Electronic Colloquium on Computational Complexity, Report No. 134 (2025)

$\mathsf{AC}^0[p]$-Frege Cannot Efficiently Prove that Constant-Depth Algebraic Circuit Lower Bounds are Hard

Jiaqi Lu[*] Imperial College London

Rahul Santhanam[†] University of Oxford

Iddo Tzameret[‡] Imperial College London

September 19, 2025

Abstract

We study whether lower bounds against constant-depth algebraic circuits computing the Permanent over finite fields (Limaye–Srinivasan–Tavenas [J. ACM, 2025] and Forbes [CCC'24]) are hard to prove in certain proof systems. We focus on a DNF formula that expresses that such lower bounds are hard for constant-depth algebraic proofs. Using an adaptation of the diagonalization framework of Santhanam and Tzameret (SIAM J. Comput., 2025), we show *unconditionally* that this family of DNF formulas does not admit polynomial-size propositional $\mathsf{AC}^0[p]$-Frege proofs, infinitely often. This rules out the possibility that the DNF family is easy, and establishes that its status is either that of a hard tautology for $\mathsf{AC}^0[p]$-Frege or else unprovable (i.e., not a tautology). While it remains open whether the DNFs in question are tautologies, we provide evidence in this direction. In particular, under the plausible assumption that certain (weak) properties of multilinear algebra—specifically, those involving tensor rank—do not admit short constant-depth algebraic proofs, the DNFs *are* tautologies. We also observe that several weaker variants of the DNF formula are provably tautologies, and we show that the question of whether the DNFs are tautologies connects to conjectures of Razborov (ICALP'96) and Krajíček (J. Symb. Log., 2004).

Additionally, our result has the following special features:

(**i**) **Existential depth amplification**: the DNF formula considered is parameterised by a constant depth d bounding the depth of the algebraic proofs. We show that there *exists some fixed* depth d such that if there are no small depth-d algebraic proofs of certain circuit lower bounds for the Permanent, then there are no such small algebraic proofs in *any* constant depth.

(**ii**) **Necessity**: We show that our result is a necessary step towards establishing lower bounds against constant-depth algebraic proofs, and more generally against any sufficiently strong proof system. In particular, showing there are no short proofs for our DNF formulas, obtained by replacing 'constant-depth algebraic circuits' with any "reasonable" algebraic circuit class $\mathcal{C}$, is *necessary* in order to prove any super-polynomial lower bounds against algebraic proofs operating with circuits from $\mathcal{C}$.

[*]Department of Computing. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 101002742, *EPRICOT* project).

[†]This project was partly funded by the EPSRC grant EP/Z534158/1, *Integrated Approach to Computational Complexity: Structure, Self-Reference and Lower Bounds.*

[‡]Department of Computing. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 101002742, *EPRICOT* project). It was also supported by the Engineering and Physical Sciences Research Council (EPSRC) under grant EP/Z534158/1, *Integrated Approach to Computational Complexity: Structure, Self-Reference and Lower Bounds.*. Email: iddo.tzameret@gmail.com

ISSN 1433-8092

1 Introduction

Propositional proof complexity studies the sizes of proofs for propositional tautologies in *propositional proof systems* of interest [CR79, BP98, Kra19]. In general, a propositional proof system Q is defined by a polynomial-time computable binary relation R_Q such that a formula ϕ is a propositional tautology if and only if there is some y such that $R_Q(\phi, y)$ holds; any such y is called a *proof* of ϕ. The R-proof size of ϕ is the size of the smallest y such that $R_Q(\phi, y)$ holds. For natural sequences of tautologies (such as the Pigeonhole Principle, Tseitin graph formulas, and formulas encoding circuit lower bounds) ϕ_n and propositional proof systems of interest (such as Resolution, Frege and Extended Frege), we would like to understand how the proof size of ϕ_n grows with the size of the formula $|\phi_n|$, and in particular whether the proof size is polynomially bounded or not as a function of the size of the formula.

In their seminal paper on propositional proof complexity, Cook and Reckhow [CR79] observed that $\mathsf{NP} = \mathsf{coNP}$ if and only if there is a propositional proof system in which every sequence of tautologies has polynomially bounded proof size. This is the basis of the "*Cook-Reckhow program*" [BP98], of which the ideal limit is the separation of NP from coNP (and hence also P from NP) by showing super-polynomial proof size lower bounds for progressively stronger propositional proof systems.

The Cook-Reckhow program can be seen as a dual, nondeterministic, analogue of the *circuit complexity approach* to P vs NP, which proceeds by showing super-polynomial circuit size lower bounds for progressively stronger circuit classes. This analogy is further strengthened by the fact that there is a close correspondence between Boolean circuit classes and inference-based propositional logic, i.e., Frege-style proof systems [BP98], in which new proofs are derived from axioms and previously derived proof lines using simple sound derivation rules. The basic Resolution proof system works with proof lines that are clauses; the Frege proof system with lines that are Boolean formulas; and the Extended Frege (EF) proof system with lines that are essentially Boolean circuits. The strength of these Frege-style proof systems is generally believed to grow as the proof lines get more expressive, just as the corresponding circuit classes are believed to increase in computational power as they grow more expressive.

1.1 Hard Formulas

We say that a sequence of formulas ϕ_n is *hard* if there is no proof of ϕ_n of polynomial size in $|\phi_n|$. One of the earliest steps in the Cook-Reckhow program—establishing a lower bound on the size of proofs—was taken by Haken [Hak85]. He showed a super-polynomial lower bound for the Pigeonhole Principle formulas in Resolution. Soon after, Ajtai [Ajt88] showed a super-polynomial lower bound for the Pigeonhole Principle in AC^0-Frege, which is the Frege-style system where proof lines are constant-depth Boolean circuits. Since then, several improvements and extensions of Ajtai's result have been obtained [Ajt94, BIK$^+$96a, BIK$^+$96b], and the lower bound techniques used in these works mirror the random restriction techniques used to prove lower bounds against the circuit class AC^0, further reinforcing the analogy between proof complexity lower bounds and circuit complexity lower bounds (cf. [PBI93, KPW95]). In the circuit complexity setting, we also know lower bounds for the circuit class $\mathsf{AC}^0[p]$ of constant-depth Boolean circuits with prime modular gates, shown using the polynomial method of Razborov and Smolensky [Raz87, Smo87]. Can the polynomial method or other techniques used to show proof complexity lower bounds for the corresponding Frege-style proof system $\mathsf{AC}^0[p]$-Frege?

Despite much effort, this question has remained open for more than three decades, and continues to be a frontier question in proof complexity. It is already highlighted as a key open problem in the 1998 survey of Beame and Pitassi [BP98], and the recent survey of Razborov re-iterates this [Raz16b]. Progress on the question has focused on restricted *algebraic* subsystems of $\mathsf{AC}^0[p]$-Frege such as the

Nullstellensatz [BIK$^+$96a] and Polynomial Calculus [CEI96] proof systems, which we discussed next.

1.2 Related Work on Algebraic Proof Systems

Much of the research on $\mathsf{AC}^0[p]$-Frege lower bounds has focused on subsystems such as Nullstellensatz and Polynomial Calculus, which encode a CNF and the Boolean constraints on its variables as polynomials and reduce the task of proving that the CNF has no satisfying Boolean assignments[1] to proving that these polynomials do not have a common zero [BIK$^+$96a, BIK$^+$96b]. Nullstellensatz and Polynomial Calculus are propositional proof systems in the Cook-Reckhow sense [CR79], since the verification of proofs can be done in deterministic polynomial time.

Pitassi [Pit97] proposed more powerful algebraic proof systems where verifying the correctness of a proof requires identity testing of algebraic circuits, i.e., checking whether a given algebraic circuit is identically zero. Identity testing is known to be doable in randomized polynomial time, but it remains a long-standing open question whether it can be done in deterministic polynomial time for general algebraic circuits. Thus, the more general algebraic systems proposed by Pitassi are not propositional proof systems in the traditional Cook-Reckhow sense, but they do have efficient *randomized* verification.

Grochow and Pitassi [GP18] defined a strong algebraic proof system in this sense, called the *Ideal Proof System* (IPS), where a single algebraic circuit acts as a *certificate* that a set of polynomial equations does not have a common zero. The size of an IPS proof is simply the size of the corresponding algebraic circuit which acts as a certificate. [GP18] showed that IPS is at least as strong as EF, and also that super-polynomial IPS lower bounds for *any* sequence of unsatisfiable formulas implies that the Permanent does not have polynomial-size algebraic circuits. This gives a long sought-after connection between proof complexity lower bounds for strong proof systems and (algebraic) circuit complexity lower bounds, but for an algebraic proof system with randomized verification rather than for a propositional proof system. In a more recent paper [ST25], the implication from proof complexity lower bounds to algebraic complexity lower bounds was strengthened to an *equivalence* for a certain explicit sequence of formulas.

Given that a proof in IPS is a single algebraic circuit, we can define and study variants of IPS where this circuit is of a restricted form. This has been done in several works in the past decade [FSTW21, AF22, GHT22, HLT24], which seek to show proof complexity lower bounds for subsystems of IPS or to find closer connections between IPS variants and propositional proof systems. Proof complexity lower bounds in this setting are typically shown by adapting algebraic circuit lower bound techniques. One apparent drawback to this approach is that the hard candidates in these works are not propositional formulas, but rather purely algebraic instances (e.g., $x_1 + \cdots + x_n + 1 = 0$) that cannot be directly translated to propositional logic (cf. [EGLT25] for a discussion of this point).

In 2021, a breakthrough super-polynomial algebraic circuit lower bound against constant-depth algebraic circuits was shown by Limaye, Srinivasan and Tavenas [LST25] for large enough fields, and very recently has been extended by Forbes [For24] to fields of characteristic p for any prime p. This motivates the question of whether a similar super-polynomial *proof size* lower bound holds for constant-depth IPS, where the certificate is a constant-depth algebraic circuit. Some progress on this question has been made in recent work [AF22, GHT22, HLT24], but it remains open whether there are unsatisfiable CNF formulas (or propositional logic formulas more generally) requiring super-polynomial constant-depth IPS proofs.

[1] We will sometimes switch back and forth in our discussion between the tasks of refuting that a CNF formula is satisfiable and of proving that a DNF formula is a tautology, which are equivalent by De Morgan's laws.

1.3 Framework and Results

While most lower bounds in propositional proof complexity for concrete tautologies rely on combinatorial or algebraic techniques, one can also try to use logic—specifically, diagonalization—for this purpose. This is a natural idea, but in the propositional setting it faces an inherent obstacle: self-reference. Suppose that a formula Φ expresses a proof-complexity lower bound. Ideally, one would like Φ to encode the statement that "Φ itself has no short proofs." This, however, is impossible: any reasonable propositional encoding of Φ must use at least $|\Phi|$ symbols, so Φ would necessarily be longer than itself.[2]

One way to circumvent this problem, due to Krajíček, is to encode proofs implicitly, and hence more economically, via a circuit that computes the bits of the proof given their index (see [Kra04a]).

Santhanam–Tzameret [ST25] proposed a different approach: instead of referring to itself directly, a formula refers to a smaller version of itself. Concretely, they introduced the *iterated lower bound formulas*, which encode inductively the statement that "the previous-level formula has no short proof," thus avoiding direct self-reference. This yields diagonalization-based formulas that provably lack short proofs infinitely often. Specifically, if at level ℓ the formula

$$\varphi_\ell := \text{"there is no short proof of } \varphi_{\ell-1}\text{"}$$

has either a long proof or a short proof, then in both cases it establishes the truth of the no-short-proof claim for $\varphi_{\ell-1}$. If φ_ℓ has no proof at all, then in particular it has no short proof. Thus, we are done: either φ_ℓ or $\varphi_{\ell-1}$ has no short proof, and this holds for infinitely many ℓ.

By referring to a smaller (and therefore different) version of itself, the resulting formulas demonstrate that φ_ℓ has no short proofs infinitely often. However, this still leaves open whether these formulas are hard tautologies or not tautologies at all (and thus vacuously unprovable). Let us elaborate on the difference between showing that a statement is not easy, and showing that it is hard, i.e., that it is a tautology with no short proofs.

In general, given a proof system, every propositional formula φ falls into one of three categories: 1. a tautology with a short proof (easy), 2. a tautology without short proofs (hard), or 3. not a tautology (hence unprovable). This is depicted in Table 1.

Proof Complexity ↓ / Validity →	**Tautology**	**Non-tautology**
Easy	1. easy formula	**X**
Hard	2. hard formula	**X**
Unprovable	**X**	3. trivially unprovable

Table 1: A priori, every propositional formula falls into one of the cells labeled 1, 2, or 3.

The standard aim in proof complexity is to establish hardness, i.e., to identify a formula in cell 2. For strong proof systems this remains open. Thus, a natural intermediate step is *ruling out that a formula is easy* (cell 1), while leaving open whether it belongs to cell 2 or 3. For this to be meaningful, one must consider sequences of formulas whose tautological status is unknown. A natural choice is formulas expressing open lower bounds in complexity theory. This viewpoint was emphasised by Razborov [Raz15], who highlighted the importance of studying the proof complexity of natural statements whose validity is unknown. This approach was pursued in the 1990s by Razborov and others [Raz95a, Raz95b, Raz98, Raz15] (see also Krajíček [Kra04b] and Raz [Raz04]).[3]

[2]Friedman [Fri79] and Pudlak [Pud86, Pud87] show how to adapt the proof of Gödel's Second Incompleteness Theorem to give *sub-linear* proof size lower bounds for strong enough propositional proof systems.

[3]Typically, these propositional formulas are known unconditionally to be tautologies for random objects. For example,

In this work, as in [ST25], we follow this approach. We show that a family of DNF formulas of unknown validity has no short proofs. This situation is illustrated in Table 2.

Proof Complexity ↓ / Validity →	**Tautology**	**Non-tautology**
Easy	1. **X**	**X**
Hard	2. hard formula	**X**
Unprovable	**X**	3. trivially unprovable

Table 2: The DNF formula under consideration in this paper is shown to lie in either cell 2 or cell 3; in particular, we *rule out* the possibility that it is in cell 1 (with respect to the proof system $\mathsf{AC}^0[p]$-Frege).

There is another reason to consider statements of unknown validity when proving lower bounds for strong systems: very few plausible hard candidates are known for Frege and Extended Frege. The only compelling ones are random CNFs and circuit lower bound formulas, and for neither do we have effective methods for certifying validity. Indeed, confirming the validity of a fixed formula ϕ typically requires a proof, and existing proof methods can usually be captured in polynomial-size Frege or Extended Frege proofs—implying that ϕ is not a plausible hardness candidate for these systems.

As mentioned above, iterated lower bound formulas provide a simple way to show that a sequence of formulas is not easy. However, unlike formulas asserting SAT $\notin \mathsf{P/poly}$, it is unclear whether there is any reason to believe they are tautologies. What [ST25] demonstrated is that assuming some circuit lower bounds one can go beyond the iterated lower bound formulas: rule out that some formulas are easy for formulas that are *more likely to be tautologies*—specifically, formulas expressing that proving algebraic *circuit* lower bounds is hard. In particular, [ST25] combined the diagonalization framework with the [GP18] reduction from proof complexity to circuit lower bounds that showed that a proof-size lower bound implies an algebraic circuit-size lower bound (VP $\neq$ VNP). Hence, instead of stating recursively that lower bounds are hard to prove, one formulates φ to be the statement "there are no short proofs of algebraic circuit lower bounds of the Permanent". This avoids the recursively defined statement of the iterated lower bound formulas, and is a statement whose validity is easier to assess and use. In this way we obtain:

$$\text{circuit lower bound} \quad \Rightarrow \quad \text{no short proof of } \overbrace{\text{`no short proof of the circuit lower bound'}}^{\varphi}.$$

However, this still leaves an extra conditional layer: we need to rely on $\mathsf{VP} \neq \mathsf{VNP}$ (the circuit lower bound) to rule out that φ is easy.

In the present work, we eliminate this conditionality in the constant-depth regime. Using the unconditional constant-depth algebraic circuit lower bound of Limaye, Srinivasan and Tavenas [LST25], we extend the diagonalization framework to constant-depth circuits.

We demonstrate formulas that *unconditionally* are not easy for $\mathsf{AC}^0[p]$-Frege, such that:

- **Plausibly tautologies:** we give some evidence supporting the validity of these formulas;
- **Necessary (complete):** any hard instance for a sufficiently strong proof system must imply that our formulas are also not easy for that system;

if a formula encodes a circuit lower bound for some Boolean function f, then for random f the lower bound holds. Thus, while the interesting candidate formula stating, say, SAT $\not\subseteq \mathsf{P/poly}$ is not known to be a tautology, for a random f the statement $f \notin \mathsf{P/poly}$ is.

- **Amplifying:** exhibiting a form of existential depth amplification: there exists a constant d, such that if our formulas are hard for depth-d proofs then they are hard for *every* constant-depth d' proofs.

Theorem 1.1 (Corollary of the more formal Theorem 1.2 below)**.** *For every prime p there is an explicit sequence $\{\phi_n\}$ of DNF formulas (of unknown validity) such that there are no polynomial-size $\mathsf{AC}^0[p]$-Frege proofs of $\{\phi_n\}$.*

As mentioned above, Theorem 1.1 is shown via a diagonalization argument applied to the algebraic proof system IPS. It exploits the lower bound in [LST25] and Forbes' finite fields version [For24], and the fact that Grochow-Pitassi [GP18] showed that constant-depth IPS over finite prime fields simulates $\mathsf{AC}^0[p]$-Frege.

The formulas ϕ_n express proof size lower bounds for algebraic proof systems, and the evidence for their validity comes from several sources.

Remark (Switching between DNF tautologies and unsatisfiable CNF formulas)**.** *Since we work with the* refutation *system IPS (whose constant-depth version simulates $\mathsf{AC}^0[p]$-Frege), it is more natural to consider CNF formulas that are conjectured to be unsatisfiable, rather than DNF formulas that are conjectured to be valid. This is a matter of convenience, because of the trivial equivalence between showing that a CNF is unsatisfiable and showing that its complementary DNF is valid. We sometimes abuse notation and use "refutations" and "proofs" interchangeably; in all cases, the exact meaning should be clear from the context.*

The CNF formulas we consider are related to the circuit lower bound formulas considered by Razborov [Raz95a, Raz95b, Raz98, Raz15], but with two differences: we consider *algebraic* rather than Boolean circuits, and our formulas express lower bounds for *proving* constant-depth algebraic lower bounds in constant-depth IPS, rather than express the circuit lower bounds directly. This allows us to adapt the diagonalization technique used to show an equivalence between circuit complexity and proof complexity in [ST25] to derive an *unconditional* proof complexity lower bound in our setting.

To state our result more precisely we need the following notation. Let $\mathbb{F}$ be some underlying (finite) field. We let:

- $\mathrm{ckt}_d(\mathrm{perm}_n, s)$: a CNF formula expressing that the Permanent on $n \times n$ matrices has depth-d algebraic circuits of size s over $\mathbb{F}$.
- $\text{ref-IPS}_d(\phi, t)$: a CNF formula expressing that ϕ has size-t IPS refutations of depth-d over $\mathbb{F}$.
- The ***diagonalizing CNF formula*** is:

$$\psi_{d,d',n} := \text{ref-IPS}_{d'}(\mathrm{ckt}_d(\mathrm{perm}_n, n^{O(1)}), N^{O(1)}),$$

  where $N = |\mathrm{ckt}_d(\mathrm{perm}_n, n^{O(1)})|$ is $O(2^{n^{O(1)}})$, expressing that depth-d' IPS refutes in polynomial-size that the permanent is computed by a depth-d algebraic circuit of polynomial-size.

The use of $O(1)$ in the notation above is informal. We use it to avoid using more quantifiers that would make the statement below hard to parse.

Theorem 1.2 (Informal Statement; Theorem 5.3)**.** *For every constant prime p and for all positive integers d, there is a positive integer d' such that for all positive integers d'', there are no polynomial-size depth-d'' IPS refutations of the formula $\psi_{d,d',n}$, for infinitely many n over $\mathbb{F}_p$.*

Theorem 1.2 states the existence of no polynomial constant-depth IPS refutations for formulas that themselves express constant-depth IPS short refutation of constant-depth circuit upper bounds. Here the size of a proof is always measured as a function of the length of the formula being proved. The result holds for any finite field, and further for sequence of prime fields of increasing size that are not too large, as we show below in Theorem 1.7.

Theorem 1.1 follows from Theorem 1.2 as follows: i) take the same p as in the statement of Theorem 1.2; ii) let the formulas ϕ_n in the former result to be the negation of $\psi_{d,d',n}$ in the latter result[4], where d (the stated depth of circuit computing Permanent) is chosen to be large enough and d' (the stated depth of IPS refutations) is chosen as a function of d so that Theorem 1.2 holds; iii) finally, use the fact that constant-depth IPS over fields of characteristic p simulates $\mathsf{AC}^0[p]$-Frege [GP18].

1.3.1 Implications and Several Important Aspects of Theorem 1.2

Supporting evidence for the unsatisfiability of the diagonalizing CNF formula. We describe three forms of supporting evidence that $\psi_{d,d',n}$ is unsatisfiable.

(I) *Hardness of multilinear algebra for constant-depth proofs.* To establish the unsatisfiability of $\psi_{d,d',n}$, it suffices to show that constant-depth algebraic circuit upper bound formulas do not admit small-size constant-depth IPS refutations.

The tensor rank principle denoted $\mathrm{TRankP}^r_{m,n}(A)$ and introduced in [GGL+25], states that an order-r tensor A of rank m can be written as the sum of n rank-1 tensors of order r. This principle is easily shown to be unsatisfiable when $m > n$. In [GGL+25] the tensor rank principle was reduced to *constant-depth* algebraic circuit upper bound formulas in constant-depth IPS. As a result, proving super-polynomial-size lower bounds for the tensor rank principle against constant-depth IPS implies the unsatisfiability of the diagonalizing CNF formula:

Corollary 1.3 ([GGL+25]; informal, see Corollary 6.6)**.** *If the sequence of CNF formulas $\psi_{d,d',n}$ are satisfiable then the tensor rank principle $\mathrm{TRankP}^r_{m,n}(A)$ admits polynomial-size refutations in depth-$O(d')$ IPS.*

It is known from [LST25, For24] that the determinant cannot be computed by polynomial-size constant-depth algebraic circuits over any field. Consequently, following the informal alignment between proof complexity and circuit complexity, one expects that proof systems operating with constant-depth algebraic circuits cannot efficiently prove statements expressing linear (or multilinear) algebraic properties (whose standard proofs use notions like rank and determinants). Therefore, it is reasonable to expect that $\mathrm{TRankP}^r_{m,n}(A)$ does not admit polynomial-size refutations in constant-depth IPS, and hence, that $\psi_{d,d',n}$ is unsatisfiable by the corollary.

