρ between the vertices of V_i corresponding to like terms in T_ρ. For each edge $e \in M_\rho$ corresponding to an $m' \coloneqq m(\rho)$, we add e to G_x iff $m(x) = 1$ $(= m'(x))$.

The edges incident to an $v_m \in V_i$ can be determined by querying the variables of p_i (which defines ρ and hence M_ρ) and m. Hence the graph G_x can be described by depth-d decision trees $\mathcal{T}_v$. Let us finally check the correctness requirements. As in the proof of Lemma 11, we may check the simpler conditions *(C1')* and *(C2')*.

$v^* \in V^*$: Observe that v^* has always degree 1: it has a fixed out-group neighbor determined by M (independent of x) and no in-group neighbors. Hence *(C1')* is trivially satisfied.

$v_m \in V_i$: If $p_i(x) = 1$ (i.e., $C_i(x) = 0$ and $o_{v_m} = C_i$ is a feasible solution for x), then *(C2')* is trivially satisfied. So suppose $p_i(x) = 0$ (i.e., $C_i(x) = 1$ and $o_{v_m} = C_i$ is not a feasible solution). We show that $\deg(v_m)$ is even, which will verify *(C2')*. Let $\rho \coloneqq x \restriction \mathrm{vars}(p_i)$ and $m' \coloneqq m(\rho)$. If $m(x) = 0$, then $\deg(v_2) = 0$ is even. If $m(x) = 1$, then m' is non-zero. In this case v_m will have both an out- and in-group neighbor so that $\deg(v_m) = 2$ is even.

□

A Appendix: TFNP Class Definitions

For each TFNP subclass there is a canonical definition of its communication or query analogue: we simply let communication protocols or decision trees (rather than circuits) implicitly define the objects that appear in the original Turing machine definition. Each communication class C^{cc} (resp. query class C^{dt}) is defined via a C-*protocol* (resp. C–*decision tree*) that solves a two-party search problem $S \subseteq \{0,1\}^{n/2} \times \{0,1\}^{n/2} \times \mathcal{O}$ (resp. $S \subseteq \{0,1\}^n \times \mathcal{O}$). The class C^{cc} (resp. C^{dt}) is then defined as the set of all n-bit search problems S that admit a polylog(n)-cost C-protocol (resp. C–decision tree). We only define the communication analogues below with the understanding that a query version can be obtained by replacing mentions of a protocol $\Pi_v(x,y)$ by a decision tree $\mathcal{T}_v(x)$; the *cost* of a C–decision tree is defined as $\max_{v,x} \#(\text{queries made by } \mathcal{T}_v(x))$. In what follows, *sink* means out-degree 0, and *source* means in-degree 0.

Definition 1. (FP)

Syntax: Π is a (deterministic) protocol outputting values in $\mathcal{O}$.
Object: n/a
Correctness: $\Pi(x,y) \in S(x,y)$.
Cost: $|\Pi| \coloneqq$ communication cost of Π.

Definition 2. (EoML)

Syntax: V is a vertex set with a distinguished vertex $v^* \in V$. For each $v \in V$: $o_v \in \mathcal{O}$ and Π_v is a protocol outputting a tuple $(s_v(x,y), p_v(x,y), \ell_v(x,y)) \in V \times V \times \mathbb{Z}$.
Object: Dag $G_{x,y} = (V,E)$ where $(v,u) \in E$ iff $s_v(x,y) = u$, $p_u(x,y) = v$, $\ell_v(x,y) > \ell_u(x,y)$.
Correctness: If v^* is a sink or non-source in $G_{x,y}$, then $o_{v^*} \in S(x,y)$.
If $v \neq v^*$ is a sink or source in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 3. (SoML)

Syntax: Same as in Definition 2.
Object: Same as in Definition 2.
Correctness: If v^* is a sink or non-source in $G_{x,y}$, then $o_{v^*} \in S(x,y)$.
If $v \neq v^*$ is a sink in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 4. (PLS)

Syntax: V is a vertex set. For each $v \in V$: $o_v \in \mathcal{O}$ and Π_v is a protocol outputting a pair $(s_v(x,y), \ell_v(x,y)) \in V \times \mathbb{Z}$.
Object: Dag $G_{x,y} = (V,E)$ where $(v,u) \in E$ iff $s_v(x,y) = u$ and $\ell_v(x,y) > \ell_u(x,y)$.
Correctness: If v is a sink in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 5. (PPAD)