(II) *Weaker versions of the CNF that are provably unsatisfiable.* To establish the unsatisfiability of $\psi_{d,d',n}$, it suffices to show that constant-depth algebraic circuit upper bound formulas do not admit small-size constant-depth IPS refutations. Here we mention two recent results that establish this for weaker variants of $\psi_{d,d',n} = \text{ref-IPS}_{d'}(\text{ckt}_d(\text{perm}_n, n^{O(1)}), N^{O(1)})$, namely when the proof system is Polynomial Calculus with Resolution (PCR) instead of depth-d' IPS (denoted "$\text{IPS}_{d'}$" in $\psi_{d,d',n}$) and when the lower bound statement is against either algebraic circuits of unrestricted depth instead of general algebraic circuits (denoted "ckt_d" in $\psi_{d,d',n}$), or against noncommutative algebraic branching programs.

[4]We take negation because we consider $\mathsf{AC}^0[p]$-Frege to be a *proof system* for DNF tautologies, while constant-depth IPS is a *refutation system* for unsatisfiable CNFs.

The refutation system PCR can be considered roughly as depth-2 IPS (see [GP18]). Note that when we increase the strength of the algebraic circuit model replacing ckt_d in $\psi_{d,d',n}$, we are actually *weakening* the statement, since proving lower bounds against stronger circuit model is *harder*, meaning that it is *easier* to show that such lower bounds are harder.

Corollary 1.4 ([GGL+25]; Theorem 6.11)**.** *Let f be any polynomial in n variables over $\mathbb{F}_2$. The CNF formula* $\text{ref-PCR}(\mathrm{ckt}(f, n^{O(1)}), N^{O(1)})$*, stating that* PCR *over $\mathbb{F}_2$ has a polynomial-size refutation of the statement that f is computable by polynomial-size algebraic circuits, is unsatisfiable.*

When the proof system in $\psi_{d,d',n}$ is weakened again to PCR instead of depth-d' IPS, while the algebraic circuit model is very weak (which *strengthens* the proof complexity lower bound statement), namely, noncommutative algebraic branching program (denoted ncABP), we have the following:

Corollary 1.5 ([GGRT25])**.** *Let f be any noncommutative polynomial in n variables over $\mathbb{F}_2$. The CNF formula* $\text{ref-PCR}(\mathrm{ncABP}(f, n^{O(1)}), N^{O(1)})$*, stating that* PCR *over $\mathbb{F}_2$ has polynomial-size refutations of the statement that f is computable by polynomial-size noncommutative algebraic branching programs, is unsatisfiable.*

Corollary 1.5 is proved via a reduction similar to the reduction from iterated proof complexity generators to Boolean circuit upper bound formulas in [Raz15]. Since an algebraic branching program is characterized by iterated matrix multiplication, one can reduce the iterated rank principle to the ncABP upper bound formulas [GGRT25]. Thus, Corollary 1.5 follows by an exponential lower bound for the iterated rank principle.

(III)*Algebraic analogues of proof complexity conjectures*: Krajíček [Kra04b] and Razborov [Raz96, Raz16b] have conjectured the following:[5] Extended Frege cannot efficiently prove *any* super-polynomial Boolean circuit lower bound. Since Extended Frege is essentially a proof system that operates with Boolean circuits, this conjecture says that proof systems operating with Boolean circuits cannot efficiently prove Boolean circuit lower bounds. We raise the following analogous conjecture:

> **Constant-depth algebraic analogue of Krajíček-Razborov conjecture:** For every d, there is a d' such that there are no polynomial-size depth-d' IPS proofs of super-polynomial depth-d lower bounds for *any* polynomial f.

(Of course, the conjectured statement above depends on a natural encoding or formulation of the lower bound statement.)

The constant-depth algebraic analogue of the Krajíček-Razborov conjecture implies the unsatisfiability of the CNF formulas $\psi_{d,d',n}$ for arbitrary d and large enough d'. The formulas $\psi_{d,d',n}$ assert this only for $f = \mathrm{perm}$, and therefore follow trivially from the conjecture.

Diagonalizing formulas are necessary for lower bounds. We show that the diagonalizing formulas are not easy is *logically necessary* in order to prove super-polynomial lower bounds for $\mathcal{C}$-IPS for *any* "reasonable" algebraic circuit class $\mathcal{C}$. In other words, we show that the non-easiness of the diagonalizing formulas is implied by *any* super-polynomial lower bounds on tautologies for algebraic proofs operating with circuits from $\mathcal{C}$. To show this, we use the circuits-to-proofs connection of [GP18].

[5]Razborov conjectured in [Raz15] that Frege cannot efficiently prove super-polynomial circuit lower bounds for any Boolean function. More specifically, [Raz15, Conjecture 1] with suitable parameters for the underlying combinatorial designs implies under some hardness assumptions that Frege cannot efficiently prove that SAT$\not\subseteq$ P/poly. Further conjectures about the impossibility of *Extended* Frege to efficiently prove circuit lower bounds have been circulated in the proof complexity literature and discussions (cf. [Raz16a, Raz21, Kra11]).

The notation $\mathcal{C}$-IPS stands for the IPS proof system in which an IPS refutation (i.e., certificate) is written as an algebraic circuit from the class $\mathcal{C}$ (for instance, depth-d circuits, for a constant d, algebraic formulas, etc.). A "reasonable" algebraic circuit class $\mathcal{C}$ is one for which the Grochow-Pitassi implication from $\mathcal{C}$-IPS lower bounds to $\mathcal{C}$ circuit lower bounds holds, and moreover this implication is efficiently provable in $\mathcal{C}$-IPS. Our methods in this paper imply that all the commonly studied algebraic circuit classes which contain the class of constant-depth algebraic circuits are reasonable.

Theorem 1.6 (Informal; Theorem 5.10; If algebraic proofs are not p-bounded then it is not easy to prove that circuit lower bounds are hard for algebraic proofs.)**.** *Let $\mathcal{C}$ be any "reasonable" algebraic circuit class. If there is a sequence $\{\phi_n\}_n$ of unsatisfiable CNF formulas that requires super-polynomial size $\mathcal{C}$-*IPS *proofs for infinitely many n, then the sequence $\{\psi_n\}_n$ of CNF formulas does not have polynomial size $\mathcal{C}$-*IPS *proofs for infinitely many n, where $\psi_n = \text{ref-}\mathcal{C}\text{-IPS}(\mathcal{C}\text{-ckt}(\text{perm}_n, n^{O(1)}), N^{O(1)})$, with $N = |\mathcal{C}\text{-ckt}(\text{perm}_n, n^{O(1)})|$.*

Theorem 1.6 is shown by abstracting the argument of Theorem 1.2 and combining the resulting generalization with the Grochow-Pitassi implication from proof complexity lower bounds to circuit complexity lower bounds [GP18].

Existential depth amplification. The statements $\psi_{d,d',n}$ themselves refer to proof complexity lower bounds. Nevertheless, the lower bounds stated in the formulas and the lower bounds we get from our result are *different.*

First, the formulas state hardness of proving circuit lower bounds, while we get hardness of proving *proof complexity lower bounds.*

Second, notice the quantification over depths in Theorem 1.2: there is some *fixed* depth d' such that if depth-d' IPS lower bounds on certain circuit lower bounds for the Permanent hold, then we get super-polynomial lower bounds for proofs of *any* constant depth. This shows that we can escalate the depth-d' IPS lower bounds stated in the formulas to *any* constant-depth IPS lower bounds. In other words, our result does not merely repeat the stated lower bound we assumed against depth-d' proofs of circuit lower bounds, but goes beyond it to rule out short proofs in *any* depth of proof complexity lower bounds.

1.3.2 Extension to Larger Fields

While [ST25] and Theorem 1.2 crucially use fields of constant size, and the ability to efficiently encode computation over finite fields of constant size by CNF formulas, we show further how to encode and reason about larger and growing fields. This technical contribution may be interesting on its own right.

Specifically, by reasoning about the bits of polynomial expressions with algebraic proofs, Theorem 1.2 can be extended to underlying fields of size polynomially bounded by $|\psi_{d,d',n}|$. Bit arithmetic in proof complexity was used before (cf. [Goe90, Bus87, AGHT24, IMP20]). We show how to reason about iterated addition, iterated multiplication and modular arithmetic in constant-depth IPS over polynomial-size fields.

Theorem 1.7 (Informal Statement; Corollary 7.23)**.** *Let $\{p_n\}$ be any sequence of primes such that $p_n = O(2^n)$, and $\mathbb{F}_{p_n}$ be the field of size p_n. For all positive integers d, there is a positive integer d' such that for all positive integers d'', there are no polynomial-size depth-d'' IPS refutations of the formula $\psi_{d,d',n} = \text{ref-IPS}_{d'}(\text{ckt}_d(\text{perm}_n, n^{O(1)}), N^{O(1)})$ for infinitely many n over $\mathbb{F}_{p_n}$, where $N = |\text{ckt}_d(\text{perm}_n, n^{O(1)})|$ is $O(2^{n^{O(1)}})$.*

1.3.3 The Diagonalization Argument

Here we explain informally the idea behind the proof of Theorem 1.2, showing that the diagonalizing CNFs have no short refutations. This is where most of the nontrivial technical work lies. The key idea is to combine diagonalization with the known implication from proof complexity lower bounds to circuit complexity lower bounds [GP18].

The argument builds on the following three nontrivial technical points:

1. There is a reasonable CNF encoding expressing that the permanent polynomial can be computed by bounded-depth small-size algebraic circuits "VNP = VAC^0" (this is $\text{ckt}_d(\text{perm}_n, n^{O(1)})$ from before).[6]
2. There is a reasonable CNF encoding of the statement that there are constant-depth IPS refutations of size s for a CNF ϕ (this is $\text{ref-IPS}_d(\phi, t)$ from before).
3. If ϕ is unsatisfiable and there are short and constant-depth IPS refutations of "constant-depth IPS efficiently refutates ϕ", then there are short and constant-depth IPS refutations of "VNP = VAC^0".

The work of [ST25] formalized the Grochow-Pitassi implication from IPS lower bounds for CNFs to VNP $\neq$ VP within IPS. Assumption 3 above is a novel formalisation of a *constant-depth* version of the Grochow-Pitassi implication within constant-depth IPS.

According to [LST25], the permanent polynomial cannot be computed by constant-depth small-size algebraic circuits, which means "VNP = VAC^0" is an unsatisfiable CNF, using Assumption 1. Assume for the sake of contradiction that $\psi_{d,d',n}$, namely, $\text{ref-IPS}_{d'}(\text{"VNP} = \mathsf{VAC}^0\text{"}, \mathsf{poly})$ (or more formally, $\text{ref-IPS}_{d'}(\text{ckt}_d(\text{perm}_n, n^{O(1)}), N^{O(1)})$) has polynomial-size refutations in constant-depth IPS. Then by Assumption 3, "VNP = VAC^0" has polynomial-size constant-depth IPS refutations. But this contradicts the soundness of IPS, since IPS has (polynomial-size) refutations of the statement that "VNP = VAC^0" has polynomial-size IPS-refutations (formally, one has to account for instances of different sizes when following this argument, and we explain this more precisely below).

To clarify further the argument, we describe a slightly more detailed overview, highlighting the logic behind the argument. We show that infinitely often there are no polynomial-size constant-depth IPS refutations of the formula $\text{ref-IPS}_{d'}(\text{ckt}_d(\text{perm}_n, n^c), n^{c'})$, expressing that there exists a constant c' such that $\text{ckt}_d(\text{perm}_n, n^c)$ has IPS refutations of size bounded from above by the polynomial $n^{c'}$ and depth bounded by d'.

Proof sketch of Theorem 1.2 (formally Theorem 5.3). Let i.o. abbreviate *infinitely often*, and let a.e. denote its converse *almost everywhere*, that is, "always except for finite many cases". Let $G \vdash^{f(n)}_{d} 1 = 0$ stands for an IPS refutation of G of refutation-size bounded from above by $f(n)$ and refutation-depth bounded from below by d. Our goal is to prove:

$$\forall\, c, d\ \exists\, c', d'\ \forall\, c'', d''\ \text{i.o.}\ \underbrace{\text{ref-IPS}_{d'}(\overbrace{\text{ckt}_d(\text{perm}_n, n^c)}^{\gamma}, |\gamma|^{c'})}_{\lambda} \nvdash^{|\lambda|^{c''}}_{d''} 1 = 0, \tag{1}$$

meaning that for all constants c, d there are constants c', d', such that for all constants c'', d'', λ does not have polynomial-size $|\lambda|^{c''}$ and constant depth d'' refutation, infinitely often.

Assume by way of contradiction that the converse holds, namely:

$$\exists\, c, d\ \forall\, c', d'\ \exists\, c'', d''\ \text{a.e.}\ \underbrace{\text{ref-IPS}_{d'}(\overbrace{\text{ckt}_d(\text{perm}_n, n^c)}^{\gamma}, |\gamma|^{c'})}_{\lambda} \vdash^{|\lambda|^{c''}}_{d''} 1 = 0, \tag{2}$$

[6] We use VAC^0 to denote Valiant's analogue of AC^0, in the same way that VP and VNP correspond to P and NP. The class VAC^0 consists of families of polynomials computable by constant-depth, polynomial-size algebraic circuits.

Using the bounded-depth version of the argument of [GP18] (i.e., Item 3 above), we get that if VNP has polynomial-size depth-d circuits, then depth-d IPS is polynomially bounded, namely:

$$\exists c_1, d_1 \ \underbrace{\mathrm{ckt}_d(\mathrm{perm}_{|\gamma|}, |\gamma|^c)}_{\gamma'} \left|\frac{|\gamma'|^{c_1}}{d_1}\right. \mathrm{ref\text{-}IPS}_d(\mathrm{ckt}_d(\mathrm{perm}_n, n^c), |\gamma|^c) \left|\frac{|\lambda|^{c''}}{d''}\right. 1 = 0,$$

where $|\lambda|$ is polynomially bounded by $|\gamma'|$. The second part from left of the refutation above is given by the fact that $\mathrm{ref\text{-}IPS}_{d'}(\mathrm{ckt}_d(\mathrm{perm}_n, n^c), |\gamma|^{c'})$ can be efficiently refuted in bounded-depth IPS for any d' and c'. Hence, we take d' to be d and c' to be c, yielding the second part of the refutation above.

Therefore, in particular, by combining the two proofs into one, we get

$$\exists c_2, d_2 \ \mathrm{ckt}_d(\mathrm{perm}_{|\gamma|}, |\gamma|^c) \left|\frac{|\gamma'|^{c_2}}{d_2}\right. 1 = 0.$$

Thus,

$$\exists c_2, d_2, \ \underbrace{\mathrm{ref\text{-}IPS}_{d_2}(\mathrm{ckt}_d(\mathrm{perm}_{|\gamma|}, |\gamma|^c), |\gamma'|^{c_2})}_{\Lambda}$$

is a *satisfiable* CNF formula. Since Equation (2) holds almost everywhere, we can take n to be $|\gamma|$. Then, by taking d' to be d_2 and c' to be c_2 in Equation (2),

$$\exists C, D \ \underbrace{\mathrm{ref\text{-}IPS}_{d_2}(\mathrm{ckt}_d(\mathrm{perm}_{|\gamma|}, |\gamma|^c), |\gamma'|^{c_2})}_{\Lambda} \left|\frac{|\Lambda|^{C}}{D}\right. 1 = 0,$$

which means Λ is refutable. By the soundness of IPS, Λ is unsatisfiable which is a contradiction.

1.4 Relation to Previous Work and Conclusion

Theorem 1.1 gives a first instance in which a formula whose validity is unknown is shown unconditionally to have no short proofs in $\mathsf{AC}^0[p]$-Frege. Razborov [Raz96] and Krajíček [Kra04b] *conjecture* that all Boolean circuit lower bound formulas are hard for EF, but this is open even for AC^0-Frege and even when we relax hardness to being non-easy (as we do in our work). Our lower bound is for *proof complexity lower bound* formulas rather than for circuit lower bound formulas, but our work is somewhat similar in spirit to [Raz95a].

From a technical point of view, our work is related to recent works by [ST25, PS19], which also use diagonalization ideas. The main result of [ST25] is a *conditional* existence of hard formulas for IPS, under the assumption that VNP $\neq$ VP. In contrast, our result ruling out easy formulas for constant-depth IPS, i.e., Theorem 1.2, is unconditional. One way to interpret our result is that it strengthens the equivalence between proof complexity lower bounds and circuit complexity lower bounds shown in [ST25] to hold for constant-depth circuits, and then applies the recent breakthroughs constant-depth algebraic circuit lower bounds [LST25, For24] to get unconditional proof complexity lower bounds.

Unconditional lower bounds for conjectured tautologies are also shown in [PS24] using a somewhat different diagonalization technique, however these are for highly non-explicit formulas[7], and do not seem directly relevant to progress on lower bounds for tautologies. In contrast, Theorem 1.6 shows that Theorem 1.2 is a *necessary step* toward super-polynomial lower bounds for constant-depth IPS.

One novel aspect in Theorem 1.1 is that a proof complexity lower bound for a propositional proof system is shown via *algebraic circuit lower bounds.* The theory of feasible interpolation in propositional

[7]There are two sources of non-explicitness in [PS24]. First, they consider a *distribution* on conjectured hard formulas rather than a fixed hard formula at every length. Second the formulas refer to a non-constructively defined proof system with non-uniform verification.

proof complexity enables proof complexity lower bounds for weak systems such as Resolution and Cutting Planes to be derived from lower bounds on monotone circuit complexity [BPR97, Kra97, Pud97]. However, there is cryptographic evidence against the applicability of feasible interpolation techniques to AC^0-Frege and stronger proof systems [BPR00]. Implications from average-case circuit lower bounds to proof size lower bounds for strong systems were conjectured by Razborov [Raz15] in the context of *proof complexity generators* [ABRW04], but these conjectures are so far unproven.

We note that the question of proving proof complexity lower bounds for constant-depth IPS has in itself been highlighted and studied in recent works [AF22, GHT22, HLT24, EGLT25, BLRS25]. Andrews and Forbes [AF22] show a super-polynomial constant-depth IPS lower bound for refuting certain sets of polynomial equations. However, their hard instances do not themselves have polynomial-size constant-depth circuits, and in particular are not CNFs. Govindasamy, Hakoniemi and Tzameret [GHT22] give a super-polynomial multilinear constant-depth IPS lower bound for refuting polynomial equations expressible as small depth-2 algebraic circuits. Hakoniemi, Limaye and Tzameret [HLT24] extend and strengthen these results from multilinear to low individual degree proofs [GHT22]. However, they also show that the lower bound framework of [GHT22, HLT24] is incapable of proving lower bounds for CNFs. Elbaz, Govindasamy, Lu and Tzameret [EGLT25] showed that any constant-depth IPS lower bounds over finite fields would lead to lower bounds for CNF formulas and thus $\mathsf{AC}^0[p]$-Frege lower bounds. Alas, the strongest constant-depth IPS lower bounds [HLT24] are restricted to low individual degree refutations, for which the result of [EGLT25] do not hold. Proving lower bounds for CNFs is essential for the application to $\mathsf{AC}^0[p]$-Frege lower bounds, and none of the previous works achieve this. Our work provides an explicit CNF formula with no short refutations and with some evidence for its unsatisfiability.

2 Preliminaries

2.1 Algebraic Complexity

Let $\mathbb{F}$ be a field. Denote by $\mathbb{F}[\overline{x}]$ the field of polynomials with coefficients from $\mathbb{F}$ and variables $\overline{x} = [x_1, \cdots, x_n]$. A polynomial is a formal linear combination of monomials, where a monomial is a product of variables. Two polynomials are identical if all their monomials have the same coefficients. The degree of a polynomial is the maximum total degree of a monomial in it.

Definition 2.1 (Depth-Δ algebraic circuits and algebraic formulas)**.** *An* algebraic circuit *over a field $\mathbb{F}$ is a finite directed acyclic graph. The leaves are called input nodes, which have in-degree zero. Each input node is labelled either with a variable or a field element in $\mathbb{F}$. All the other nodes have unbounded in-degree and are labelled by $+$ or $\times$. The output of a $+$ (or $\times$) node computes the addition (product,* resp.*) of the polynomials computed by its incoming nodes. An algebraic circuit is called an algebraic* formula *if the underlying directed acyclic graph is a tree (every node has at most one outgoing edge). The* size *of an algebraic circuit C is the number of nodes in it, denoted by $|C|$. The* depth *of C is the length of the longest directed path in it, denoted by $\mathsf{Depth}(C)$. If $\mathsf{Depth}(C) = \Delta$ we call C a* depth-Δ circuit*.*

The *product-depth* of an algebraic circuit is the maximum number of product gates on a root-to-leaf path. The product-depth is, without loss of generality, equal to the depth up to a factor of two.

Definition 2.2 (Syntactic-degree $\mathsf{sdeg}(\cdot)$)**.** *Let C be an algebraic circuit and v a node in C. The* syntactic-degree $\mathsf{sdeg}(v)$ *of v is defined as follows:*

1. *If v is a field element or a variable, then $\mathsf{sdeg}(v) := 0$ and $\mathsf{sdeg}(v) := 1$, respectively;*

2. *If* $v = \sum_{i=0}^{t} u_i$ *then* $\mathsf{sdeg}(v) := \max\{\mathsf{sdeg}(u_0), \cdots, \mathsf{sdeg}(u_t)\}$*;*

3. *If* $v = \prod_{i=0}^{t} u_i$ *then* $\mathsf{sdeg}(v) := \sum_{i=0}^{t} \mathsf{sdeg}(u_i)$*.*

Definition 2.3 (VP [Val79])**.** *Over a field $\mathbb{F}$, $\mathsf{VP}_{\mathbb{F}}$ is the class of families $f = (f_n)_{n=1}^{\infty}$ of polynomials f_n such that f_n has $\mathsf{poly}(n)$ input variables, is of $\mathsf{poly}(n)$ degree, and can be computed by algebraic circuits over $\mathbb{F}$ of $\mathsf{poly}(n)$ size.*

Definition 2.4 (VNP [Val79])**.** *Over a field $\mathbb{F}$, $\mathsf{VNP}_{\mathbb{F}}$ is the class of families $g = (g_n)_{n=1}^{\infty}$ of polynomials g_n such that g_n has $\mathsf{poly}(n)$ input variables and is of $\mathsf{poly}(n)$ degree, and can be written as*

$$g_n(x_1, \cdots, x_{\mathsf{poly}(n)}) = \sum_{\bar{e} \in \{0,1\}^{\mathsf{poly}(n)}} f_n(\bar{e}, \bar{x})$$

for some family $(f_n) \in \mathsf{VP}_{\mathbb{F}}$.

Definition 2.5 (VAC^0)**.** *Over a field $\mathbb{F}$, $\mathsf{VAC}^0_{\mathbb{F}}$ is the class of families $f = (f_n)_{n=1}^{\infty}$ of polynomials f_n such that f_n has $\mathsf{poly}(n)$ input variables, is of $\mathsf{poly}(n)$ degree, and can be computed by algebraic circuits over $\mathbb{F}$ of $\mathsf{poly}(n)$ size and depth $O(1)$.*

Notice that VP,VNP and VAC^0 are nonuniform complexity classes.

Definition 2.6 (Projection reduction [Val79])**.** *A polynomial $f(x_1, \cdots, x_n)$ is a* projection *of a polynomial $g(y_1, \cdots, y_m)$ if there is a mapping σ from $\{y_1, \cdots, y_m\}$ to $\{0, 1, x_1, \cdots, x_n\}$ such that $f(x_1, \cdots, x_n) = g(\sigma(y_1), \cdots, \sigma(y_m))$. A family of polynomials (f_n) is a* polynomial projection *or* p-projection *of another family (g_n) if there is a function $t(n) = n^{\Theta(1)}$ such that f_n is a projection of $g_{t(n)}$ for all (sufficiently large) n. We say that f is projection-reducible to g if f is a projection of g.*

The symmetric group, denoted by S_n, over n elements $\{1, \ldots, n\}$ is the group whose elements are all bijective functions from $[n]$ to $[n]$ and whose group operation is that of function composition. The sign $\mathrm{sgn}(\sigma)$ of a permutation $\sigma \in S_n$ is 1, if the permutation can be obtained with an even number of transpositions (exchanges of two [not necessarily consecutive] entries); otherwise, it is -1.

Definition 2.7 (Determinant)**.** *The* Determinant *of an $n \times n$ matrix A is defined as*

$$\det(A) := \sum_{\sigma \in S_n} (\mathrm{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)}).$$

Definition 2.8 (Permanent)**.** *The* Permanent *of an $n \times n$ matrix $A = (a_{ij})$ is defined as*

$$\mathrm{perm}(A) := \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}.$$

It is known that the Permanent polynomial is complete under p-projections for VNP when the field $\mathbb{F}$ is a field of characteristic different from 2. The Determinant polynomial is not known to be complete for VP under p-projections.

Theorem 2.9 ([Val79])**.** *For every field $\mathbb{F}$, every polynomial family on n variables that is computable by an algebraic formula of size u is projection reducible to the Determinant polynomial (over the same field) on $u + 2$ variables. For every field $\mathbb{F}$, except those that have characteristic 2, every polynomial family in $\mathsf{VNP}_{\mathbb{F}}$ is projection reducible to the Permanent polynomial (over the same field) with polynomially more variables.*

Definition 2.10 (Iterated Matrix Multiplication)**.** *Let n and d be such that $N = dn^2$. The* Iterated Matrix Multiplication $\mathsf{IMM}_{n,d}$ *on $N = dn^2$ variables is defined as the following polynomial. The underlying variables are partitioned into d sets $X_1, \cdots, X_d$ of size n^2, each of which is represented as an $n \times n$ matrix with distinct variable entries. Then $\mathsf{IMM}_{n,d}$ is defined to be the polynomial that is the* $(1,1)$*th entry of the product matrix $X_1 \cdot X_2 \cdots X_d$.*

Iterated Matrix Multiplication is in VP.

Theorem 2.11 (Super-polynomial lower bounds against constant-depth circuits over large field [LST25])**.** *Assume $d \leq \frac{\log n}{100}$ and the characteristic of $\mathbb{F}$ is 0 or greater than d. For any product-depth $\Delta \geq 1$, any algebraic circuit C computing $\mathsf{IMM}_{n,d}$ of product-depth at most Δ must have size at least $n^{d^{\exp -O(\Delta)}}$.*

[BDS24] improved the lower bound for IMM against constant-depth. Let $\mu(\Delta) = 1/(F(\Delta) - 1)$, where $F(n) = \Theta(\varphi^n)$ is the nth Fibonacci number (starting with $F(0) = 1$, $F(1) = 2$) and $\varphi = (1 + \sqrt{5})/2$ is the golden ratio.

Theorem 2.12 ([BDS24])**.** *Fixed a field $\mathbb{F}$ of characteristic 0 or greater than d. Let N, d, Δ be such that $d = O(\log / \log\log N)$. Then, any product-depth Δ circuit computing $\mathsf{IMM}_{n,d}$ on $N = dn^2$ variables must have size at least $N^{\Omega(d^{\mu(2\Delta)}/\Delta)}$.*

A recent result by Forbes [For24] extended these results to *any* field, including finite fields.

Corollary 2.13 (Super-polynomial lower bounds on constant-depth circuits over any field [For24])**.** *Let $\mathbb{F}$ be any field, and $d = o(\log n)$. Then the iterated matrix multiplication polynomial $\mathsf{IMM}_{n,d}$ where X_i requires*

$$n^{\Theta(d^{\mu(2\Delta)}/\Delta)}$$

size algebraic circuits of product depth Δ.

Since $\mathsf{IMM}_{n,d}$ is a p-projection of the Permanent polynomial with $\mathsf{poly}(n, d)$ many variables, it follows that the Permanent does not have constant-depth polynomial-size circuits over any field, in the following sense.

Theorem 2.14 (No polynomial-size constant depth circuits for the Permanent [LST25, For24])**.** *Let $\Delta \geq 1$ and let $\mathbb{F}$ be any field. There are no constants c_1, c_2, such that the Permanent polynomial* $\mathrm{perm}(A)$ *of the $n \times n$ symbolic matrix A over the field $\mathbb{F}$ is computable by an algebraic circuit of size $c_1 n^{c_2}$ and depth Δ where Δ is independent with n, for sufficiently large n.*

2.2 Proof Complexity

Definition 2.15 (Propositional proof system, [CR79])**.** *A propositional proof system is a polynomial-time computable relation $R(\cdot, \cdot)$ such that for each $x \in \{0,1\}^*$, $x \in \mathrm{TAUT}$, if and only if there exists $y \in \{0,1\}^*$ such that $R(x, y)$ holds. Given $x \in \mathrm{TAUT}$, any y for which $R(x, y)$ holds is called an R-proof of x. A propositional proof system R is polynomially bounded (**p-bounded**) if there exists a polynomial p such that for each $x \in \mathrm{TAUT}$, there is an R-proof y of x of size at most $p(|x|)$ (*i.e. $|y| \leq p(|x|)$*).*

Definition 2.16 (p-simulation)**.** *Let P and Q be propositional proof systems. We say that P* p-simulates *Q, written $Q \leq_p P$, if there exists a polynomial $p(\cdot)$ such that for every propositional tautology φ and every Q-proof π of φ of size s, there exists a P-proof π' of φ whose size is at most $p(s)$.*

Definition 2.17 (Frege, AC^0-Frege and $\mathsf{AC}^0[p]$-Frege)**.** *A* Frege rule *is an inference rule of the form:* $B_1, \ldots, B_n \implies B$*, where* $B_1, \ldots, B_n, B$ *are propositional formulas. If* $n = 0$*, then the rule is an axiom. A* Frege system *is specified by a finite set,* R*, of rules. Given a collection* R *of rules, a derivation of a 3DNF formula* f *is a sequence of formulas* $f_1, \ldots, f_m$ *such that each* f_i *is either an instance of an axiom scheme or follows from previous formulas by one of the rules in* R *and such that the final formula* f_m *is* f*.*

AC^0-Frege *are Frege proofs but with the additional restriction that each formula in the proof has bounded depth.*

$\mathsf{AC}^0[p]$-Frege *are bounded-depth Frege proofs that also allow unbounded fan-in* MOD_p *connectives, namely* MOD_p^i *for* $i \in \{0, \ldots, p-1\}$*.* $\mathrm{MOD}_p^i(x_1, \ldots, x_k)$ *evaluates to true if the number of* x_i *that are true is congruent to* $i \mod (p)$ *and evaluates to false otherwise.*

Definition 2.18 (Polynomial Calculus [CEI96])**.** *Given a field* $\mathbb{F}$ *and a set of variables, a* polynomial calculus *(*PC*) refutation of the set of axioms* P *is a sequence of polynomials such that the last line is the polynomial* 1 *and each line is either an axiom or is derived from the previous lines using the following inference rules:*

$$\frac{f \qquad g}{\alpha f + \beta g}$$

and

$$\frac{f}{x \cdot f},$$

where $\alpha, \beta \in \mathbb{F}$ *are any scalars and* x *is an variable. The refutation has degree* d *if all the polynomials in it have degrees at most* d*.*

The degree *of a* PC *proof is defined as the maximal degree of a polynomial appearing in it, and its* size *is the number of different* monomials *in this proof.*

Definition 2.19 (Polynomial Calculus with Resolution [ABSRW02])**.** *Let* $\mathbb{F}$ *be a fixed field.* Polynomial Calculus with Resolution *(*PCR*) is the proof system whose lines are polynomials from* $\mathbb{F}[x_1, \cdots, x_n, \overline{x_1}, \cdots, \overline{x_n}]$*, where* $\overline{x_1}, \cdots, \overline{x_n}$ *are treated as new formal variables.* PCR *has all default axioms and inference rules of* PC *(including, of course, those that involve new variables* $\overline{x_i}$*), plus additional default axioms* $x_i + \overline{x_i} = 1$ $(i \in [n])$*.*

For a clause C*, denote by* Γ_C *the monomial*

$$\Gamma_C \coloneqq \prod_{\overline{x} \in C} x \cdot \prod_{x \in C} \overline{x}$$

and for a CNF τ*, let* $\Gamma_\tau \coloneqq \{\Gamma_C | C \in \tau\}$*. (Note that* τ *is unsatisfiable if and only if the polynomials* Γ_τ *have no common root in* $\mathbb{F}$ *satisfying all default axioms of* PCR*.) A* PCR refutation *of a* CNF τ *is a* PCR *proof of the contradiction* $1 = 0$ *from* Γ_τ*.*

The degree *of a* PCR *proof is defined as the maximal degree of a polynomial appearing in it, and its* size *is the number of different* monomials *in this proof.*

PC and PCR are equivalent with respect to the degree measure (via the linear transformation $\overline{x} \implies 1 - x_i$).

2.3 Ideal Proof System

Given $f_1, \cdots, f_m \in \mathbb{F}[x_1, \cdots, x_n]$ over some field $\mathbb{F}$, Hilbert's Nullstellensatz shows that $f_1(\overline{x}) = \cdots = f_m(\overline{x}) = 0$ is unsatisfiable (over the algebraic closure of $\mathbb{F}$) if and only if there are polynomials $g_1, \cdots, g_m \in \mathbb{F}[\overline{x}]$ such that $\sum_j g_j(\overline{x}) f_j(\overline{x}) = 1$ (as a formal identity), or equivalently, that 1 is in the ideal generated by the $\{f_j\}_j$.

Definition 2.20 ((Boolean) Ideal proof system (IPS) [GP18])**.** *Let $f_1(\overline{x}), \cdots, f_m(\overline{x}), p(\overline{x}) \in \mathbb{F}[x_1, \cdots, x_n]$ be a collection of polynomials. An* IPS *proof of $p(\overline{x}) = 0$ from $\{f_j(\overline{x})\}_{j=1}^m$, showing that $p(\overline{x}) = 0$ is semantically implied from the assumptions $\{f_j(\overline{x}) = 0\}_{j=1}^m$ over 0-1 assignments, is an algebraic circuit $C(\overline{x}, \overline{y}, \overline{z}) \in \mathbb{F}[\overline{x}, y_1, \cdots, y_m, z_1, \cdots, z_n]$, such that (the equalities in what follows stand for formal polynomial identities[8])*

1. *$C(\overline{x}, \overline{0}, \overline{0}) = 0$.*

2. *$C(\overline{x}, f_1(\overline{x}), \cdots, f_m(\overline{x}), x_1^2 - x_1, \cdots, x_n^2 - x_n) = p(\overline{x})$.*

The size of the IPS *proof is the size of the circuit C. The variables $\overline{y}, \overline{z}$ are sometimes called the placeholder variables since they are used as a placeholder for the axioms. An* IPS *proof $C(\overline{x}, \overline{y}, \overline{z})$ of $1 = 0$ from $\{f_j(\overline{x}) = 0\}_{j=1}^m$ is called an* IPS *refutation of $\{f_j(\overline{x}) = 0\}_{j=1}^m$. If C comes from a restricted class of algebraic circuits $\mathcal{C}$, then this is called a $\mathcal{C}$-*IPS *refutation.*

We shall also use the algebraic version of IPS (which does not use the Boolean axioms):

Definition 2.21 ((Algebraic) Ideal proof system ($\mathsf{IPS}^{\mathsf{alg}}$) [GP18])**.** *Let $f_1(\overline{x}), \cdots, f_m(\overline{x}), p(\overline{x}) \in \mathbb{F}[x_1, \cdots, x_n]$ be a collection of polynomials. An $\mathsf{IPS}^{\mathsf{alg}}$ proof of $p(\overline{x}) = 0$ from $\{f_j(\overline{x})\}_{j=1}^m$, showing that $p(\overline{x}) = 0$ is semantically implied from the assumptions $\{f_j(\overline{x}) = 0\}_{j=1}^m$ over assignments by field elements, is an algebraic circuit $C(\overline{x}, \overline{y}) \in \mathbb{F}[\overline{x}, y_1, \cdots, y_m]$, such that*

1. *$C(\overline{x}, \overline{0}) = 0$.*

2. *$C(\overline{x}, f_1(\overline{x}), \cdots, f_m(\overline{x})) = p(\overline{x})$.*

The size and refutation are defined similarly to Definition 2.20.

Now, we introduce some notation we will use in the following sections. Let $\overline{\mathcal{F}} = \{f_i(\overline{x}) = 0\}_{i=1}^m$ be a collection of circuit equations, namely the f_i's are written as algebraic circuits. We use $|\overline{\mathcal{F}}|$ to denote the total size of the circuit equations in $\overline{\mathcal{F}}$. We denote by $C : \overline{\mathcal{F}} \,\left|\frac{s,\Delta}{\mathsf{IPS}}\right.\, 1 = 0$ the fact that $\overline{\mathcal{F}}$ has an IPS refutation C of size at most s and depth at most Δ. If we do not care about the explicit size of the IPS refutation, we denote by $C : \overline{\mathcal{F}} \,\left|\frac{*,\Delta}{\mathsf{IPS}}\right.\, 1 = 0$ the fact that $\overline{\mathcal{F}}$ has an IPS refutation C of size polynomially bounded by $|\overline{\mathcal{F}}|$ and of depth Δ.

When we deal with algebraic IPS refutations, we will use the same notation as above, only using $\mathsf{IPS}^{\mathsf{alg}}$ instead of IPS.

Polynomial identities are proved for free in IPS, which was observed in [AGHT24], and this also holds for constant-depth IPS proofs.

Proposition 2.22. *If $C(\overline{x})$ is a Depth-Δ algebraic circuit in the variables $\overline{x}$ over the field $\mathbb{F}$ that computes the zero polynomial, then there is an Depth-Δ* IPS *proof of $C(\overline{x}) = 0$ of size $|F|$.*

The following proposition can be regarded as a constant-depth analogue of Proposition A.5 in [AGHT24].

Proposition 2.23 (proof by boolean cases in bounded-depth IPS)**.** *Let $\mathbb{F}$ be a field. Let $\overline{\mathcal{F}}$ be a collection of m many circuit equations over n many variables $\overline{x}$. Assume that for every fixed assignment $\overline{\alpha} \in \{0,1\}^r$ where $0 \le r \le n$ we have*

$$\sum_{i=1}^{m} G_i \cdot F_i + \sum_{i=1}^{r} L_i \cdot (x_i - \alpha_i) + \sum_{i=1}^{n} Q_i \cdot (x_i^2 - x_i) = f(\overline{x})$$

[8]That is, $C(\overline{x}, \overline{0}, \overline{0})$ computes the zero polynomial and $C(\overline{x}, f_1(\overline{x}), \cdots, f_m(\overline{x}), x_1^2 - x_1, \cdots, x_n^2 - x_n)$ computes the polynomial $p(\overline{x})$

where each G_i, L_i and Q_i has size s and depth-2 (in other words, each G_i, L_i, and Q_i is just a summation of terms[9]*), then there exist G'_i and Q'_i such that*

$$\sum_{i=1}^{m} G'_i \cdot F_i + \sum_{i=1}^{n} Q'_i \cdot (x_i^2 - x_i) = f(\overline{x})$$

where each G'_i, Q'_i has size $c^r \cdot s$ and depth-2 for some constant c independent of r.

Proof. We prove by induction on r.

Base case: $r = 0$. Assume that

$$\sum_{i=1}^{m} G_i \cdot F_i + \sum_{i=1}^{n} Q_i \cdot (x_i^2 - x_i) = f(\overline{x})$$

where each G_i and Q_i has size s and depth-2, then clearly there exist G'_i and Q'_i such that

$$\sum_{i=1}^{m} G'_i \cdot F_i + \sum_{i=1}^{n} Q'_i \cdot (x_i^2 - x_i) = f(\overline{x})$$

where each G'_i, Q'_i has size $c^r \cdot s$ (which is s in the base case) and depth-2 for some constant c independent of r.

induction step: $r > 0$. Suppose for any assignment $\alpha \in \{0,1\}^r$,

$$\sum_{i=1}^{m} G_i \cdot F_i + \sum_{i=1}^{r} L_i \cdot (x_i - \alpha_i) + \sum_{i=1}^{n} Q_i \cdot (x_i^2 - x_i) = f(\overline{x})$$

where each G_i, L_i and Q_i has size s and depth-2. We aim to show that there exist G'_i and Q'_i such that

$$\sum_{i=1}^{m} G'_i \cdot F_i + \sum_{i=1}^{n} Q'_i \cdot (x_i^2 - x_i) = f()$$

where each G'_i, Q'_i has size $c^r \cdot s$ and depth-2 for some constant c independent of r.

Then, by our assumption, we know that for every fixed assignment $\overline{\alpha} \in \{0,1\}^{r-1}$ we have:

$$\sum_{i=1}^{m} G_i \cdot F_i + \sum_{i=2}^{r} L_i \cdot (x_i - \alpha_i) + M \cdot x_1 + \sum_{i=1}^{n} Q_i \cdot (x_i^2 - x_i) = f(\overline{x}) \quad (3)$$

$$\sum_{i=1}^{m} P_i \cdot F_i + \sum_{i=2}^{r} K_i \cdot (x_i - \alpha_i) + N \cdot (1 - x_1) + \sum_{i=1}^{n} W_i \cdot (x_i^2 - x_i) = f(\overline{x}) \quad (4)$$

where each G_i, L_i, M, Q_i, P_i, K_i, N and W_i is of size s and depth 2. By the induction hypothesis,

$$\sum_{i=1}^{m} G'_i \cdot F_i + M' \cdot x_1 + \sum_{i=1}^{n} Q'_i \cdot (x_i^2 - x_i) = f(\overline{x}) \quad (5)$$

$$\sum_{i=1}^{m} P'_i \cdot F_i + N' \cdot (1 - x_1) + \sum_{i=1}^{n} W'_i \cdot (x_i^2 - x_i) = f(\overline{x}) \quad (6)$$

where each G'_i, M', Q'_i, P'_i, N' and W'_i is of size $c^{r-1} \cdot s$ and depth 2.

[9]A term is a monomial multiplied by a field element.

By multiplying Equation (5) and Equation (6) with $1 - x_1$ and x_1, respectively, we get

$$\sum_{i=1}^{m}(1-x_1)\cdot G'_i \cdot F_i + (1-x)\cdot M' \cdot x_1 + \sum_{i=1}^{n}(1-x_1)Q'_i \cdot (x_i^2 - x_i) = (1-x_1)\cdot f(\overline{x}) \tag{7}$$

$$\sum_{i=1}^{m} x_1 \cdot P'_i \cdot F_i + x_1 \cdot N' \cdot (1-x_1) + \sum_{i=1}^{m} x_1 \cdot W'_i \cdot (x_i^2 - x_i) = x_1 \cdot f(\overline{x}) \tag{8}$$

By summing Equation (7) and Equation (8), we get

$$\sum_{i=1}^{m}[(1-x_1)G'_i + x_1 P'_i]\cdot F_i + [(1-x_1)M' + (1-x_1)Q'_1 + x_1 N' + x_1 W'_1]\cdot (x_1^2 - x_1) +$$
$$\sum_{i=2}^{n}[(1-x_1)Q'_i + x_1 W'_i]\cdot (x_i^2 - x_i) = f(\overline{x})$$

Note that both $(1-x_1)G'_i + x_1P'_i$, $(1-x_1)M' + (1-x_1)Q'_1 + x_1N' + x_1W'_1$ and $(1-x_1)Q'_i + x_1W'_i$ can be computed by an algebraic circuit of size at most $6\cdot c^{r-1}\cdot s \le c^r \cdot s$ and depth 2 for large enough c independent with r. This concludes the proof of the proposition. □

The following theorem is from [GP18], and it already holds for $\mathsf{IPS}^{\mathsf{alg}}$.