Syntax: V is a vertex set with a distinguished vertex $v^* \in V$. For each $v \in V$: $o_v \in \mathcal{O}$ and Π_v is a protocol outputting a pair $(s_v(x,y), p_v(x,y)) \in V \times V$.
Object: Digraph $G_{x,y} = (V,E)$ where $(v,u) \in E$ iff $s_v(x,y) = u$ and $p_u(x,y) = v$.
Correctness: If v^* is a sink or non-source in $G_{x,y}$, then $o_{v^*} \in S(x,y)$.
If $v \neq v^*$ is a sink or source in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 6. (PPADS)

Syntax: Same as in Definition 5.
Object: Same as in Definition 5.
Correctness: If v^* is a sink or non-source in $G_{x,y}$, then $o_{v^*} \in S(x,y)$.
If $v \neq v^*$ is a sink in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 7. (PPA)

Syntax: V is a vertex set with a distinguished vertex $v^* \in V$. For each $v \in V$: $o_v \in \mathcal{O}$ and Π_v is a protocol outputting a subset $\Pi_v(x,y) \subseteq V$ of size at most 2.
Object: Undirected graph $G_{x,y} = (V,E)$ where $\{v,u\} \in E$ iff $v \in \Pi_u(x,y)$ and $u \in \Pi_v(x,y)$.
Correctness: If v^* has degree $\neq 1$ in $G_{x,y}$, then $o_{v^*} \in S(x,y)$.
If $v \neq v^*$ has degree $\neq 2$ in $G_{x,y}$, then $o_v \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Definition 8. (PPP)

Syntax: V is a vertex set with a distinguished vertex $v^* \in V$. For each unordered pair $\{v,u\} \in \binom{V}{2}$: $o_{\{v,u\}} \in \mathcal{O}$. For each $v \in V$: Π_v is a protocol outputting values in $V - v^*$.
Object: Bipartite graph $G_{x,y} = (V \times (V - v^*), E)$ where $(v,w) \in E$ iff $\Pi_v(x,y) = w$.
Correctness: If (v,w) and (u,w), $v \neq u$, are edges in $G_{x,y}$, then $o_{\{v,u\}} \in S(x,y)$.
Cost: $\log|V| + \max_v |\Pi_v|$.

Acknowledgements

We thank Ankit Garg (who declined a co-authorship) for extensive discussions about monotone circuits. M.G. was supported by the Michael O. Rabin Postdoctoral Fellowship. P.K. was supported in parts by NSF grants CCF-1650733, CCF-1733808, and IIS-1741137. R.R. was supported by NSERC.

References

[ABRW04] Michael Alekhnovich, Eli Ben-Sasson, Alexander Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004. doi:10.1137/S0097539701389944.

[BCE+98] Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi. The relative complexity of NP search problems. *Journal of Computer and System Sciences*, 57(1):3–19, 1998. doi:10.1006/jcss.1998.1575.

[BDN18] Yakov Babichenko, Shahar Dobzinski, and Noam Nisan. The communication complexity of local search. Technical report, arXiv, 2018. arXiv:1804.02676.

[BFS86] László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986. doi:10.1109/SFCS.1986.15.

[BGIP01] Sam Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, 2001. doi:10.1006/jcss.2000.1726.

[BGW99] László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999. doi:10.1007/s004930050058.

[BIQ+17] Aleksandrs Belovs, Gábor Ivanyos, Youming Qiao, Miklos Santha, and Siyi Yang. On the polynomial parity argument complexity of the combinatorial Nullstellensatz. In *Proceedings of the 32nd Computational Complexity Conference (CCC)*, volume 79, pages 30:1–30:24. Schloss Dagstuhl, 2017. doi:10.4230/LIPIcs.CCC.2017.30.

[BM04] Joshua Buresh-Oppenheim and Tsuyoshi Morioka. Relativized NP search problems and propositional proof systems. In *Proceedings of the 19th Conference on Computational Complexity (CCC)*, pages 54–67, 2004. doi:10.1109/CCC.2004.1313795.

[BPS07] Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007. doi:10.1137/060654645.

[BR98] Paul Beame and Søren Riis. More on the relative strength of counting principles. In *Proceedings of the DIMACS Workshop on Proof Complexity and Feasible Arithmetics*, volume 39, pages 13–35, 1998.