Theorem 2.24 (Superpolynomial IPS lower bounds imply VNP $\neq$ VP [GP18])**.** *For any field $\mathbb{F}$, a superpolynomial lower bound on $\mathsf{IPS}^{\mathsf{alg}}$ (also IPS) refutations over $\mathbb{F}$ for any family of CNF formulas implies $\mathsf{VNP}_{\mathbb{F}} \neq \mathsf{VP}_{\mathbb{F}}$. The same result holds if we assume that the $\mathsf{IPS}^{\mathsf{alg}}$ (IPS) refutation size lower bound holds only infinitely often.*

Lemma 2.25 ([GP18])**.** *Every family of unsatisfiable CNF formulas (φ_n) has a family of $\mathsf{IPS}^{\mathsf{alg}}$ (also IPS) certificates (C_n) in $\mathsf{VNP}_{\mathbb{F}}$.*

2.4 Encoding in Fixed Finite Fields

In the section, we are working in the finite field $\mathbb{F}_q$ where q is a constant (independent of the size of the formulas and their number of variables). When we work with CNF formulas in IPS we assume that the CNF formulas are translated as follows:

Definition 2.26 (Algebraic translation of CNF formulas)**.** *Given a CNF formula in the variables $\overline{x}$, every clause $\bigvee_{i\in P} x_i \vee \bigvee_{j\in N} \neg x_j$ is translated into $\prod_{i\in P}(1-x_i)\cdot\prod_{j\in N} x_j = 0$. (Note that these terms are written as algebraic circuits as displayed, where products are not multiplied out.)*

Notice that a CNF formula is satisfiable by 0-1 assignment if and only if the assignment satisfies all the equations in the algebraic translation of the CNF.

The following definitions are taken from [ST25], and we supply them here for completeness.

Definition 2.27 (Algebraic extension axioms and unary bits [ST25])**.** *Given a circuit C and a node g in C, we call the equation*

$$x_g = \sum_{i=0}^{q-1} i \cdot x_{g_i}$$

the algebraic extension axiom of g, with each variables x_{g_i} being the ith unary-bit of g.

Definition 2.28 (Plain CNF encoding of bounded-depth algebraic circuit $\mathsf{cnf}(C(\overline{x}))$ [ST25])**.** *Let $C(\overline{x})$ be a circuit in the variables $\overline{x}$. The plain* CNF *encoding of the circuit $C(\overline{x})$ denoted* $\mathsf{cnf}(C(\overline{x}))$ *consists of the following* CNF*s in the unary-bit variables corresponding to all the gates in C and all the extra extension variables in Item 3:*

1. *If x_i is an input node in C, the plain* CNF *encoding of C uses the variables $x_{x_{i0}}, \cdots, x_{x_{i(q-1)}}$ that are the unary-bits of x_i, and contains the clauses that express that precisely one unary-bit is 1 and all other unary-bits are 0:*
$$\bigvee_{j=0}^{q-1} x_{x_i j} \wedge \bigwedge_{j \neq l \in \{0,\cdots,q-1\}} (\neg x_{x_i j} \vee \neg x_{x_i l}).$$

2. *If $\alpha \in \mathbb{F}_q$ is a scalar input node in C, the plain* CNF *encoding of C contains the $\{0,1\}$ constants corresponding to the unary-bits of α. These constants are used when fed to (translation of) gates according to the wiring of C in item 4.*

3. *For every node g in $C(\overline{x})$ and every satisfying assignment $\overline{\alpha}$ to the plain* CNF *encoding, the corresponding unary-bit x_{g_i} evaluates to 1 if and only if the value of g is $i \in \{0, \cdots, q-1\}$ (when the algebraic inputs $\overline{x} \in (\mathbb{F}_q)^*$ to $C(\overline{x})$ take on the values corresponding to the Boolean assignment $\overline{\alpha}$; "$*$" here means the Kleene star). This is ensured with the following equations: if g is a $\circ \in \{+, \times\}$ node that has inputs $u_1, \cdots, u_t$. Then we consider the following equations:*
$$\begin{aligned} &u_1 \circ u_2 = v_1^g \\ &u_{i+2} \circ v_i^g = v_{i+1}^g, \qquad 1 \le i \le t-3 \\ &u_t \circ v_{t-2}^g = g. \end{aligned}$$
In other words, we add the extension variables *v_i^g for each $+, \times$ gate, to sequentially compute the unbounded fan-in gate g into a sequence of binary operations in the obvious way. For simplicity, we denote each equation above by $x \circ y = z$. Then, for each $x \circ y = z$ we have a* CNF *ϕ in the unary-bits variables of x, y, z that is satisfied by assignment precisely when the output unary-bits of z get their correct values based on the (constant-size) truth table of $\circ$ over $\mathbb{F}_q$ and the input unary-bits of x, y (we ensure that if more than one unary-bit is assigned 1 in any of the unary-bits of x, y, z then the* CNF *is unsatisfiable).*

4. *For every unary-bit variable x_{g_i}, we have the Boolean axiom (recall we write these Boolean axioms explicitly since we are going to work with* $\mathsf{IPS}^{\mathsf{alg}}$*):*
$$x_{g_i}^2 - x_{g_i} = 0.$$

Therefore, we can see that the formula size of $\mathsf{cnf}(C(\overline{x}) = 0)$ *is* $\mathsf{poly}(q^2 \cdot |C|)$*.*

Note that the only variables in a plain CNF encoding are unary-bit variables.

Definition 2.29 (Plain CNF encoding of a bounded-depth circuit equation $\mathsf{cnf}(C(\overline{x}) = 0)$ [ST25])**.** *Let $C(\overline{x})$ be a circuit in the variables $\overline{x}$. The plain* CNF *encoding of the circuit equation $C(\overline{x}) = 0$ denoted* $\mathsf{cnf}(C(\overline{x}) = 0)$ *consists of the following* CNF *encoding from Definition 2.28 in the unary-bits variables of all the gates in C (and only in the unary-bit variables), together with the equations:*
$$x_{g_{out}0} = 1 \quad \text{and} \quad x_{g_{out}i} = 0, \quad \text{for all } i = 1, \cdots, q-1,$$
which express that $g_{out} = 0$, where g_{out} is the output node of C.

Definition 2.30 (Extended CNF encoding of a circuit equation (circuit, *resp.*); $\mathsf{ecnf}(C(\overline{x}) = 0)$ ($\mathsf{ecnf}(C(\overline{x}))$, *resp.*) [ST25]). *Let $C(\overline{x})$ be a circuit in the variables $\overline{x}$ over the finite filed $\mathbb{F}_q$. The extended* CNF *encoding of the circuit equation $C(\overline{x}) = 0$ (circuit $C(\overline{x})$,* resp.*), in symbols $\mathsf{ecnf}(C(\overline{x}) = 0)$ ($\mathsf{ecnf}(C(\overline{x}))$,* resp.*), is defined to be a set of algebraic equations over $\mathbb{F}_q$ in the variables x_g and $x_{g0}, \cdots, x_{gq-1}$ which are the unary-bit variables corresponding to the node g in C, that consist of:*

1. *the plain* CNF *encoding of the circuit equation $C(\overline{x}) = 0$ (circuit $C(\overline{x})$,* resp.*), namely, $\mathsf{cnf}(C(\overline{x}) = 0)$ ($\mathsf{cnf}(C(\overline{x}))$,* resp.*); and*

2. *the algebraic extension axiom of g, for every gate g in C.*

Since we work with extension variables for each gate in a given circuit equation $C(\overline{x}) = 0$, it is more convenient to express circuit equations as a set of equations that correspond to the straight line program of $C(\overline{x})$ (which is equivalent in strength formulation to algebraic circuits):

Definition 2.31 (Straight line program (SLP)). *An* SLP *of a circuit $C(\overline{x})$, denoted by $\mathsf{SLP}(C(\overline{x}))$, is a sequence of equations between variables such that the extension variable for the output node computes the value of the circuit assuming all equations hold. Formally, we choose any topological order $g_1, g_2, \cdots, g_i, \cdots, g_{|C|}$ on the nodes of the circuit C (that is, if g_j has a directed path to g_k in C then $j < k$) and define the following set of equations to be the* SLP *of $C(\overline{x})$:*

$g_i = g_{j1} \circ g_{j2} \circ \cdots \circ g_{jt}$ for $\circ \in \{+, \times\}$ iff g_i is a $\circ$ node in C with t incoming edges from $g_{j1}, \cdots, g_{jt}$.

An SLP *representation of a circuit equation $C(\overline{x}) = 0$ means that we add to the* SLP *above the equation $g_{|C|} = 0$, where $g_{|C|}$ is the output node of the circuit.*

The below lemma, which we refer to as the translation lemma in this paper, shows that we can derive the circuit equations from the extended CNF formulas encoding those circuit equations with some additional axioms and vice versa.

Lemma 2.32 (Translating between extended CNFs and circuit equations in fixed finite fields [ST25]). *Let $\mathbb{F}_q$ be a finite field, and let $C(\overline{x})$ be a circuit of depth Δ in the $\overline{x}$ variables over $\mathbb{F}_q$. Then, the following both hold*

$$\mathsf{ecnf}(C(\overline{x}) = 0) \,\Big|\!\frac{\;{*,O(\Delta)}\;}{\mathsf{IPS}^{\mathsf{alg}}}\, C(\overline{x}) = 0 \tag{9}$$

$$\begin{array}{r} \left\{x_g = \sum_{i=0}^{q-1} i \cdot x_{gi} : g \text{ a node in } C\right\}, \\ \left\{x_{gi}^2 - x_{gi} = 0 : g \text{ is a node in } C,\ 0 \le i < q\right\}, \\ \left\{\sum_{i=0}^{q-1} x_{gi} = 1 : g \text{ is a node in } C\right\},\ C(\overline{x}) = 0,\ \mathsf{SLP}(C(\overline{x})) \end{array} \,\Big|\!\frac{\;{*,O(\Delta)}\;}{\mathsf{IPS}^{\mathsf{alg}}}\, \mathsf{ecnf}\big(C(\overline{x}) = 0\big), \tag{10}$$

Proposition 2.33 (Proposition 3.7 in [ST25]). *Let $C(\overline{x}) = 0$ be a circuit equation over $\mathbb{F}_q$ where q is any constant prime. Then, $C(\overline{x}) = 0$ is unsatisfiable over $\mathbb{F}_q$ iff $\mathsf{cnf}(C(\overline{x}) = 0)$ is an unsatisfiable* CNF *iff $\mathsf{ecnf}(C(\overline{x}) = 0)$ is an unsatisfiable set of equations over $\mathbb{F}_q$.*

Using results in [EGLT25], we could remove some extension axioms used in [ST25] when working over fixed finite fields. We use $\mathsf{UBIT}_j(x)$ to denote the following Lagrange polynomial:

$$\mathsf{UBIT}_j(x) := \frac{\prod_{i=0, i\neq j}^{q-1}(x - i)}{\prod_{i=0, i\neq j}^{q-1}(j - i)} \tag{11}$$

where x can be a single variable or an algebraic circuit. Hence, it is easy to observe that

$$\mathsf{UBIT}_j(x) = \begin{cases} 1, & x = j, \\ 0, & \text{otherwise.} \end{cases}$$

Also, suppose x has size $|x|$ and depth $\mathsf{Depth}(x)$ (when x is a single variable, it has size 1 and depth 1), $\mathsf{UBIT}_j(x)$ can be computed by an algebraic circuit of size $O(|x|^{q-1})$ and depth $\mathsf{Depth}(x) + 2$.

Definition 2.34 (Semi-CNF SCNF encoding of a bounded-depth circuit equation $\mathsf{SCNF}(C(\overline{x}) = 0)$)**.** *Let $C(\overline{x})$ be a circuit in the variables $\overline{x}$. The semi-CNF encoding of the circuit equation $C(\overline{x}) = 0$ denoted $\mathsf{SCNF}(C(\overline{x}))$ is a substitution instance of the plain CNF encoding in Definition 2.29 where each unary-bits x_{uj} of all the gates and extra extension variables*[10] *u is substituted with $\mathsf{UBIT}_j(C_u)$ where C_u is the bounded-depth algebraic circuit computed by u.*[11]

We call $x^q - x = 0$ the *field axiom* for the variable x.

Lemma 2.35 (Translate semi-CNFs from circuit equations in Fixed Finite Fields, [EGLT25])**.** *Let $\mathbb{F}_q$ be a finite field, and let $C(\overline{x})$ be a circuit of depth Δ in the $\overline{x}$ variables over $\mathbb{F}_q$. Then, the following hold*

$$\{x^q - x = 0 : x \text{ is a variable in } C\}, C(\overline{x}) = 0 \left|\frac{\;*,O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{SCNF}(C(\overline{x}) = 0)$$

Lemma 2.36 (Translate circuit equations from semi-CNFs in fixed finite fields, [EGLT25])**.** *Let $\mathbb{F}_q$ be a finite field, and let $C(\overline{x})$ be a circuit of depth Δ in the $\overline{x}$ variables over $\mathbb{F}_q$. Then, the following hold*

$$\{x^q - x = 0 : x \text{ is a variable in } C\}, \mathsf{SCNF}(C(\overline{x}) = 0) \left|\frac{\;*,O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. C(\overline{x}) = 0$$

We will use our new translation lemma for the next section, which is our main result in fixed finite fields. For polynomial-size finite fields, we have a different translation lemma that we will explain later.

Proposition 2.37. *Let $C(\overline{x}) = 0$ be a circuit equation over $\mathbb{F}_q$ where q is any constant prime. Then, $C(\overline{x}) = 0$ is unsatisfiable over $\mathbb{F}_q$ iff $\mathsf{scnf}(C(\overline{x}) = 0)$ is an unsatisfiable SCNF.*

3 Universal Algebraic Circuits for Bounded-Depth

Here, we develop the necessary technical information regarding universal circuits. This is a novel adaptation of the work of Raz [Raz10] to the bounded-depth setting, in which both the universal circuit and the circuits it encodes are of bounded depth.

In this section, we will work with algebraic circuits whose edges can be labelled by field elements. This does not make too much difference, as we can easily replace them with a multiplication, which only increases the depth and size of a circuit up to a factor of 2.

For general algebraic circuits, we have the following.

Theorem 3.1 (Existence of universal circuits for homogeneous polynomials [Raz10])**.** *Let $\mathbb{F}$ be a field and $\overline{x}$ be n variables, and let $C^{\text{hom}}_{s,d}$ denote the class of all homogeneous polynomials of total degree exactly d in $\mathbb{F}[\overline{x}]$ that have algebraic circuits of size at most s. Then there is a circuit $U(\overline{x}, \overline{w}) \in \mathbb{F}[\overline{x}, \overline{w}]$ of size $O(d^2 s^8)$ and syntactic-degree $O(d)$ such that $\overline{w}$ are $K_{s,d} = O(d^2 s^8)$ many variables which are disjoint from $\overline{x}$, that is universal for $C^{\text{hom}}_{s,d}$ in the following sense: if $f(\overline{x}) \in C^{\text{hom}}_{s,d}$, then there exists $\overline{a} \in \mathbb{F}^{K_{s,d}}$ such that $U(\overline{x}, \overline{a}) = f(\overline{x})$. Notice that given s and d, $K_{s,d}$ can be computed efficiently.*

[10] These extension variables are used in Item 3 of Definition 2.28 to help encode the circuit.

[11] This C_u can be constructed from SLPs easily.

We use the following simple adaptation from [ST25]:

Definition 3.2 (Universal circuits for polynomials [ST25])**.** *The universal circuit for degree d and size s circuits is defined as:*

$$U(\overline{x}, \overline{w}) = \sum_{i=0}^{d} U_i(\overline{x}, \overline{w}),$$

where $U_i(\overline{x}, \overline{w})$ is the universal circuit for homogeneous $\overline{x}$-polynomials of i degree $C^{\mathrm{hom}}_{s,i}$ and where the $\overline{w}$-variables in each distinct $U_i(\overline{x}, \overline{w})$ are pairwise disjoint.

The size of $U(\overline{x}, \overline{w})$ is $\sum_{i=0}^{d} O(i^4 s^8) = O(d^5 s^8)$.

Definition 3.3 (Circuit-graph [Raz10])**.** *Let Φ be an algebraic circuit. We denote by G_Φ the underlying graph of Φ, together with the labels of all nodes. That is, the entire circuit, except for the labels of the edges. We call G_Φ, the circuit-graph of Φ.*

We will need universal circuits for bounded-depth circuits, where the universal circuits are bounded-depth themselves. We say a circuit-graph G is depth-Δ if $\mathsf{Depth}(G) \leq \Delta$.

Definition 3.4 (Normal-Depth-Form)**.** *Let G be a depth-Δ circuit-graph. We say that G is in Normal-Depth-Form if it satisfies:*

1. *All edges from the leaves are to $+$ nodes.*
2. *All output-nodes are $+$ nodes.*
3. *The nodes of G are alternating. That is, if v is a $+$ node and (u, v) is an edge, then u is either a leaf or a $\times$ node and if v is a $\times$ node and (u, v) is an edge then u is a $+$ node.*
4. *The out-degree of every node is at most 1.*
5. *The depth of every leaf is the same.*

We say that an algebraic circuit Φ is in normal-depth-form *if the circuit-graph G_Φ is in normal-depth-form.*

Lemma 3.5 (Existence of normal-depth-form algebraic circuits for bounded-depth algebraic circuits)**.** *Let $\mathbb{F}$ be a field and Δ be a constant. Let Φ be a depth-Δ algebraic circuit of size s for a polynomial $g \in \mathbb{F}[x_1, \cdots, x_n]$. Then, there exists an algebraic circuit Φ' that computes g such that Φ' is a normal-depth-form, and the number of nodes in Φ' is $\mathsf{poly}(s)$. Moreover, given Φ (as an input), Φ' can be efficiently constructed.*

Proof. First, we turn our depth-Δ algebraic circuit Φ into a depth-Δ algebraic formula φ of size $\mathsf{poly}(s \cdot 2^\Delta)$. Since Δ is a constant, φ is of size $\mathsf{poly}(s)$.

Then, by merging nodes and adding dummy nodes ($+$ nodes or $\times$ nodes such that only have one input and one output), we can construct Φ' from φ. To be specific, for an edge (u, v) in φ, if both u and v are $+$ nodes (*resp.*, $\times$ nodes), we merge them to one $+$ node (*resp.*, one $\times$ node). Then, if (u, v) is an edge after merging u is a $\times$ node, and v is a leaf, then we add a $+$ node o between u and v such that (u, o) and (o, v) are labelled with 1. If the output node u is a $\times$ node, we add a $+$ node v above it and label (u, v) with 1. Also, if the depth of a leaf is smaller than the maximum depth, by alternatively adding dummy $+$ nodes and $\times$ nodes, we can make the depth of every leaf the same.

We can see that Φ' has bounded-depth and is of size $\mathsf{poly}(s)$. Φ' is in normal-depth-form. □

Remark 3.6. *Note that if we consider the universal circuit for bounded-depth circuits with size polynomially bounded by the number of variables, we can further bound the maximum fan-in of multiplication gates in the normal-depth-form. We bound the multiplication fan-in of φ by replacing each $\times$ node with polynomial many $\times$ nodes. This can be achieved since the size of the circuit is polynomially bounded. After this replacement, φ is a bounded-depth algebraic formula with bounded multiplication fan-in of size* $\mathsf{poly}(s)$.

Theorem 3.7 (Existence of bounded-depth universal circuits for polynomials computed by bounded-depth circuits)**.** *Let $\mathbb{F}$ be a field and $\overline{x}$ be n variables, and let $\mathsf{VAC}^0_{s,\Delta}$ denote the class of all polynomials in $\mathbb{F}[\overline{x}]$ that have algebraic circuits of size at most s and depth at most Δ, where Δ is a constant. Then there is a circuit $U(\overline{x},\overline{w}) \in \mathbb{F}[\overline{x},\overline{w}]$ of size $\mathsf{poly}(s)$ and depth $\mathsf{poly}(\Delta)$ such that $\overline{w}$ are $K_{s,d}$ variables that are disjoint from $\overline{x}$, that is universal for $\mathsf{VAC}^0_{s,\Delta}$ in the following sense: if $f(\overline{x}) \in \mathsf{VAC}^0_{s,\Delta}$, then there exists $\overline{a} \in \mathbb{F}^{K_{s,d}}$ such that $U(\overline{x},\overline{a}) = f(\overline{x})$. Notice that given s and d, $K_{s,d}$ can be computed efficiently and is bounded by $\mathsf{poly}(s)$. Also, the universal circuit preserves the maximum multiplication fan-in.*

Proof. Let $\mathbb{F}$ be a field. A polynomial $g \in \mathbb{F}[\overline{x}]$ is computed by an algebraic circuit of size s and depth at most Δ where Δ is a constant. Then, by Lemma 3.5, there exists a bounded-depth algebraic circuit Φ' that computes g such that Φ' is in normal-depth-form of size $\mathsf{poly}(s)$ and depth $2\Delta' - 1$, which is a constant. Let t be the maximum multiplication fan-in in Φ'.

We partition the nodes in Φ' into $2\Delta'$ levels by their depths as follows:

- For every $i \in \{1, \cdots, \Delta' - 1\}$, level $2i$ contains the $\times$ nodes where each of them has a length $2i - 1$ path to the output node.
- For every $i \in \{1. \cdots, \Delta'\}$, level $2i - 1$ contains the $+$ nodes where each of them has a length $2i - 2$ path to the output node.
- Level $2\Delta'$ contains all the leaves.

Now, we construct the universal circuit of depth $2\Delta' - 1$ as follows:

- For every $+$ node in level $2i - 1$ ($i \in \{1, \cdots, \Delta'\}$), the children of it are all the nodes in level $2i$. There is only one $+$ node in level 1 as the output node.
- For every $\times$ node in level $2i$ ($i \in \{1. \cdots, \Delta' - 1\}$), the children of it are $+$ nodes in level $2i + 1$. All the $\times$ nodes in level $2i$ ($i \in \{1, \cdots, \Delta' - 1\}$) are partitioned into t many groups, where $\times$ nodes in group j ($1 \le j \le t$) has j many children in level $2i$. Also, all $\times$ nodes in the same group have distinct children.