[BR17] Yakov Babichenko and Aviad Rubinstein. Communication complexity of approximate Nash equilibria. In *Proceedings of the 49th Symposium on Theory of Computing (STOC)*, pages 878–889. ACM, 2017. doi:10.1145/3055399.3055407.

[Bul17] Andrei Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

[BW01] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001. doi:10.1145/375827.375835.

[CFL14] Stephen Cook, Yuval Filmus, and Dai Tri Man Lê. The complexity of the comparator circuit value problem. *ACM Transactions on Computation Theory*, 6(4):15:1–15:44, 2014. doi:10.1145/2635822.

[Cha13] Siu Man Chan. Just a pebble game. In *Proceedings of the 28th Conference on Computational Complexity (CCC)*, pages 133–143, 2013. doi:10.1109/CCC.2013.22.

[CP14] Siu Man Chan and Aaron Potechin. Tight bounds for monotone switching networks via fourier analysis. *Theory of Computing*, 10(15):389–419, 2014. doi:10.4086/toc.2014.v010a015.

[DP11] Constantinos Daskalakis and Christos Papadimitriou. Continuous local search. In *Proceedings of the 22nd Symposium on Discrete Algorithms (SODA)*, pages 790–804. SIAM, 2011. URL: http://dl.acm.org/citation.cfm?id=2133036.2133098.

[dRNV16] Susanna de Rezende, Jakob Nordström, and Marc Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 295–304. IEEE, 2016. doi:10.1109/FOCS.2016.40.

[FGMS18] John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. End of potential line. Technical report, arXiv, 2018. arXiv:1804.03450.

[FV98] Tomás Feder and Moshe Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.

[Gál01] Anna Gál. A characterization of span program size and improved lower bounds for monotone span programs. *Computational Complexity*, 10(4):277–296, 2001. doi:10.1007/s000370100001.

[GGKS18] Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 902–911. ACM, 2018. doi:10.1145/3188745.3188838.

[GP14] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Proceedings of the 46th Symposium on Theory of Computing (STOC)*, pages 847–856. ACM, 2014. doi:10.1145/2591796.2591838.

[GPW15] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1077–1088. IEEE, 2015. doi:10.1109/FOCS.2015.70.

[GPW16] Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 86:1–86:15. Schloss Dagstuhl, 2016. doi:10.4230/LIPIcs.ICALP.2016.86.

[GR18] Mika Göös and Aviad Rubinstein. Near-optimal communication lower bounds for approximate Nash equilibria. In *Proceedings of the 59th Symposium on Foundations of Computer Science (FOCS)*, 2018. To appear. arXiv:1805.06387.

[GS92] Michelangelo Grigni and Michael Sipser. Monotone complexity. In *Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 57–75. Cambridge University Press, 1992. URL: http://dl.acm.org/citation.cfm?id=167687.167706.

[GS18] Anat Ganor and Karthik C. S. Communication complexity of correlated equilibrium with small support. In *Proceedings of the 22nd International Conference on Randomization and Computation (RANDOM)*, volume 116, pages 12:1–12:16. Schloss Dagstuhl, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.12.

[HG18] Alexandros Hollender and Paul Goldberg. The complexity of multi-source variants of the End-of-Line problem, and the concise mutilated chessboard. Technical report, Electronic Colloquium on Computational Complexity (ECCC), 2018. URL: https://eccc.weizmann.ac.il/report/2018/120/.

[HN12] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time–space trade-offs in proof complexity. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 233–248. ACM, 2012. doi:10.1145/2213977.2214000.

[HR90] Bernd Halstenberg and Rüdiger Reischuk. Relations between communication complexity classes. *Journal of Computer and System Sciences*, 41(3):402–429, 1990. doi:10.1016/0022-0000(90)90027-I.

[HY17] Pavel Hubáček and Eylon Yogev. Hardness of continuous local search: Query complexity and cryptographic lower bounds. In *Proceedings of the 28th Symposium on Discrete Algorithms (SODA)*, pages 1352–1371, 2017. doi:10.1137/1.9781611974782.88.

[IPU94] Russell Impagliazzo, Toniann Pitassi, and Alasdair Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *Proceedings of the 9th Symposium on Logic in Computer Science (LICS)*, pages 220–228. IEEE, 1994. doi:10.1109/LICS.1994.316069.