Since the out-degree of every $+$ node in Φ' is at most 1, it is sufficient to have $\mathsf{Size}(\Phi')/k$ many $\times$ nodes with in-degree k in the above level. Therefore, there are at most $\mathsf{Size}(\Phi') + \lceil \frac{\mathsf{Size}(\Phi')}{2} \rceil + \lceil \frac{\mathsf{Size}(\Phi')}{3} \rceil + \cdots + \lceil \frac{\mathsf{Size}(\Phi')}{t} \rceil \le \mathsf{Size}(\Phi')^2$ many $\times$ nodes in the above level. Therefore, there are $O(\mathsf{Size}(\Phi')^3)$ many edges between $+$ nodes in level $2i$ and $\times$ nodes in level $2i + 1$.

Hence, we get a depth $2\Delta' - 1$ universal circuit of size $\mathsf{poly}(\mathsf{Size}(\Phi')) = \mathsf{poly}(s)$. Also, note that such universal circuit has the same maximum multiplication fan-in as Φ'. $\square$

4 Extracting Coefficients and IPS Refutation Formula

Let $f(\overline{x},\overline{w}) \in \mathbb{F}[\overline{x},\overline{w}]$ be a polynomial, and let $M = \prod_{i\in I} x_i^{\alpha_i} \cdot \prod_{j\in J} w_j^{\beta_j}$ be a monomial in $f(\overline{x},\overline{w})$, for some $\alpha_i, \beta_i \in \mathbb{N}$ (where $0 \in \mathbb{N}$). Then, we call $\Sigma_{i\in I}\alpha_i$ the $\overline{x}$-degree of M.

Definition 4.1 ($\mathsf{coeff}_M(\cdot)$ [ST25])**.** *Let $f(\overline{x},\overline{w})$ be a polynomial in $\mathbb{F}[\overline{x},\overline{w}]$ in the disjoint sets of variables $\overline{x},\overline{w}$. Let M be an $\overline{x}$-monomial of degree j. Then, $\mathsf{coeff}_M(f(\overline{x},\overline{w}))$ is the (polynomial) coefficient in $\mathbb{F}[\overline{w}]$ (that is, in the $\overline{w}$-variables only) of M in $f(\overline{x},\overline{w})$.*

Note that $f(\overline{x},\overline{w}) = \Sigma_{M_i} M_i \cdot \mathsf{coeff}_{M_i}(f(\overline{x},\overline{w}))$, where the M_i's are all possible $\overline{x}$-monomials of degree at most d, for d the maximal $\overline{x}$-degree of a monomial in $f(\overline{x},\overline{w})$.

Proposition 4.2 (Computation of coefficients in general circuits [ST25])**.** *Let $f(\overline{x},\overline{w}) \in \mathbb{F}[\overline{x},\overline{w}]$ be a polynomial in $\mathbb{F}[\overline{x},\overline{w}]$ in the disjoint sets of variables $\overline{x},\overline{w}$. Suppose that M is an $\overline{x}$-monomial of degree d, and assume that there is an algebraic circuit computing $f(\overline{x},\overline{w})$ of size s and syntactic-degree l. Then, there is a circuit of size $O(7^d \cdot s)$ computing $\mathsf{coeff}_M(f(\overline{x},\overline{w}))$ of syntactic-degree $l^{O(1)}$.*

While [ST25] gave the above proposition about the computation of the coefficient of an $\overline{x}$-monomial in general algebraic circuits, we present the computation of the coefficient of an $\overline{x}$-monomial in *bounded-depth* circuits.

Since we can decrease the maximum multiplication fan-in to n in each polynomial-size bounded-depth circuit with at most a polynomial-size blow up and depth blowing up by at most a constant factor, we can assume that the maximum multiplication fan-in in each polynomial-size bounded-depth circuit is n.

Proposition 4.3 (Computation of coefficients in bounded-depth circuits)**.** *Let $f(\overline{x},\overline{w}) \in \mathbb{F}[\overline{x},\overline{w}]$ be a polynomial in $\mathbb{F}[\overline{x},\overline{w}]$ in the disjoint sets of variables $\overline{x},\overline{w}$. Suppose that M is an $\overline{x}$-monomial of degree d, and assume that there is an algebraic circuit $C(\overline{x},\overline{w})$ computing $f(\overline{x},\overline{w})$ of maximum multiplication fan-in t, size s, syntactic-degree l and depth Δ where Δ is a constant such that*

1. *All edges from the leaves are to + nodes.*
2. *All output-nodes are + nodes.*
3. *The nodes of G are alternating. That is, if v is a + node and (u,v) is an edge, then u is a $\times$ node, and if v is a $\times$ node and (u,v) is an edge then u is either a leaf or a + node.*

Then, there is a bounded-depth algebraic circuit of depth Δ, size $O(2^{(t+d)d} \cdot s)$ computing $\mathsf{coeff}_M(f(\overline{x},\overline{w}))$ of syntactic-degree $l^{O(1)}$.

Proof. For a variable x_i, we show how to construct a circuit, which is in the same depth as $C(\overline{x},\overline{w})$, computing a polynomial $g(\overline{x},\overline{w})$ such that $f = x_i \cdot g + h$ with h having no occurrences of x_i. Then, using d such iterations for each of the d variables in M, we shall get the circuit D that computes the coefficient of M in $f(\overline{x},\overline{w})$ of the same depth. Then, by assigning zeros to all $\overline{x}$-variables in D, we can eliminate all the monomials in D in both $\overline{x}$ and $\overline{w}$ variables. Now, we prove the following claim. The following claim shows how to construct the circuit that extracts the coefficient of a single variable. To construct the circuit that extracts the coefficient of a monomial of degree at most d, we apply d iterations of the following claim. We denote by $C(\overline{x},\overline{w}) \restriction_{x_i=0}$ the polynomial $C(\overline{x},\overline{w})$ where x_i is assigned 0.

Claim 4.4. *Let $C(\overline{x},\overline{w})$ be an algebraic circuit over the field $\mathbb{F}$ of maximum addition fan-in t_1, maximum multiplication fan-in t_2, syntactic-degree l and depth Δ such that*

1. *All edges from the leaves are to + nodes.*
2. *The output node is a + node.*

3. The nodes of G are alternating. That is, if v is a $+$ node and (u, v) is an edge, then u is a $\times$ node, and if v is a $\times$ node and (u, v) is an edge then u is either a leaf or a $+$ node.

Then, for every variables x_i, there is a depth-Δ circuit in the same form as C (i.e., with Item 1-Item 3 holding) of maximum addition fan-in $t_1 \cdot (2^{t_2} - 1)$, maximum multiplication fan-in $t_2 + 1$, size $O(2^{t_2}|C|)$ and syntactic-degree $l^{O(1)}$ that computes the polynomial $g(\overline{x}, \overline{w})$, such that $C(\overline{x}) = x_i \cdot g(\overline{x}, \overline{w}) + C(\overline{x}, \overline{w}) \restriction_{x_i=0}$.

Proof of claim. The proof is obtained by induction on circuit size. Denote by p the polynomial computed by C and for every gate v in C denote by p_v the polynomial computed at gate v.

Denote by $P_{x_i}(p_v)$ the unique polynomial such that $p_v = x_i \cdot P_{x_i}(p_v) + p_v \restriction_{x_i=0}$. For a $\times$ gate v with fan-in t in C, we add at most 2^t new gates. Each gate v itself is duplicated twice so that the first duplicate computes $P_{x_i}(p_v)$ and the second duplicate computes $p_v \restriction_{x_i=0}$.

Base case:

Case 1: $p_v = x_i$. Then, $P_{x_i}(p_v) := 1$ and $p_v \restriction_{x_i=0} := 0$.

Case 2: $p_v = x_j$, for $j \neq i$. Then, $P_{x_i}(p_v) := 0$ and $p_v \restriction_{x_i=0} := x_j$.

Case 3: $p_v = \alpha$, for $\alpha \in \mathbb{F}$. Then, $P_{x_i}(p_v) = 0$ and $p_v \restriction_{x_i=0} = \alpha$.

Induction step:

Case 1: $p_v = \Sigma_{j=1}^{t_1} u_j$. Then, $P_{x_i}(p_v) = \Sigma_{j=1}^{t_1} P_{x_i}(p_{u_j})$ and $p_v \restriction_{x_i=0} = \Sigma_{j=1}^{t_1} p_{u_j} \restriction_{x_i=0}$.

Case 2: $p_v = \prod_{j=1}^{t_2} u_j$. Then,

$$P_{x_i}(p_v) = \sum_{j=1}^{t_2} \left(\sum_{\substack{S \subseteq [t_2] \\ |S|=k}} \prod_{k \in S} P_{x_i}(p_{u_k}) \prod_{l \notin S} p_{u_l} \restriction_{x_i=0} \right) x_i^{j-1}$$

which is equivalent to $\prod_{j=1}^{t_2}(x_i P_{x_i}(p_{u_j}) + p_{u_j} \restriction_{x_i=0}) - \prod_{j=1}^{t_2} p_{u_j} \restriction_{x_i=0}$ "divided by x_i", namely when we decrease by 1 the power of every x_i^b in every monomial in this polynomial (noting that x_i appears with a positive power $b \geq 1$ in every monomial), and

$$p_v \restriction_{x_i=0} = \prod_{j=1}^{t_2} p_{u_j} \restriction_{x_i=0} .$$

Note that by expanding brackets, $P_{x_i}(p_v)$ is written explicitly with $2^{t_2} - 1$ many terms as depth-2 circuits that have one $+$ node at the top and $2^{t_2} - 1$ many $\times$ nodes as children.

Then, by merging the $+$ node in $P_{x_i}(p_v) = \Sigma_{j=1}^{t_1} P_{x_i}(p_{u_j})$ and $P_{x_i}(p_v) = \sum_{j=1}^{t_2} \left(\sum_{\substack{S \subseteq [t_2] \\ |S|=k}} \prod_{k \in S} P_{x_i}(p_{u_k}) \prod_{l \notin S} p_{u_l} \restriction_{x_i=0} \right) x_i^{j-1}$ (Since this formula is written explicitly and the circuit is alternating, which means there is always a $+$ node above any $\times$ node), our circuit has the same depth as C and is in the following form:

1. All edges from the leaves are to $+$ nodes.

2. The output nodes is a $+$ node.

3. The nodes of G are alternating. That is, if v is a $+$ node and (u, v) is an edge, then u is a $\times$ node, and if v is a $\times$ node and (u, v) is an edge then u is either a leaf or a $+$ node.

Moreover, by computing the $x_i^2, x_i^3, \cdots, x_i^{t_2}$ at the bottom of the circuit using a trivial depth-2 circuit with maximum multiplication fan-in t_2, the maximum addition fan-in is $t_1 \cdot (2^{t_2}-1)$ and the maximum multiplication fan-in is t_2+1. The size of the circuit after one iteration is $2 \cdot (2^{t_2}-1) \cdot |C|$. $\square_{\text{claim}}$

As we showed above, we can construct a circuit of the same depth as C that extracts the coefficient of a single variable in size $2 \cdot (2^{t_2}-1) \cdot |C|$. To construct a circuit of the same depth as C that extracts the coefficient of a monomial of degree at most d, we just need to do d iterations of the above claim. Therefore, after d iterations, the size of the circuits after the last iteration is $|C| \cdot \prod_{i=0}^{d-1}(2^{t_2+i}-1) \cdot 2^d = O(2^{(t_2+d)d}|C|)$ and the syntactic-degree is $l^{O(1)}$. This concludes the proof of Proposition 4.3. $\square$

Definition 4.5 (Bounded-depth IPS refutation predicate $\mathsf{IPS}_{\mathsf{ref}}(s, \Delta, l, \overline{\mathcal{F}})$)**.** *Let $\overline{\mathcal{F}}$ be a* CNF *formula with m clauses and n variables $\overline{x}$ written as a set of polynomial equations according to Definition 2.26. Let $U(\overline{x}, \overline{y}, \overline{w})$ be the bounded-depth universal circuit for depth Δ and size s circuits in the $\overline{x}$ variables and the m placeholder variables $\overline{y}$, and the $K_{s,\Delta}$ edge label variables $\overline{w}$. We formalize the existence of a size s, depth Δ circuit that computes the* IPS *refutation of $\overline{\mathcal{F}}$ in degree at most l, denoted* $\mathsf{IPS}_{\mathsf{ref}}(s, \Delta, l, \overline{\mathcal{F}})$*, with the following set of circuit equations (in the $\overline{w}$ variables only):*

$$\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{0}, \overline{w})) = 0$$

$$\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{\mathcal{F}}, \overline{w})) = \begin{cases} 1, & M_i = 1 \text{ (i.e., \textit{the constant 1 monomial})}; \\ 0, & \textit{otherwise}, \end{cases}$$

where $i \in [N]$ so that $\{M_i\}_{i=1}^N$ are the set of all possible $\overline{x}$-monomials of degree at most l, and $N = \sum_{j=0}^{l} \binom{n+j-1}{j} = 2^{O(n+l)}$ is the number of monomials of total degree at most l over n variables, and $\overline{0}$ is the all-zero vector of length m.

The size of $\mathsf{IPS}_{\mathsf{ref}}(s, \Delta, l, \overline{\mathcal{F}})$ *is $O(2^{(t+l)l} \cdot |U(\overline{x}, \overline{y}, \overline{w})| \cdot |\overline{\mathcal{F}}| \cdot N)$ where t is maximum multiplication fan-in in $U(\overline{x}, \overline{y}, \overline{w})$.*

Definition 4.6 (Formalization of $\mathsf{VNP} = \mathsf{VAC}^0$)**.** *The formalization of* $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ *denoted "*$\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$*", expressing that there is a bounded-depth universal circuit for size s and depth Δ circuits that compute the Permanent polynomial of dimension n (with $\overline{x}$ being the n^2 variables of the Permanent), is the following set of polynomial equations (in the $\overline{w}$-variables only):*

$$\{\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) = b_i : 1 \le i \le N\},$$

where $\overline{b} = \mathsf{coeff}(\mathrm{perm}(\overline{x})) \in \mathbb{F}^N$ is the coefficient vector of the polynomial $\mathrm{perm}(\overline{x})$ of dimension n, $U(\overline{x}, \overline{w})$ is the constant-depth universal circuit for polynomials of depth at most Δ and have circuits of size at most s, $\overline{w}$ are the $K_{s,\Delta}$ edge variables, $\{M_i\}_{i=1}^N$ is the set of all possible $\overline{x}$-monomials of degree at most l, and $N = \Sigma_{j=0}^{l} \binom{n^2+j-1}{j} = 2^{O(n^2+l)}$ is the number of monomials of total degree at most l over n^2 variables. Then, the size of the above set of polynomial equations is $O(2^{(t+l)l} \cdot |U(\overline{x}, \overline{w})| \cdot N)$ where t is maximum multiplication fan-in in $U(\overline{x}, \overline{w})$.

5 No Short $\mathsf{AC}^0[p]$-Frege Proofs for Diagonalizing DNF Formulas

This section presents our main result. we show unconditionally that constant-depth IPS cannot efficiently refute certain constant-depth IPS upper bounds. As a corollary, we obtain the same result for $\mathsf{AC}^0[p]$-Frege, since this proof system is simulated by constant-depth IPS over $\mathbb{F}_p$ (Theorem 5.6). More precisely, we prove that constant-depth IPS does not admit polynomial-size refutations of the diagonalizing CNF formulas $\Phi_{t,l,\Delta',n,s,\Delta}$, infinitely often. These formulas express the existence of size-t, depth-Δ' IPS refutation of the statement that the Permanent polynomial is computable by size-s, depth-Δ algebraic circuits.

In addition, this section contains the proof of Theorem 1.6, which shows that ruling out short proofs for the diagonalizing formulas is a necessary step towards constant-depth IPS lower bounds. We begin by introducing the formulas that will be used throughout the argument.

Let $N = \sum_{j=0}^{l} \binom{n^2+j-1}{j} = 2^{O(n^2+l)}$ be the number of monomials of total degree at most l over n^2 variables. We shall work over a finite field $\mathbb{F}_q$ here to enable the encoding of circuit equations as CNF formulas.

- $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$: circuit equations expressing that there is an algebraic circuit of size s and depth Δ that *agrees* with the Permanent polynomial of dimension n on all the coefficients of monomials of degree at most l (i.e., every monomial M computed by the algebraic circuit has the same coefficient as in the Permanent polynomial).
  - Type: circuit equations;
  - Number of variable: $K_{s,\Delta} = \mathsf{poly}(s, \Delta)$;
  - Size: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\varphi^{\mathsf{cnf}}_{n,s,l,\Delta}$: the CNF encoding of $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ based on Definition 2.29.
  - Type: CNF formula;
  - Number of variable: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$;
  - Size: $O(q \cdot 2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$: the SCNF encoding of $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ based on Definition 2.34 together with the field axioms $(x^q - x = 0)$ for all variables.
  - Type: SCNF formula;
  - Number of variable: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \overline{\mathcal{F}})$: circuit equations expressing that there exists an algebraic circuit for size t and depth Δ that agrees with the IPS refutation of $\overline{\mathcal{F}}$ on all the coefficients of monomials of degree at most l.
  - Type: circuit equations;
  - Number of variable: $K_{t,\Delta} = \mathsf{poly}(t, \Delta)$;
  - Size: $O(2^{(n+l)l} \cdot \mathsf{poly}(t, \Delta) \cdot |\overline{\mathcal{F}}| \cdot N)$.
- $\Phi_{t,l,\Delta',n,s,\Delta}$: *The diagonalizing CNF formula.* More precisely, the CNF encoding of $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta', l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$ expressing that there is an algebraic circuit (the purported IPS refutation) of size t and depth Δ' that agrees with the IPS refutation of $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$ on all the coefficients of monomials of degree at most l.
  - Type: CNF formulas;
  - Number of variable: $K_{t,\Delta}$ which is $\mathsf{poly}(t, \Delta)$;
  - Size: $O(q \cdot 2^{(n+l)l} \cdot \mathsf{poly}(t, \Delta) \cdot |\mathcal{F}| \cdot N)$.

The following lemma easily follows from Lemma 2.25.

Lemma 5.1. *Every family of unsatisfiable formulas (φ_n), which contains a set of unsatisfiable* CNF *formulas, has a family of* $\mathsf{IPS}^{\mathsf{alg}}$ *(also* IPS*) certificates (C_n) in* $\mathsf{VNP}_{\mathbb{F}}$*.*

Since Semi-CNFs are substitution instances of CNFs, we get the following lemma by substituting the occurrence of Boolean variables with their corresponding UBIT.

Lemma 5.2. $\{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}\}_n$ *is a family of unsatisfiable* SCNF *formulas and has a family of* $\mathsf{IPS}^{\mathsf{alg}}$ *(also* IPS*) certificates (C_n) in* $\mathsf{VNP}_{\mathbb{F}}$*.*

We fix $l : \mathbb{N} \to \mathbb{N}$ to be a (monotone) size function $l(r) = \lceil r^{\epsilon} \rceil$ for some constant ϵ.

The main result of this section is the following.

Theorem 5.3 (Main; no short proofs over fixed finite field)**.** *Let q be a constant prime. The* CNF *family $\{\Phi_{t,l,\Delta',n,s,\Delta}\}$ does not have constant-depth polynomial-size* IPS *refutations infinitely often over $\mathbb{F}_q$, in the following sense: for every constant Δ there exist constants c_1 and Δ', such that for every sufficiently large constant c_2 and every constants Δ'' and c_0, for infinitely many $n, t(n), s(n) \in \mathbb{N}$, such that $t(n) > |\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}|^{c_1}$ and $n^{c_1} < s(n) < n^{c_2}$, $\Phi_{t,l,\Delta',n,s,\Delta}$ has no* IPS *refutation of size at most $|\Phi_{t,l,\Delta',n,s,\Delta}|^{c_0}$ and depth at most Δ''.*

Before proving this theorem, we provide an overview of its proof.

Proof overview: By way of contradiction, we assume that there exists a constant Δ such that for every constant Δ', for every sufficiently large n, and for every $t(n), s(n) \in \mathbb{N}$, such that $t(n) > |\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}|^{c_1}$ and $n^{c_1} < s(n) < n^{c_2}$, $\Phi_{t,l,\Delta',n,s,\Delta} := \mathsf{cnf}(\mathsf{IPS}_{\mathsf{ref}}(t, \Delta', l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ has a small depth-$O(1)$ refutation. Then, by substituting the occurrences of Boolean variables with the correspondence UBITs, we can assume that $\mathsf{scnf}(\mathsf{IPS}_{\mathsf{ref}}(t, \Delta', l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ has a small and depth-$O(\Delta')$ refutation.

1. By applying Lemma 2.36, from $\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ where m, t, l are parameters, that meet the conditions in Theorem 5.3, we can derive the circuit equations

$$\mathsf{VNP} = \mathsf{VAC}^0(m, t, l, \Delta),$$

in $O(\Delta)$ depth and polynomial-size IPS. Recall that $\mathsf{VNP} = \mathsf{VAC}^0(m, t, l, \Delta)$ is the set of circuit equations expressing that there is an algebraic circuit of size t and depth Δ that agrees with the Permanent polynomial of dimension m on all the coefficients of monomials of degree at most l.

2. In Lemma 5.4 we prove (the contrapositive of) the bounded-depth version of Theorem 2.24 *within bounded-depth* IPS. The contrapositive of the bounded-depth version of Theorem 2.24 expresses that if the Permanent polynomial can be computed by bounded-depth polynomial-size circuits, then bounded-depth IPS can efficiently refute any family of CNF formulas. To be more specific, from $\mathsf{VNP} = \mathsf{VAC}^0(m, t, l, \Delta)$ we derive $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$ in depth $O(\Delta)$ and polynomial size. Note that $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$ is the set of circuit equations expressing that IPS can refute $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$ in size t and depth Δ.

3. Applying Lemma 2.35, we can derive $\mathsf{scnf}(\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ from $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$ in depth $O(\Delta)$ polynomial size IPS. Since we assumed that for any constant Δ', $\mathsf{scnf}(\mathsf{IPS}_{\mathsf{ref}}(t, \Delta', l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ has a small and bounded-depth refutation, it follows that $\mathsf{scnf}(\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ has a small and bounded-depth refutation, particularly when we take $\Delta' = \Delta$.

Hence, we get a small and bounded-depth refutation of $\varphi^*_{m,t,l,\Delta}$, as follows:

$$\begin{aligned}
\varphi^*_{m,t,l,\Delta} &\left|\frac{{}^{*,O(\Delta)}}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VNP} = \mathsf{VAC}^0(m,t,l,\Delta) \qquad \text{Lemma 2.36 (see Item 1)}\\
&\left|\frac{{}^{*,O(\Delta)}}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{IPS}_{\mathsf{ref}}(t,\Delta,l,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}) \qquad \text{Lemma 5.4 (see Item 2)}\\
&\left|\frac{{}^{*,O(\Delta)}}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{scnf}(\mathsf{IPS}_{\mathsf{ref}}(t,\Delta,l,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})) \qquad \text{Lemma 2.35 (see Item 3)}\\
&\left|\frac{{}^{*,O(\Delta)}}{\mathsf{IPS}^{\mathsf{alg}}}\right. 1 = 0 \qquad \text{by assumption.}
\end{aligned}$$

Then we know that for some constant Δ''' and large enough w, the system of circuit equations $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,\Delta''',l,\varphi^*_{m,t,l,\Delta})$ is satisfiable. Hence, by Proposition 2.33, $\mathsf{cnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,\Delta''',l,\varphi^*_{m,t,l,\Delta}))$ is satisfiable.

By assumption, for any constant Δ', for all sufficiently large n, for all proper $t(n), s(n) \in \mathbb{N}$, $\Phi_{t,l,\Delta',n,s,\Delta} := \mathsf{cnf}(\mathsf{IPS}_{\mathsf{ref}}(t,\Delta',l,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ has a small and depth-$O(1)$ refutation. Taking $t = w$, $\Delta' = \Delta'''$, $n = m$, $s = t$, we know that $\mathsf{cnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,\Delta''',l,\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}))$ is refutable which implies it is not satisfiable. This is a contradiction.

Proof of Theorem 5.3. For the sake of contradiction, we assume that there exists a constant Δ for all constants c_1, Δ' such that there exist constants c_2, Δ'' and c_0 for all $n, t(n), s(n) \in \mathbb{N}$ such that $t(n) > |\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}|^{c_1}$ and $n^{c_1} < s(n) < n^{c_2}$,

$$\underbrace{\mathsf{cnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta',\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))}_{\lambda} \left|\frac{{}^{|\lambda|^{c_0},\Delta''}}{\mathsf{IPS}}\right. 1 = 0. \tag{12}$$

By substituting the occurrence of Boolean variables with their corresponding UBITs, we get that there exists a constant Δ for all constants c_1, Δ' such that there exist constants c_2, Δ'' and c_0 for all $n, t(n), s(n) \in \mathbb{N}$, if $t(n) > |\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}|^{c_1}$ and $n^{c_1} < s(n) < n^{c_2}$,

$$\underbrace{\mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta',\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))}_{\lambda} \left|\frac{{}^{|\lambda|^{c_0},\Delta''}}{\mathsf{IPS}^{\mathsf{alg}}}\right. 1 = 0. \tag{13}$$

Since the Boolean axioms of each UBIT can be easily derived, as shown in the proof of Lemma 2.35, IPS can be replaced by $\mathsf{IPS}^{\mathsf{alg}}$ here.

We take $\Delta' = \Delta$, which gives us the following:

$$\underbrace{\mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))}_{\lambda} \left|\frac{{}^{|\lambda|^{c_0},\Delta''}}{\mathsf{IPS}^{\mathsf{alg}}}\right. 1 = 0. \tag{14}$$

Let $m = 6(L' + P')$ where L' is the number of variables in $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$ and P' is the number of placeholders needed for axioms in $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$. Let $\gamma := \varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ be the Semi-CNF formulas $\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ together with the field axioms for the variables in $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$.

By Lemma 5.4, there exist constants c_3 and Δ''' such that

$$\overbrace{\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}}^{\gamma} \left|\frac{{}^{|\gamma|^{c_3},\Delta'''}}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})).$$

Combining the above equation with the assumption, we know that

$$\varphi^{\mathsf{scnf}}_{m,t,l,\Delta} \Big|\frac{|\gamma|^{c_3},\Delta'''}{\mathsf{IPS}^{\mathsf{alg}}} \underbrace{\mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))}_{\lambda} \Big|\frac{|\lambda|^{c_0},\Delta''}{\mathsf{IPS}^{\mathsf{alg}}} 1=0. \tag{15}$$

Since $|\lambda|$ is polynomially bounded by $|\gamma|$, there exists a constant c_1 such that

$$|\gamma|^{c_3} + |\lambda|^{c_0} < |\gamma|^{c_1}.$$

We pick a big enough c_1 such that

$$\underbrace{\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}}_{\gamma} \Big|\frac{|\gamma|^{c_1},\Delta'''+\Delta''}{\mathsf{IPS}^{\mathsf{alg}}} 1=0.$$

From the above equation, we can conclude that $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,l,\Delta'''+\Delta'',\varphi^{\mathsf{scnf}}_{m,t,l,\Delta})$ is satisfiable for some $w \geq |\gamma|^{c_1}$ over $\mathbb{F}_q$ and is polynomially bounded by $|\gamma|$. Hence, by Proposition 2.33, $\mathsf{cnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,l,\Delta'''+\Delta'',\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}))$ is also satisfiable over $\mathbb{F}_q$. However, by our assumption, when we take $\Delta' = \Delta''' + \Delta''$, we know that for all big enough w and t, $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,l,\Delta'''+\Delta'',\varphi^{\mathsf{scnf}}_{m,t,l,\Delta})$ is refutable. By the soundness of IPS, $\mathsf{cnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(w,l,\Delta'''+\Delta'',\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}))$ should be unsatisfiable which is a contradiction. ☐

It remains to prove the following.

Lemma 5.4 (Constant-depth version of Grochow-Pitassi formalization in $\mathsf{IPS}^{\mathsf{alg}}$)**.** *There are constants c_3 and Δ''' such that under the above notation and parameters:*

$$\overbrace{\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}}^{\gamma} \Big|\frac{|\gamma|^{c_3},\Delta'''}{\mathsf{IPS}^{\mathsf{alg}}} \mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})).$$

Proof. Recall that $\gamma := \varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ is the Semi-CNF formula $\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ together with field axioms for all variables in it. Let N' be the number of $\overline{x}$-monomials with degree at most l over m variables. According to Lemma 2.36, from $\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$, we can derive the following circuit equations in polynomial-size $\mathsf{IPS}^{\mathsf{alg}}$

$$\{\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w})) = b_i : 1 \leq i \leq N'\},$$

where $\overline{b} = \mathsf{coeff}(\mathrm{perm}(\overline{x})) \in \mathbb{F}_q^{N'}$ is the coefficient vector of the Permanent polynomial $\mathrm{perm}(\overline{x})$, and $U(\overline{x},\overline{w})$ is the universal circuit for algebraic circuits of depth at most Δ and of size at most t. Formally, there exist constants c_4 and Δ_1 such that

$$\lambda \Big|\frac{|\lambda|^{c_4},\Delta_1}{\mathsf{IPS}^{\mathsf{alg}}} \{\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w})) = b_i : 1 \leq i \leq N'\}$$

which is

$$\lambda \Big|\frac{|\lambda|^{c_4},\Delta_1}{\mathsf{IPS}^{\mathsf{alg}}} \mathsf{VNP} = \mathsf{VAC}^0(m,t,l,\Delta).$$

Now, we will show that from $\mathsf{VNP} = \mathsf{VAC}^0(m,t,l,\Delta)$, there is a depth-$O(\Delta)$ polynomial size IPS derivation of $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$.

Claim 5.5. *Suppose M_i is an $\overline{x}$-monomial with degree at most l. Let $\overline{a}$ be any possibly partial substitution of polynomials (including field elements) for the variables $\overline{x}$. Given* $\mathsf{VNP} = \mathsf{VAC}^0(m,t,l,\Delta) := \{\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w})) = b_i : 1 \le i \le N'\}$*, we can derive*

$$\mathsf{coeff}_M(U(\overline{x}\restriction_{\overline{a}},\overline{w})) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x}\restriction_{\overline{a}})) \tag{16}$$

in depth-$O(\Delta)$ polynomial size $\mathsf{IPS}^{\mathsf{alg}}$ *for every $\overline{x}$-monomial M with degree at most l.*

Proof of Claim 5.5. First, it is easy to get the following polynomial identity,

$$U(\overline{x}\restriction_{\overline{a}},\overline{w}) = U(\overline{x},\overline{w})\restriction_{\overline{a}}\ . \tag{17}$$

And by Definition 4.1, we have the following polynomial identity,

$$U(\overline{x},\overline{w})\restriction_{\overline{a}} = \left(\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot M_i\right)\restriction_{\overline{a}}\ . \tag{18}$$

And since $\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))$ only contains $\overline{w}$ variables, we have the following polynomial identity,

$$\left(\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot M_i\right)\restriction_{\overline{a}} = \sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}})\,. \tag{19}$$

Hence, combining three polynomial identities in Equation (17), Equation (18) and Equation (19) above, we have the following polynomial identity,

$$U(\overline{x}\restriction_{\overline{a}},\overline{w}) = \sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}})\,. \tag{20}$$

Let M be any $\overline{x}$-monomial with degree at most l. By the above polynomial identity, we have the following polynomial identity,

$$\mathsf{coeff}_M(U(\overline{x}\restriction_{\overline{a}},\overline{w})) = \mathsf{coeff}_M(\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}}))\,. \tag{21}$$

Since coeff_M is a linear operator, we have the following polynomial identity,

$$\mathsf{coeff}_M(\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}})) = \sum_{i\in[N']} \mathsf{coeff}_M(\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}}))\,. \tag{22}$$

Since $\mathsf{coeff}_M(\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot(M_i\restriction_{\overline{a}})) = \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot\mathsf{coeff}_M(M_i\restriction_{\overline{a}})$, we have the following polynomial identity,

$$\sum_{i\in[N']} \mathsf{coeff}_M(\mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot (M_i\restriction_{\overline{a}})) = \sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot \mathsf{coeff}_M((M_i\restriction_{\overline{a}}))\,. \tag{23}$$

Hence, combining the three polynomial identities in Equation (21), Equation (22) and Equation (23), we have the following polynomial identity,

$$\mathsf{coeff}_M(U(\overline{x}\restriction_{\overline{a}},\overline{w})) = \sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x},\overline{w}))\cdot \mathsf{coeff}_M((M_i\restriction_{\overline{a}})) \tag{24}$$

in depth-$O(\Delta)$ linear-size $\mathsf{IPS}^{\mathsf{alg}}$ (we increased the depth of $U(\overline{x}, \overline{w})$ by a constant factor using Proposition 4.3).

Also, since we have already derived $\{\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) = b_i : 1 \le i \le N'\}$ from λ, by multiplying $\mathsf{coeff}_M(M_i \restriction_{\overline{a}})$ to each circuit equation and adding them, we can derive the following circuit equation in depth-$O(\Delta)$ polynomial-size $\mathsf{IPS}^{\mathsf{alg}}$,

$$\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) \cdot \mathsf{coeff}_M((M_i \restriction_{\overline{a}})) = \sum_{i\in[N']} b_i \cdot \mathsf{coeff}_M(M_i \restriction_{\overline{a}}) . \tag{25}$$

By definition, we have the following polynomial identity,

$$\sum_{i\in[N']} b_i \cdot (M_i \restriction_{\overline{a}}) = \mathrm{perm}(\overline{x} \restriction_{\overline{a}}) . \tag{26}$$

Therefore, for any $\overline{x}$-monomial M with degree at most l, we have the following polynomial identity,

$$\mathsf{coeff}_M(\sum_{i\in[N']} b_i \cdot (M_i \restriction_{\overline{a}})) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x} \restriction_{\overline{a}})) . \tag{27}$$

Also, since b_i are just field elements and linearity of coeff_M,

$$\mathsf{coeff}_M(\sum_{i\in[N']} b_i \cdot (M_i \restriction_{\overline{a}})) = \sum_{i\in[N']} b_i \cdot \mathsf{coeff}_M(M_i \restriction_{\overline{a}}) . \tag{28}$$

Combining the two polynomial identities in Equation (27) and Equation (28) above and using Proposition 2.22 again, we prove the following polynomial identities,

$$\sum_{i\in[N']} b_i \cdot \mathsf{coeff}_M(M_i \restriction_{\overline{a}}) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x} \restriction_{\overline{a}})) \tag{29}$$

in depth-$O(\Delta)$ polynomial size $\mathsf{IPS}^{\mathsf{alg}}$.

Hence, by combining those three circuit equations in Equation (24), Equation (25) and Equation (29),

$$\mathsf{coeff}_M(U(\overline{x} \restriction_{\overline{a}}, \overline{w})) = \sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) \cdot \mathsf{coeff}_M((M_i \restriction_{\overline{a}}))$$
$$\sum_{i\in[N']} \mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) \cdot \mathsf{coeff}_M((M_i \restriction_{\overline{a}})) = \sum_{i\in[N']} b_i \cdot \mathsf{coeff}_M(M_i \restriction_{\overline{a}})$$
$$\sum_{i\in[N']} b_i \cdot \mathsf{coeff}_M(M_i \restriction_{\overline{a}}) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x} \restriction_{\overline{a}})),$$

we can derive

$$\mathsf{coeff}_M(U(\overline{x} \restriction_{\overline{a}}, \overline{w})) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x} \restriction_{\overline{a}})) \tag{30}$$

in depth-$O(\Delta)$ polynomial size $\mathsf{IPS}^{\mathsf{alg}}$. This concludes the proof of Claim 5.5. □

Now, we divide $\overline{x}$ into $\overline{x'}$ and $\overline{y}$ two disjoint parts of variables, where $\overline{y}$ represents the placeholder for axioms and $\overline{x'}$ represents the rest.

By Theorem 2.14, we know that $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$, which contains $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$ that denotes the SCNF encoding of the circuit equation $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ is unsatisfiable. Therefore, by Lemma 5.2, there exists a VNP-$\mathsf{IPS}^{\mathsf{alg}}$ refutation for $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$.

Note that the Permanent polynomial $\mathrm{perm}(\overline{x})$ of dimension $m = 6(L' + P')$ is complete for VNP with $m/6 = L'+P'$ variables which is the number of variables in $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t, l, \Delta, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$ [Val79]. Therefore, there exists an substitution $\overline{\alpha}$ such that $\mathrm{perm}(\overline{x} \restriction_{\overline{\alpha}})$ computes exactly a VNP-$\mathsf{IPS}^{\mathsf{alg}}$ refutation of $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$.

Notice that under substitution $\overline{\alpha}$, $\overline{y}$ must be unassigned because they are the variables representing placeholders which means $\mathrm{perm}(\overline{x} \restriction_{\overline{\alpha}}) = \mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y})$.

We use $\overline{y} \restriction_{\overline{0}}$ to represent that all $\overline{y}$ variables are assigned zero, and $\overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}}$ to represent that each y_i is replaced by the corresponding ith axiom from $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$.

By Claim 5.5, from

$$\mathsf{VNP} = \mathsf{VAC}^0(m, t, l, \Delta),$$

we can derive the following circuit equation,

$$\mathsf{coeff}_M(U(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\overline{0}}, \overline{w})) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\overline{0}})) \tag{31}$$

where $\overline{\alpha}$ is the substitution such that $\mathrm{perm}(\overline{x} \restriction_{\overline{\alpha}})$ computes exactly the VNP-$\mathsf{IPS}^{\mathsf{alg}}$ refutation of $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$.

Therefore, $\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\overline{0}})$ is the $\mathsf{IPS}^{\mathsf{alg}}$ refutation of $\varphi^{\mathsf{scnf}}_{n,s,l,d}$ with placeholder variables replaced with all zero. By the definition of $\mathsf{IPS}^{\mathsf{alg}}$, we have the following polynomial identity,

$$\mathsf{coeff}_M(\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\overline{0}})) = 0\,. \tag{32}$$

Combining Equation (31) and Equation (32), we have

$$\mathsf{coeff}_M(U(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\overline{0}}, \overline{w})) = 0\,. \tag{33}$$

Again, by Claim 5.5, we have

$$\mathsf{coeff}_M(U(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}}, \overline{w})) = \mathsf{coeff}_M(\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}}))\,. \tag{34}$$

Note that $\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}})$ is the $\mathsf{IPS}^{\mathsf{alg}}$ refutation of $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$ with placeholder variables replaced with all axioms in $\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}$. By the definition of $\mathsf{IPS}^{\mathsf{alg}}$, we have the following polynomial identity,

$$\mathsf{coeff}_M(\mathrm{perm}(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}})) = \begin{cases} 1, & M_i = 1 \ (i.e., \text{the constant 1 monomial}); \\ 0, & \text{otherwise}, \end{cases} \,. \tag{35}$$

Combining Equation (34) and Equation (35), we have

$$\mathsf{coeff}_M(U(\overline{x'} \restriction_{\overline{\alpha}}, \overline{y} \restriction_{\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}}, \overline{w})) = \begin{cases} 1, & M_i = 1 \ (i.e., \text{the constant 1 monomial}); \\ 0, & \text{otherwise}. \end{cases} \tag{36}$$

Note that Equation (33) and Equation (36) are exactly the circuit equations in $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t, l, \Delta, \varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$. Also, note that we can prove Equation (33) and Equation (36) for every monomial M in parallel, and there are only a constant many polynomial identities and derivations in the proof for each M. Moreover, the depth of each polynomial identity and derivation is bounded by $O(\Delta)$. Hence, we can conclude that there exist constants c_5 and $\Delta_2 = O(\Delta)$ such that

$$\underbrace{\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}}_{\gamma} \,\Big|\frac{|\gamma|^{c_5},\Delta_2}{\mathsf{IPS}^{\mathsf{alg}}}\; \mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})\,. \tag{37}$$

Note that $\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}$ already includes all the field axioms of variables in $\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta})$. By Lemma 2.35, we can derive $\mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))$ in depth-$O(\Delta)$ polynomial-size $\mathsf{IPS}^{\mathsf{alg}}$.

We can conclude that there exist constants c_3 and Δ''' such that

$$\underbrace{\varphi^{\mathsf{scnf}}_{m,t,l,\Delta}}_{\gamma} \,\Big|\frac{|\gamma|^{c_3},\Delta'''}{\mathsf{IPS}^{\mathsf{alg}}}\; \mathsf{scnf}(\mathsf{IPS}^{\mathsf{alg}}_{\mathsf{ref}}(t,l,\Delta,\varphi^{\mathsf{scnf}}_{n,s,l,\Delta}))\,. \tag{38}$$

□

Theorem 1.2 in the introduction which uses a fixed prime field of size p follows from Theorem 5.3 by setting $\psi_{d,d',n} = \Phi_{t,l,\Delta',n,s,\Delta}$, where $d = \Delta, d' = \Delta'$ and s and t are chosen to be appropriate polynomially bounded functions as in the statement of Theorem 5.3.

To get the no-short proof result against $\mathsf{AC}^0[p]$-Frege we use the following simulation:

Theorem 5.6 (Depth-preserving simulation of Frege systems by the Ideal proof system [GP18])**.** *Let p be prime and $\mathbb{F}$ any field of characteristic p. Then $\mathsf{IPS}_{\mathbb{F}}$ p-simulates $\mathsf{AC}^0[p]$-Frege in such a way that depth-d $\mathsf{AC}^0[p]$-Frege proofs are simulated by depth-$O(d)$ $\mathsf{IPS}_{\mathbb{F}}$ proofs. In particular, $\mathsf{AC}^0[p]$-Frege is p-simulated by bounded-depth $\mathsf{IPS}_{\mathbb{F}}$.*

Therefore, a corollary of Theorem 5.3 is (see the argument in Section 1.3 that comes after Theorem 1.2):

Corollary 5.7. *For every prime p there is an explicit sequence $\{\phi_n\}$ of DNF formulas (of unknown validity) such that there are no polynomial-size $\mathsf{AC}^0[p]$-Frege proofs of $\{\phi_n\}$.*

5.1 Ruling Out Easiness for Diagonalizing CNFs is Necessary

We now turn to establishing Theorem 1.6. We first define the notion of a "reasonable" circuit class.

Definition 5.8. *We say $\mathcal{C}$ is a* reasonable *algebraic circuit class if $\mathcal{C}$-IPS can efficiently prove the $\mathcal{C}$-IPS analogue of Lemma 5.4, i.e., that $\mathcal{C}$ lower bounds for Permanent follow from $\mathcal{C}$-IPS lower bounds for CNF formulas.*

The main result of [ST25] can be seen as showing that the class of general algebraic circuits is reasonable, and Theorem 5.3 shows that the class of constant-depth algebraic circuits is reasonable as well. The proof of Theorem 5.3 establishes that every natural algebraic class intermediate in power between constant-depth circuits and general circuits is reasonable as well.

We observe that Lemma 2.25 can be generalised for an arbitrary algebraic circuit class $\mathcal{C}$.

Theorem 5.9 (Grochow-Pitassi for $\mathcal{C}$)**.** *Let $\mathcal{C}$ be any algebraic circuit class. For any field $\mathbb{F}$, if $\mathcal{C}$-IPS is not p-bounded, namely there exists a super-polynomial lower bound on algebraic $\mathcal{C}$-IPS refutations (hence also on $\mathcal{C}$-IPS refutations) over $\mathbb{F}$ for a family of unsatisfiable CNF formulas $\{\phi_n\}_n$, then $\mathsf{VNP}_{\mathbb{F}} \neq \mathcal{C}_{\mathbb{F}}$.*

Proof. By Lemma 2.25, any family of unsatisfiable CNF formulas $\{\phi_n\}_n$ has an IPS refutation that is computable in VNP. However, since $\mathcal{C}$-IPS is not p-bounded, there exists a family of unsatisfiable CNF formulas $\{\phi_n\}_n$ that require super-polynomial size $\mathcal{C}$-IPS proofs. Hence no IPS refutation for the family $\{\phi_n\}_n$ is in $\mathcal{C}$. Thus we have that the VNP refutation is not in $\mathcal{C}$, and hence that $\mathsf{VNP} \neq \mathcal{C}$. □

Now, we show that the $\mathcal{C}$-analogue of Theorem 1.2 is necessary to prove $\mathcal{C}$-IPS lower bounds for unsatisfiable CNFs, when $\mathcal{C}$ is a reasonable algebraic circuit class.