[JPY88] David Johnson, Christos Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988. doi:10.1016/0022-0000(88)90046-3.

[Juk12] Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27 of *Algorithms and Combinatorics*. Springer, 2012.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[KW88] Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. In *Proceedings of the 20th Symposium on Theory of Computing (STOC)*, pages 539–550. ACM, 1988. doi:10.1145/62212.62265.

[KW93] Mauricio Karchmer and Avi Wigderson. On span programs. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 102–111, 1993. doi:10.1109/SCT.1993.336536.

[Lov79] László Lovász. On determinants, matchings, and random algorithms. In *Proceedings of the 2nd Conference on Fundamentals of Computation Theory (FCT)*, pages 565–574, 1979.

[Mor05] Tsuyoshi Morioka. *Logical Approaches to the Complexity of Search Problems: Proof Complexity, Quantified Propositional Calculus, and Bounded Arithmetic*. PhD thesis, University of Toronto, 2005.

[MP91] Nimrod Megiddo and Christos Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991. doi:10.1016/0304-3975(91)90200-L.

[MS92] Ernst Mayr and Ashok Subramanian. The complexity of circuit value and network stability. *Journal of Computer and System Sciences*, 44(2):302–323, 1992. doi:10.1016/0022-0000(92)90024-D.

[Mul87] Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987. doi:10.1007/BF02579205.

[Oli15] Igor Oliveira. *Unconditional Lower Bounds in Complexity Theory*. PhD thesis, Columbia University, 2015. doi:10.7916/D8ZP45KT.

[Pap94] Christos Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994. doi:10.1016/S0022-0000(05)80063-7.

[Pot17] Aaron Potechin. Bounds on monotone switching networks for directed connectivity. *Journal of the ACM*, 64(4):29:1–29:48, 2017. doi:10.1145/3080520.

[PR17] Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *Proceedings of the 49th Symposium on Theory of Computing (STOC)*, pages 1246–1255. ACM, 2017. doi:10.1145/3055399.3055478.

[PR18] Toniann Pitassi and Robert Robere. Lifting Nullstellensatz to monotone span programs over any field. In *Proceedings of the 50th Symposium on Theory of Computing (STOC)*, pages 1207–1219. ACM, 2018. doi:10.1145/3188745.3188914.

[PS98] Pavel Pudlák and Jiří Sgall. Algebraic models of computation and interpolation for algebraic proof systems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 39:279–295, 1998. doi:10.1090/dimacs/039.

[Pud10] Pavel Pudlák. On extracting computations from propositional proofs (a survey). In *Proceedings of the 30th Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8, pages 30–41. Schloss Dagstuhl, 2010. doi:10.4230/LIPIcs.FSTTCS.2010.30.

[Raz85a] Alexander Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical notes of the Academy of Sciences of the USSR*, 37(6):485–493, 1985. doi:10.1007/BF01157687.

[Raz85b] Alexander Razborov. Lower bounds on the monotone complexity of some Boolean functions. *Doklady Akademii Nauk USSR*, 285:798–801, 1985.

[Raz95] Alexander Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya of the RAN*, pages 201–224, 1995.

[RM99] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999. doi:10.1007/s004930050062.

[Rou18] Tim Roughgarden. Complexity theory, game theory, and economics. Technical report, arXiv, 2018. arXiv:1801.00734.

[RPRC16] Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen Cook. Exponential lower bounds for monotone span programs. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 406–415. IEEE, 2016. doi:10.1109/FOCS.2016.51.

[RW16] Tim Roughgarden and Omri Weinstein. On the communication complexity of approximate fixed points. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 229–238. IEEE, 2016. doi:10.1109/FOCS.2016.32.

[Sch78] Thomas Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Symposium on Theory of Computing (STOC)*, pages 216–226. ACM, 1978. doi:10.1145/800133.804350.

[Sok17] Dmitry Sokolov. Dag-like communication and its applications. In *Proceedings of the 12th Computer Science Symposium in Russia (CSR)*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9_26.

[Tar88] Éva Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988. doi:10.1007/BF02122563.

[Urq87] Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, 1987. doi:10.1145/7531.8928.

[Zhu17] Dmitriy Zhuk. A proof of CSP dichotomy conjecture. In *Proceedings of the 58th Symposium on Foundations of Computer Science (FOCS)*, pages 331–342, 2017. doi:10.1109/FOCS.2017.38.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il