We use $\{\Phi^{\mathcal{C}}_n\}$ to denote the $\mathcal{C}$ analogue of $\{\Phi_{t,l,\Delta',n,s,\Delta}\}$ from Theorem 5.3.

Theorem 5.10 (Necessity of the main theorem)**.** *Let $\mathcal{C}$ be any reasonable algebraic circuit class. Let p be a sequence of primes, and $\mathbb{F}_p$ be the prime field. If $\mathcal{C}$-IPS is not p-bounded over $\mathbb{F}_p$, which means there is a super-polynomial lower bound on algebraic $\mathcal{C}$-IPS refutations (hence also on $\mathcal{C}$-IPS refutations) over $\mathbb{F}_p$ for a family of unsatisfiable CNF formulas $\{\phi_n\}_n$, then the CNF family $\{\Phi^{\mathcal{C}}_n\}$ does not have polynomial-size $\mathcal{C}$-IPS refutations infinitely often over $\mathbb{F}_p$ in the following sense: there exists a constant c_1 such that for every sufficiently large constant c_2 and every constant c_0, for infinitely many $n, t(n), s(n) \in \mathbb{N}$, $t(n) > 2^{(n^{c_1})}$ and $n^{c_1} < s(n) < n^{c_2}$, $\Phi^{\mathcal{C}}_n$ has no $\mathcal{C}$-IPS refutation over $\mathbb{F}_p$ of size at most $|\Phi^{\mathcal{C}}_n|^{c_0}$.*

Proof. Our definition of a reasonable algebraic circuit class $\mathcal{C}$ abstracts out the properties of $\mathcal{C}$ required to prove an *implication* from $\mathsf{VNP} \neq \mathcal{C}$ to super-polynomial lower bounds on $\Phi^{\mathcal{C}}_n$ against $\mathcal{C}$-IPS, using essentially the same proof as for Theorem 5.3. By Theorem 5.9, super-polynomial $\mathcal{C}$-IPS lower bounds for any sequence $\{\phi_n\}_n$ of unsatisfiable CNFs implies $\mathsf{VNP} \neq \mathcal{C}$. The desired result follows from combining these two implications. □

Theorem 5.10 is the formal version of Theorem 1.6 in the Introduction.

6 Supporting Evidence for the Diagonalizing CNFs as Unsatisfiable

In this section, we present two results providing supporting evidence that the diagonalizing CNF Φ is unsatisfiable.

6.1 Tensor Rank Hardness Entails that Φ is Unsatisfiable

Here we show that a lower bound against constant-depth IPS refutations of a formula expressing a tensor with tensor rank m cannot be decomposed to the summation of n rank-1 tensors, implying that the diagonalization formula Φ is unsatisfiable.

It is now known from [LST25] that basic linear-algebraic operations such as matrix rank, determinant, and plausibly solving systems of linear equations cannot be efficiently carried out by constant-depth algebraic circuits, i.e., VAC^0. The rank of an $n \times n$ matrix over a field is only known to be computable in uniform NC^2 via Mulmuley's parallel algorithm [Mul86] (in fact, integer determinant is computable already in $\#\mathsf{SAC}^1$ [Coo85]). In contrast, determining the rank of a 3-dimensional tensor is known to be NP-hard [Hås90, Shi16].

The following is the rank principle as studied in [GGRT25] (cf. [SU04, Kra09, GGPS23]).

Definition 6.1 (Rank Principle)**.** *Let $\mathbb{F}$ be a field and m, n be two positive integers such that $m > n$. We denote $\mathrm{RankP}^m_n(A)$ the system of degree-2 polynomial equations stating that the rank of an $m \times m$ matrix A is at most n over $\mathbb{F}$. More precisely, the* rank principle *$\mathrm{RankP}^m_n(A)$ is defined over $2mn$*

variables arranged into two matrices $X \in \mathbb{F}^{m\times n}$ and $Y \in \mathbb{F}^{n\times m}$, For every $i, j \in [m]$, there is an equation in $\mathrm{RankP}^m_n(A)$ *stating that the (i, j)th entry of the product XY is equal to $A_{i,h}$. That is*

$$\sum_{k\in[n]} x_{i,k}y_{k,j} - A_{i,j}, \qquad i, j \in [m]. \tag{39}$$

By linear algebra, when the rank of A exceeds n, $\mathrm{RankP}^m_n(A)$ is unsatisfiable. The work of [GGL+25] considers a generalisation of the rank principle, which they call the tensor rank principle.

Definition 6.2 (Tensor Rank Principle)**.** *Let $\mathbb{F}$ be a field and m, n be two positive integers such that $m > n$. We denote* $\mathrm{TRankP}^r_{m,n}(A)$ *the system of degree-r polynomial equations stating that the tensor rank of the r-tensor $A \in \mathbb{F}^{\overbrace{(m\times\cdots\times m)}^{r\ times}}$ is at most n over $\mathbb{F}$. More precisely, the rth-order tensor rank principle* $\mathrm{TRankP}^r_{m,n}(A)$ *is defined over rmn variables arranged into r matrices $X_1, \dots, X_r \in \{0, 1\}^{m\times n}$ where each matrix X_i can be viewed as n vectors $\mathbf{x}_{j,k} \in \{0, 1\}^m$ for $k \in [n]$ and $j \in [r]$. For every $i_1, \dots, i_r \in [m]$ (not necessarily distinct), there is an equation in* $\mathrm{TRankP}^r_{m,n}(A)$ *stating that the $(i_1, \dots, i_r)$th entry of the summation of the tensor product $\bigotimes_{j=1}^r \mathbf{x}_{j,k}$ is equal to $A_{i_1,\dots,i_r}$ (where $\otimes$ denotes the tensor (outer) product of vectors). That is,*

$$\sum_{k\in[n]} \prod_{j\in[r]} x_{j,i_j,k} = A_{i_1,\dots,i_r}, \qquad i_1, \dots, i_r \in [m], \tag{40}$$

where $x_{j,i_j,k}$ denote the (i_j, k)th entry of the matrix X_j.

Additionally, for every $i \in [m]$, $k \in [n]$ and $j \in [r]$, we have a Boolean axiom $x^2_{j,i,k} - x_{j,i,k}$. Namely, $\mathrm{TRankP}^r_{m,n}(A)$ *is the set of polynomial equations stating that*

$$\sum_{k\in[n]} \bigotimes_{j=1}^{r} \mathbf{x}_{j,k} = A.$$

Note that $\bigotimes_{j=1}^r \mathbf{x}_{j,k}$ is the rth order tensor of rank 1 obtained by the outer product of all the kth columns in $X_1, \dots, X_r$. Hence, by basic algebra, it follows that the sum of n rank 1 tensors $\sum_{k\in[n]} \bigotimes_{j=1}^r \mathbf{x}_{j,k}$ has tensor rank at most n. Thus, whenever the tensor rank of A exceeds n, the formula $\mathrm{TRankP}^r_{m,n}(A)$ is unsatisfiable. For standard background on tensor rank and tensor decompositions, see the survey by Kolda and Bader [KB09]. As noted earlier, the tensor rank principle generalises the matrix equation $XY = A$, which corresponds to the rank principle. In fact, the rank principle $\mathrm{RankP}^m_n(A)$ is precisely the case $r = 2$ of the tensor rank principle, i.e., $\mathrm{TRankP}^2_{m,n}(A)$.

[GGL+25] proved that $\mathrm{TRankP}^r_{m,n}(A)$ requires $\exp(\Omega(n))$-size PCR refutations over the two-element field.

Theorem 6.3 ([GGL+25])**.** *Every* PCR *refutation over $\mathbb{F}_2$ of* $\mathrm{TRankP}^r_{m,n}(A)$ *requires 2^{cn} monomials for some constant c.*

They also exhibit a reduction from the tensor rank principle to bounded-depth algebraic circuit upper bounds statements (so that the latter are at least as hard as the former), specifically the statement "perm $\in \mathsf{VAC}^0$."

Definition 6.4 (Bounded-depth algebraic circuit upper bound formulas AUB)**.** *Let $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ be a polynomial with n-variables and degree at most l. The following set of polynomial equations*

$\mathrm{AUB}(f, s, l, \Delta)$ *(in the $\overline{w}$-variables only) express that the polynomial $f(\overline{x})$ can be computed by a bounded-depth algebraic circuit of size s and depth Δ:*

$$\{\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) = b_i : 1 \le i \le N\},$$

where $\overline{b} = \mathsf{coeff}(f(\overline{x})) \in \mathbb{F}^N$ is the coefficient vector of the polynomial f of dimension N, $U(\overline{x}, \overline{w})$ is the constant-depth universal circuit for polynomials of depth at most Δ and size at most s, $\overline{w}$ are the $K_{s,\Delta}$ (circuit) edge variables, $\{M_i\}_{i=1}^N$ is the set of all possible $\overline{x}$-monomials of degree at most l, and $N = \Sigma_{j=0}^{l} \binom{n+j-1}{j} = 2^{O(n+l)}$ is the number of monomials of total degree at most l over n variables. The size of the above set of polynomial equations is $O(2^{(t+l)l} \cdot |U(\overline{x}, \overline{w})| \cdot N)$ where t is the maximum multiplication fan-in in $U(\overline{x}, \overline{w})$.

Theorem 6.5 (Constant-depth algebraic circuit upper bounds are at least as hard as tensor rank principle [GGL+25])**.** *Suppose $\mathsf{SLP}(\mathrm{AUB}(f, s, \ell, \Delta))$ admits size-S depth-Δ' IPS refutations where $\Delta > 4$ and $s > n2\ell^4$. Then, $\mathrm{TRankP}^n_{N,\sqrt{s}}(A)$ admits size-$O(Nn \cdot S + NK \cdot |\mathsf{SLP}(\mathrm{AUB}(f, s, l, \Delta))|)$ depth-$(\Delta' + 5)$ IPS refutations where*

- *$N = \binom{n+\ell}{\ell}$ is the number of monomials in n variables and total degree at most ℓ.*
- *A is the n-tensor such that the $(\mathcal{M}, \dots, \mathcal{M})$th entry of A is $\mathsf{coeff}_{\mathcal{M}}(p)$ where $\mathcal{M}$ is a monomial in n variables and total degree at most ℓ. The rest of A are all zeros.*

Corollary 6.6 ([GGL+25])**.** *Suppose $\mathrm{TRankP}^r_{m,n}(A)$ requires 2^{n^δ}-size depth-Δ' IPS refutations, for some constant $\delta > 0$. Then, $\mathsf{SLP}(\mathrm{AUB}(f, s, \ell, \Delta))$ requires $|\mathsf{SLP}(\mathrm{AUB}(f, s, \ell, \Delta))|^{\omega(1)}$-size, depth-$(\Delta' - 5)$ IPS refutation.*

In particular, this shows that the hardness of the tensor rank principle against constant-depth IPS entails the unsatisfiability of the diagonalizing formula Φ from Section 5.

6.2 Unconditional PCR Lower Bounds for Algebraic Circuit Upper Bound Formulas

In the previous section, we showed that the tensor rank principle $\mathrm{TRankP}^r_{m,n}(A)$ can be reduced to the *bounded-depth* algebraic circuit upper bound formulas $\mathrm{AUB}(f, s, l, \Delta)$. In this section, we show that a variant of the rank principle can be reduced to *general* (unbounded depth) algebraic circuit upper bound formulas $\mathrm{AUB}(f, s, l)$. This yields an *unconditional* PCR lower bound for $\mathrm{AUB}(f, s, l)$.

The definition of algebraic circuit upper bound formulas is similar to the bounded-depth case, except that the universal circuit for bounded-depth is replaced with the universal circuit for general algebraic circuits, as defined in [Raz10] (see [ST25]). Accordingly, we adopt the same notation for general algebraic circuit upper bound formulas as for the bounded-depth case, except that the depth parameter is omitted.

Definition 6.7 (Algebraic circuit upper bound formula)**.** *Let $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ be a polynomial with n-variables and degree at most l. The following set of polynomial equations $\mathrm{AUB}(f, s, l)$ (in the $\overline{w}$-variables only) expressing that the polynomial $f(\overline{x})$ can be computed by an algebraic circuit of size s:*

$$\{\mathsf{coeff}_{M_i}(U(\overline{x}, \overline{w})) = b_i : 1 \le i \le N\},$$

where $\overline{b} = \mathsf{coeff}(f(\overline{x})) \in \mathbb{F}^N$ is the coefficient vector of the polynomial f of dimension N, $U(\overline{x}, \overline{w})$ is the universal circuit for polynomials of degree at most l and circuit size at most s, $\overline{w}$ are the $K_{s,l}$ edge variables, $\{M_i\}_{i=1}^N$ is the set of all possible $\overline{x}$-monomials of degree at most l, and $N = \Sigma_{j=0}^{l} \binom{n+j-1}{j} = 2^{O(n+l)}$ is the number of monomials of total degree at most l over n variables. The size of $\mathrm{AUB}(f, s, l)$ is $O(7^l \cdot |U(\overline{x}, \overline{w})| \cdot N)$.

Now, we define an *iterated* version of the rank principle. Let $\mathbb{F}_L$ be a finite field of characteristic L. We will denote by $\mathbb{F}_L^{\leq n}$ the set of vectors over $\mathbb{F}_L$ of length at most n. Similarly, $\mathbb{F}_L^{<n}$ denotes the set of vectors over $\mathbb{F}_L$ of length less than n. Let $L, n, K \in \mathbb{N}^+$. For every vector $\pi \in \mathbb{F}_L^{\leq n}$, let $X^\pi = (x^\pi_{i,k})_{i\in[LK],k\in[K]}$ be variable matrices (unique for each different π). Let Y be a $K \times LK$ matrix in variables $y_{k,j}$ for $k \in [K], j \in [LK]$. For every vector $\pi \in \mathbb{F}_L^n$, let A^π be an $LK \times LK$ matrix over $\mathbb{F}_L$, and $\{A^\pi\}$ be the set consisting of all A^π over all vectors π.

Definition 6.8 (Iterated Rank Principle [GGRT25])**.** *Let L, n, K be parameters in $\mathbb{N}^+$. The* Iterated Rank Principle *is* $\mathrm{IRankP}_{L,n,K}(\{A^\pi\}) :=$

$$\sum_{k\in[K]} x^\pi_{i,k} y_{k,j} - x^{\pi(\lceil \frac{j}{K}\rceil - 1)}_{i,j-(\lceil \frac{j}{K}\rceil-1)K}, \qquad \forall \pi \in \mathbb{F}_L^{<n}, i \in [LK], j \in [LK],$$

$$\sum_{k\in[K]} x^\pi_{i,k} y_{k,j} - A^\pi_{i,j}, \qquad \forall \pi \in \mathbb{F}_L^{n}, i \in [LK], j \in [LK].$$

Namely, $\mathrm{IRankP}_{L,n,K}(\{A^\pi\})$ *contains all the degree-2 polynomial equations in the following matrix multiplications (where πb, for $b \in \mathbb{F}_L$, denotes concatenation of b to π):*

$$X^\pi Y = [X^{\pi 0} X^{\pi 1} \cdots X^{\pi(L-1)}], \quad \forall \pi \in \mathbb{F}_L^{\leq n}, \tag{41}$$

$$X^\pi Y = A^\pi, \quad \forall \pi \in \mathbb{F}_L^{n}. \tag{42}$$

Note that Equation (42) are instances of the rank principle, hence the Iterated Rank Principle is no stronger than the rank principle. Intuitively, Equation (41) provides auxiliary axioms that allow us to access the nodes of a tree, in which we can embed a circuit.

Definition 6.9 (Iterated Rank Principle with Extension Variables)**.** *Let L, n, K be parameters in $\mathbb{N}^+$. The* Iterated Rank Principle with Extension Variables *denoted* $\mathrm{IRankPE}_{L,n,K}(\{A^\pi\})$ *consists of the equations in* $\mathrm{IRankP}_{L,n,K}(\{A^\pi\})$ *together with*

$$z^\pi_{i,k,j} - x^\pi_{i,k} y_{k,j}, \qquad \forall \pi \in \mathbb{F}_L^{<n}, i \in [LK], k \in [K], j \in [LK].$$

Theorem 6.10 ([GGRT25])**.** *Every* PCR *refutation over $\mathbb{F}_2$ of* $\mathrm{IRankPE}_{L,n,K}(\{A^\pi\})$ *requires* 2^{K^δ} *monomials for some constant $\delta > 0$.*

Note that the size of $\mathrm{IRankPE}_{L,n,K}(\{A^\pi\})$ is exponential in n. Hence, if we choose K large enough (e.g., $K \geq n^{100}$) this theorem gives a super-polynomial PCR lower bound. [GGL+25] further showed, through a reduction from the iterated rank principle with extension variables to (the SLP Definition 2.31 version of) $\mathrm{AUB}(f, s, l)$, that the latter admits an unconditional size lower bound against PCR.

Theorem 6.11 ([GGL+25])**.** *Every* PCR *refutation over $\mathbb{F}_2$ of* $\mathsf{SLP}(\mathrm{AUB}(f,s,l))$ *where s is polynomially bounded requires* $|\mathsf{SLP}(\mathrm{AUB}(f,s,l))|^{\omega(1)}$ *many monomials.*

7 No Short Bounded-Depth IPS Refutations for Diagonalizing CNF Formulas: the Polynomial-Size Fields Case

In this section, we work over finite fields whose characteristic is polynomially bounded by the instance size, in order to obtain a more general result. This contrasts with the previous section, where the characteristic of the finite field was a fixed global constant, independent of the instance size. Here we encode binary string arithmetic—including addition, multiplication, and modular computation into CNF formulas. Within this setting, we also establish the translation lemma, which yields a version of Theorem 5.3 over finite fields of polynomially bounded characteristic. In other words, we show that no efficient provability result holds against constant-depth IPS over polynomial-size finite fields. Since Forbes [For24] extended [LST25] to arbitrary fields, the results of this section generalize Theorem 5.3. The techniques here, however, are somewhat more involved.

We will continue to use notations such as $\mathsf{cnf}(C(\overline{x}) = 0)$ and $\mathsf{ecnf}(C(\overline{x}) = 0)$ from the fixed-field setting, but with a different interpretation: in this section they refer to the CNF and Extended CNF encodings of $C(\overline{x}) = 0$ obtained via bit-level arithmetic, which we describe below.

7.1 Bit Arithmetic

Field elements are encoded in standard binary representation. We work over the finite field $\mathbb{F}_q$. Note that the characteristic of $\mathbb{F}_q$ is not constant. More precisely, we work over a finite field $\mathbb{F}_q$ where q may grow polynomially (with the input size).

Definition 7.1 (The encoding of binary value VAL)**.** *Given a bit vector $w_{k-1}, \ldots, w_0$ of variables w, denoted $\overline{w}$, ranging over 0-1 values, define the following algebraic formulas:*

$$w = \mathsf{VAL}(\overline{w})$$
$$\mathsf{VAL}(\overline{w}) = \Sigma_{i=0}^{k-1} 2^i \cdot w_i,$$

where $\mathsf{VAL}(\overline{w})$ is an extension variable. The size of a VAL is $O(k)$.

Definition 7.2 (Arithmetization operation $\mathsf{arit}(\cdot)$)**.** *For a variable y_i, $\mathsf{arit}(y_i) = y_i$. For the truth value false $\bot$ and true $\top$ we put $\mathsf{arit}(\bot) := 0$ and $\mathsf{arit}(\top) := 1$. For logical connectives we define $\mathsf{arit}(A \wedge B) := \mathsf{arit}(A) \cdot \mathsf{arit}(B)$, $\mathsf{arit}(A \vee B) := 1 - (1 - \mathsf{arit}(A)) \cdot (1 - \mathsf{arit}(B))$, and for XOR operation we define $\mathsf{arit}(A \oplus B) := \mathsf{arit}(A) + \mathsf{arit}(B) - 2 \cdot \mathsf{arit}(A) \cdot \mathsf{arit}(B)$.*

In this way, for every Boolean formula $F(\overline{x})$ with n variables and a Boolean substitution $\overline{\alpha} \in \{0,1\}^n$, $\mathsf{arit}(F)(\overline{\alpha}) = 1$ if and only if $F(\overline{\alpha}) = \top$.

We will present the CNF encoding of unbounded fan-in algebraic circuits using bit arithmetic. However, for simplicity, we present the algebraic encoding ϕ that is "equivalent" to the CNF formula F. For "equivalent", we mean that ϕ can be derived from F in constant depth and constant size, and vice versa. The reason why this can be achieved is that each formula has only a constant number of variables. Therefore, by the implicational completeness of IPS over 0-1 assignment, IPS can derive all formulas simultaneously in constant depth. It is easy to see that all formulas we give below can be written as CNFs. Also, we write $\phi - \psi = 0$ as $\phi = \psi$.

We divide the addition bit arithmetic in a finite field into two big steps:

- Addition Step: for two k length binary representations $\overline{x} = x_{k-1}, \ldots, x_0$ and $\overline{y} = y_{k-1}, \ldots, y_0$, we do the general addition bit arithmetic which outputs a $k+1$ length binary representations $\mathsf{ADD}_k, \ldots, \mathsf{ADD}_0$.
- Modular Step: we turn this $k+1$ length binary representations $\mathsf{ADD}_k, \ldots, \mathsf{ADD}_0$ into a k length binary representation $\mathsf{ADD}'_{k-1}, \ldots, \mathsf{ADD}'_0$, which represents the same number in finite field $\mathbb{F}_q$. In other words, $\sum_{i=0}^{k} 2^i \mathsf{ADD}_i \equiv \sum_{i=0}^{k-1} 2^i \mathsf{ADD}'_i \mod q$. Note that, it is possible that $\sum_{i=0}^{k-1} 2^i \mathsf{ADD}'_i \geq q$.

For the convenience of later use of the Modular step in multiplication bit arithmetic in a finite field, we generalize our Modular step into the following:

- Given a k length binary representations $\overline{x} = x_{k-1}, \dots, x_0$ and an extra bit x' which is the $t+1$th bit where $t \geq k$. We do the addition bit arithmetic of $0, x_{k-2}, \dots, x_0$ and the k length binary representation of $(2^t x' + 2^{k-1} x_{k-1}) \mod q$, which outputs a $k+1$ length binary representation $\overline{y} = y_k, \dots, y_0$.

- Then, we do the addition bit arithmetic of $y_{k-1}, \dots, y_0$ and $(2^k y_k \mod q)$, which outputs a $k-1$ length binary representation. The reason why the output must be smaller than 2^k is as follows: Notice that, both two items $\sum_{i=0}^{k-2} 2^i x_i$ and $(2^t x' + 2^{k-1} x_{k-1}) \mod q$ are smaller than q since $\sum_{i=0}^{k-2} 2^i x_i < 2^{k-1} \leq q$. If $\sum_{i=0}^{k-1} 2^i y_i + 2^k y_k < 2^k$, then we are done. Otherwise, suppose $\sum_{i=0}^{k-1} 2^i y_i + 2^k y_k \geq 2^k$. Since $\sum_{i=0}^{k-1} 2^i y_i \leq 2^k - 1$, $\sum_{i=0}^{k-1} 2^i y_i + 2^k y_k \geq 2^k$ implies that $y_k = 1$. Hence,

$$\begin{aligned}\sum_{i=0}^{k-1} 2^i y_i + (2^k y_k \mod q) &\leq \sum_{i=0}^{k-1} 2^i y_i + 2^k y_k - q \\ &= \sum_{i=0}^{k-2} 2^i x_i + (2^t x' + 2^{k-1} x_{k-1} \mod q) - q \\ &< 2q - q \\ &= q \\ &< 2^k\end{aligned}$$

  Therefore, we know that the output of this step must be a $k-1$ length binary representation.

Now, we define the formula for the addition step.

Definition 7.3 (The carry bit CARRY_i, the addition bit ADD_i and the encoding of carry lookahead addition $\mathsf{Addition}$)**.** *Suppose we have two k length binary representations $\overline{x} = x_{k-1}, \dots, x_0$ and $\overline{y} = y_{k-1}, \dots, y_0$. We define the carry bit CARRY_i, the addition bit ADD_i and the encoding of carry lookahead addition $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$ as follows, together with the Boolean axioms for each variable:*

$$\mathsf{CARRY}_i(\overline{x}, \overline{y}) = \begin{cases} \mathsf{arit}((x_{i-1} \wedge y_{i-1}) \vee ((x_{i-1} \vee y_{i-1}) \wedge \mathsf{CARRY}_{i-1}(\overline{x}, \overline{y}))), & i = 1, \dots, k; \\ \mathsf{arit}(\bot), & i = 0, \end{cases}$$

and

$$\begin{aligned}\mathsf{ADD}_i(\overline{x}, \overline{y}) &= \mathsf{arit}(x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})), \qquad i = 0, \dots, k-1 \\ \mathsf{ADD}_k(\overline{x}, \overline{y}) &= \mathsf{CARRY}_k(\overline{x}, \overline{y})\end{aligned}$$

The size of the encoding $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$ is also $O(k)$. The encoding $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$ represents the addition of two k length binary representations $\overline{x}$ and $\overline{y}$, which gives a $k+1$ length binary representation $\overline{\mathsf{ADD}}$.

Also, we denote $\mathsf{Addition}'$ as encoding of carry lookahead addition without CARRY_k and ADD_k. In other words, $\mathsf{Addition}'(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$ represents the addition of two k length binary representations $\overline{x}$ and $\overline{y}$, which gives a k length binary representation $\overline{\mathsf{ADD}}$.

Notice that both CARRY_i and ADD_i are extension variables. Now, we define the encoding of modular.

Definition 7.4 (The encoding of modular $\mathsf{Modular}$)**.** *Suppose we have a k length binary representation $\overline{x} = x_{k-1}, \ldots, x_0$ and another binary bit x' which is the $t+1$th bit in a binary representation, the encoding of modular* $\mathsf{Modular}^t(\overline{x}, x', \overline{\mathsf{ADD}'})$ *is defined as follows, together with the Boolean axioms for each variable:*

$$\mathsf{Addition}(\overbrace{0, x_{k-2}, \ldots, x_0}^{\text{the last } k \text{ bit of } x \text{ with } x_{k-1} \text{ exchanged to } 0}, \underbrace{\overline{m}}_{\text{the } k \text{ length binary representation of } 2^t x' + 2^{k-1} x_{k-1} \mod q}, \overline{u})$$

$$\mathsf{Addition}'(\overbrace{u_{k-1}, \ldots, u_0}^{\text{the last } k \text{ bit of } u}, \underbrace{\overline{w}}_{\text{the } k \text{ length binary representation of } 2^k u_k}, \overline{\mathsf{ADD}'})$$

$\overline{m}$ and $\overline{w}$ are defined as follows. Suppose $\overline{a}$ is the binary representation of $2^t \mod q$, $\overline{b}$ is the binary representation of $2^{k-1} \mod q$ and $\overline{c}$ is the binary representation of $2^t + 2^{k-1} \mod q$, each bit m_i in $\overline{m}$ can be computed by a Boolean function $f_i^t(x', x_{k-1})$ whose truth table is as follows:

x'	x_{k-1}	$f_i^t(x', x_{k-1})$
0	0	0
0	1	b_i
1	0	a_i
1	1	c_i

Therefore, $f_i^t(x', x_{k-1})$ can be represented by a CNF *with variables x' and x_{k-1} since all $\overline{a}, \overline{b}$ and $\overline{c}$ are fixed. We denote such* CNF *as* $\mathsf{cnf}(f_i^t(x', x_{k-1}))$*. Hence, $\overline{m}$ are defined as* $m_i := \mathsf{arit}(\mathsf{cnf}(f_i^t(x', x_{k-1})))$*.*

Similarly, $\overline{w}$ are defined as $w_i := \mathsf{arit}(\mathsf{cnf}(g_i(u_k)))$ *where g_i is another Boolean function only depends on u_k.*

With the encoding of carry lookahead addition $\mathsf{Addition}$ and the encoding of modular $\mathsf{Modular}$, we define the CNF encoding of $x + y = z$.

Definition 7.5 (The CNF encoding of bit arithmetic for addition in a finite field)**.** *For any* SLP *formula $x + y - z = 0$, the* CNF *encoding of bit arithmetic for addition in a finite field, denoted as* $\mathsf{CNF\text{-}ADD}(\overline{x}, \overline{y}, \overline{z})$*, is as follows:*

- *Addition step:* $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$
- *Modular step:* $\mathsf{Modular}^k(\mathsf{ADD}_{k-1}, \ldots, \mathsf{ADD}_0, \mathsf{ADD}_k, \overline{\mathsf{ADD}'})$
- *Connection step: for each $0 \le i \le k-1$, we have* $\mathsf{ADD}'_i = z_i$*.*

For the multiplication bit arithmetic in a finite field, we divide it into three big steps:

- Multiplication Step: For two k length binary representations $\overline{x} = x_{k-1}, \ldots, x_0$ and $\overline{y} = y_{k-1}, \ldots, y_0$, we do the general multiplication bit arithmetic which outputs k many k binary representations $\overline{s_0}, \ldots, \overline{s_{k-1}}$ where $s_{ij} = x_j \wedge y_i$.
- Modular Step: for $\overline{s_i}$, we do the generalized modular step for i times, which gives us a k length binary representation whose value is the same as $\sum_{j=i}^{i+k} 2^j s_{ij}$ in the finite field $\mathbb{F}_q$.

- Addition Step: After the Modular Step, we have k many k-length binary representations. Using the addition we showed above, we can add them together in the finite field $\mathbb{F}_q$ and get a k-length binary representation.

Definition 7.6 (The encoding of multiplication MULT_i, MULT)**.** *Suppose we have two binary representations $\overline{x} = x_{k-1}, \dots, x_0$ and $\overline{y} = y_{k-1}, \dots, y_0$, we define the encoding of multiplication $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \cdots, \overline{s_{k-1}})$ as follows, together with the Boolean axioms for each variable:*

$$\mathsf{MULT}_i(\overline{x}, \overline{y}, \overline{s_i}) \coloneqq \begin{cases} s_{i,j} = \mathsf{arit}(x_{j-i} \wedge y_i), & i \le j \le k-1+i \\ s_{i,j} = \mathsf{arit}(\bot), & 0 \le j < i. \end{cases}, \qquad 0 \le i \le k-1$$

where $\overline{s_i}$ is a $k+i$ length 0-1 vector. The size of the encoding of multiplication $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \cdots, \overline{s_{k-1}})$ is $O(k^2)$.

Definition 7.7 (The CNF encoding of bit arithmetic for multiplication in a finite field)**.** *For any SLP formula $x \times y - z = 0$, the CNF encoding of bit arithmetic for multiplication in a finite field denoted as $\mathsf{CNF\text{-}MULT}(\overline{x}, \overline{y}, \overline{z})$, is as follows:*

- *Multiplication step:* $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \cdots, \overline{s_{k-1}})$
- *Modular step: for each $\overline{s_i}$ such that $1 \le i \le k-1$, we have the following i many modular formula:*

$$\begin{aligned} &\mathsf{Modular}^k(s_{i,k-1}, \dots, s_{i,0}, s_{i,k}, \overline{u_{i,k}}) \\ &\mathsf{Modular}^{k+1}(\overline{u_{i,k}}, s_{i,k+1}, \overline{u_{i,k+1}}) \\ &\vdots \\ &\mathsf{Modular}^j(\overline{u_{i,j-1}}, s_{i,j}, \overline{u_{i,j}}), \qquad k+1 \le j \le i+k-1. \end{aligned}$$

  All $\overline{u_{i,j}}$ is a k length binary representation.

- *Addition step: Now for each $\overline{s_i}$, we have a k length binary representation $\overline{u_{i,i+k-1}}$ such that $\sum_{j=0}^{i+k-1} 2^j s_{i,j} = \sum_{j=0}^{k-1} 2^j u_{i,i+k-1,j}$ in the finite field $\mathbb{F}_q$. Then, using the CNF encoding of bit arithmetic for addition in a finite field as we defined above in Definition 7.5, we have the following:*

$$\begin{aligned} &\mathsf{CNF\text{-}ADD}(\overline{s_0}, \overline{u_{1,k}}, \overline{v_1}) \\ &\mathsf{CNF\text{-}ADD}(\overline{v_1}, \overline{u_{2,k+1}}, \overline{v_2}) \\ &\vdots \\ &\mathsf{CNF\text{-}ADD}(\overline{v_i}, \overline{u_{i+1,k+i}}, \overline{v_{i+1}}), \qquad 1 \le i \le k-2 \end{aligned}$$

- *Connection step: for each $0 \le j \le k-1$, we have $v_{k-1,j} = z_j$.*

The size of $\mathsf{CNF\text{-}MULT}(\overline{x}, \overline{y}, \overline{z})$ is $O(k^3)$.

Definition 7.8 (CNF encoding of unbounded fan-in algebraic circuits; $\mathsf{cnf}(C(\overline{x}))$)**.** *Let $C(\overline{x})$ be an (unbounded fan-in) algebraic circuit in variables $\overline{x}$. The CNF encoding of $C(\overline{x})$ denoted $\mathsf{cnf}(C(\overline{x}))$ consists of the following CNFs in the binary representation bits variables of all the nodes in C and extra extension variables (and only in the binary representation bits variables):*

- *If $\alpha \in \mathbb{F}$ is a scalar input node in C, the* CNF *encoding of C contains the $\{0,1\}$ constant corresponding to the binary representation bits of α. These constants are used when fed to nodes according to the wiring of C.*

- *For every node g in $C(\overline{x})$, suppose g is a $+$ node that has inputs $u_1, \ldots, u_t$. Then, firstly, we have the formula* CNF-ADD *for each of the following equations:*

$$\begin{aligned} &u_1 + u_2 = v_1^g \\ &u_{i+2} + v_i^g = v_{i+1}^g, \qquad 1 \le i \le t-3 \\ &u_t + v_{t-2}^g = g. \end{aligned}$$

  Same for $\times$ nodes. Suppose g is a $\times$ node that has inputs $u_1, \ldots, u_t$. Then, we have the multiplication formula CNF-MULT *for each following equations:*

$$\begin{aligned} &u_1 \times u_2 = v_1^g \\ &u_{i+2} \times v_i^g = v_{i+1}^g, \qquad 1 \le i \le t-3 \\ &u_t \times v_{t-2}^g = g. \end{aligned}$$

- *For every Boolean variables u, we have the Boolean axiom*

$$u_i^2 - u_i = 0$$

We call variables v_i^g the intermediate nodes *which are nodes that do not exist in C but are used to help to encode and the* SLP *formula $u_{i+2} + v_i^g = v_{i+1}^g$ or $u_{i+2} \times v_i^g = v_{i+1}^g$ the* intermediate SLP formulas*. The size of the* CNF *encoding of unbounded fan-in algebraic circuits* $\mathsf{cnf}(C(\overline{x}))$ *is $O(tk^3|C|) = O(k^3|C|^2)$ where t is the maximum fan-in.*

Notice that, in the above CNF encoding, we can only guarantee that the output of the circuit, which is a k length binary representation $\overline{g_{out}}$, is equal to zero in the finite field $\mathbb{F}_q$. In other words, we can guarantee that $\sum_{i=0}^{k-1} 2^i g_{out,i} = 0$. However, this does not mean that $g_{out,i} = 0$ for all bits. Instead of guaranteeing each $g_{out,i}$ is equal to 0, we guarantee that each $g_{out,i} = q_i$ where $\overline{q}$ is the k length binary representation of q. Therefore, to encode the output of the algebraic circuit is equal to 0, we add q to the output, which is

$$\mathsf{CNF\text{-}ADD}(\overline{g_{out}}, \underbrace{\overline{q}}_{\text{the binary representation of } q}, \overline{g_{out}}).$$

Definition 7.9 (CNF encoding of unbounded fan-in algebraic circuit equations; $\mathsf{cnf}(C(\overline{x})) = 0$)**.** *Let $C(\overline{x}) = 0$ be a circuit equation in the variables $\overline{x}$ over a finite field $\mathbb{F}_q$. The* CNF *encoding of it denoted* $\mathsf{cnf}(C(\overline{x}) = 0)$ *consists of the* CNF *encoding of $C(\overline{x})$ from Definition 7.8 together with the equations*

- $\mathsf{CNF} = \mathsf{ADD}(\overline{g_{out}}, \underbrace{\overline{q}}_{\textit{the binary representation of } q}, \overline{g_{out}})$

- $g_{out,i} = q_i, \quad 0 \le i \le k-1$ *which express that the output node $g_{out} = 0$ and $g_{out,i}$ are the binary representation bits of g_{out}. We call these formulas the connection formulas for the output node.*

The size of the CNF *encoding of unbounded fan-in algebraic circuits equation* $\mathsf{cnf}(C(\overline{x}) = 0)$ *is $O(tk^3|C|) = O(k^3|C|^2)$ where t is the maximum fan-in.*

Definition 7.10 (Extended CNF encoding of unbounded fan-in algebraic circuit equation (circuit *resp.*); $\mathsf{ecnf}(C(\overline{x}) = 0)$ ($\mathsf{ecnf}(C(\overline{x}))$, *resp.*))**.** *Let $C(\overline{x})$ be a circuit in the $\overline{x}$ variables over the field $\mathbb{F}_q$. Then the extended* CNF *encoding of the circuit equation $C(\overline{x}) = 0$ (circuit $C(\overline{x})$,* resp.*), in symbols* $\mathsf{ecnf}(C(\overline{x}) = 0)$ *(*$\mathsf{ecnf}(C(\overline{x}))$, resp.*), is defined to be the following:*

- *the* CNF *encoding of circuit equation $C(\overline{x}) = 0$ (circuit $C(\overline{x})$,* resp.*), namely,* $\mathsf{cnf}(C(\overline{x}) = 0)(\mathsf{cnf}(C(\overline{x})), \text{resp.})$*; and*
- *the binary value formula for each binary representation bits vector.*

The size of the Extended CNF *encoding of unbounded fan-in algebraic circuits equation* $\mathsf{ecnf}(C(\overline{x}) = 0)$ *is $O(tk^3|C|) = O(k^3|C|^2)$ where t is the maximum fan-in.*

Same as the argument given in [ST25], we have the following propositions.

Proposition 7.11. *Let $C(\overline{x})$ be a circuit equation over $\mathbb{F}$. Then, $C(\overline{x})$ is unsatisfiable over $\mathbb{F}$ if and only if* $\mathsf{cnf}(C(\overline{x}) = 0)$ *is an unsatisfiable* CNF *if and only if* $\mathsf{ecnf}(C(\overline{x})) = 0$ *is an unsatisfiable set of equations over $\mathbb{F}$.*

Corollary 7.12. *If* $\mathsf{ecnf}(C(\overline{x}) = 0)$ *is unsatisfiable over $\mathbb{F}_q$ then it has an* $\mathsf{IPS}^{\mathsf{alg}}$ *refutation in* $\mathsf{VNP}_{\mathbb{F}_q}$*.*

Now, we are ready to show the translation lemma for bit arithmetic encoding, similarly to Lemma 2.35 and Lemma 2.36.

Lemma 7.13. *Let $\mathbb{F}_q$ be a finite field. Let $\overline{x} = x_{k-1}, \ldots, x_0$, $\overline{y} = y_{k-1}, \ldots, y_0$ and $\overline{z} = z_{k-1}, \ldots, z_0$ be three k length binary representations. Then,*

$$\mathsf{CNF\text{-}ADD}(\overline{x}, \overline{y}, \overline{z}) \Big|\!\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\ \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{z})$$

Lemma 7.14. *Let $\mathbb{F}_q$ be a finite field. Let $\overline{x} = x_{k-1}, \ldots, x_0$ and $\overline{y} = y_{k-1}, \ldots, y_0$ be two k length binary representations. Let* $\overline{\mathsf{ADD}} = \mathsf{ADD}_k, \mathsf{ADD}_{k-1}, \ldots, \mathsf{ADD}_0$ *be a $k+1$ length binary representation. Then,*

$$\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}}) \Big|\!\frac{O(k), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\ \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{\mathsf{ADD}_i(\overline{x}, \overline{y})}).$$

In other words, from $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$*, there is a $O(1)$-depth, $O(k)$-size* $\mathsf{IPS}^{\mathsf{alg}}$ *proof of* $\mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{\mathsf{ADD}_i(\overline{x}, \overline{y})})$*.*

Proof of Lemma 7.14. By Definition 7.3, $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$ includes

$$\begin{aligned} \mathsf{ADD}_i(\overline{x}, \overline{y}) &= \mathsf{arit}(x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})), \qquad i = 0, \cdots, k-1 \\ \mathsf{ADD}_k(\overline{x}, \overline{y}) &= \mathsf{CARRY}_k(\overline{x}, \overline{y}). \end{aligned}$$

By simple substitution, it suffices to show

$$\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}}) \Big|\!\frac{*, O(1)}{\mathsf{IPS}^{\mathsf{alg}}} \sum_{i=0}^{k-1} (x_i + y_i) \cdot 2^i = \sum_{i=0}^{k-1} (x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})) \cdot 2^i + \mathsf{CARRY}_k(\overline{x}, \overline{y}) \cdot 2^k.$$

For each $0 \le i \le k-1$, we aim to show that there is a constant-depth, constant-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the following equation:

$$2^i \cdot (x_i + y_i) = 2^i (x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})) + 2^{i+1} \mathsf{CARRY}_{i+1}(\overline{x}, \overline{y}) - 2^i \mathsf{CARRY}_i(\overline{x}, \overline{y}).$$

By substituting CARRY_{i+1} with $(x_i \vee y_i) \wedge \mathsf{CARRY}_i(\overline{x}, \overline{y}))$ according to Definition 7.3, for each $0 \le i \le k-1$, the above equation becomes

$$2^i \cdot (x_i + y_i) = 2^i(x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})) + 2^{i+1}((x_i \wedge y_i) \vee ((x_i \vee y_i) \wedge \mathsf{CARRY}_i(\overline{x}, \overline{y}))) - 2^i \mathsf{CARRY}_i(\overline{x}, \overline{y}).$$

Note that the above equation, which has only three Boolean variables, is true under any 0-1 assignment. Therefore, by Proposition 2.23, there is a constant-depth, constant-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the above equation.

Observe that all the equations above can be proved simultaneously. By summing over all these equations, there is a constant-depth, $O(k)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\sum_{i=0}^{k-1} (x_i + y_i) \cdot 2^i = \sum_{i=0}^{k-1} (x_i \oplus y_i \oplus \mathsf{CARRY}_i(\overline{x}, \overline{y})) \cdot 2^i + \mathsf{CARRY}_k(\overline{x}, \overline{y}) \cdot 2^k.$$

We can conclude

$$\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}}) \left|\frac{O(k), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{\mathsf{ADD}_i(\overline{x}, \overline{y})}).$$

□

Lemma 7.15. *Let $\mathbb{F}_q$ be a finite field. Let $\overline{x} = x_{k-1}, \ldots, x_0$ and $\overline{y} = y_{k-1}, \ldots, y_0$ be two k length binary representations. Let z be a Boolean variable. Then,*

$$\mathsf{Modular}^t(\overline{x}, z, \overline{y}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + 2^t z = \mathsf{VAL}(\overline{y}).$$

In other words, from $\mathsf{Modular}^l(\overline{x}, z, \overline{y})$, there is a $O(1)$-depth, $O(k)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{x}) + 2^t z = \mathsf{VAL}(\overline{y})$.

Proof of Lemma 7.15. Recall Definition 7.4, $\mathsf{Modular}^t(\overline{x}, z, \overline{y})$ contains

$$\mathsf{Addition}(0, \overbrace{x_{k-2}, \ldots, x_0}^{\text{the last } k-1 \text{ bit of } \overline{x}}, \underbrace{\overline{m}}_{\text{the } k \text{ length binary representation of } 2^t z + 2^{k-1} x_{k-1} \mod q}, \overline{u}),$$

$$\mathsf{Addition}'(\overbrace{u_{k-1}, \ldots, u_0}^{\text{the last } k \text{ bit of } u}, \underbrace{\overline{w}}_{\text{the } k \text{ length binary representation of } 2^k u_k}, \overline{y}).$$

By Lemma 7.14,

$$\mathsf{Addition}(0, x_{k-2}, \ldots, x_0, \overline{m}, \overline{u}) \left|\frac{O(k), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \sum_{i=0}^{k-2} 2^i x_i + \mathsf{VAL}(\overline{m}) = \mathsf{VAL}(\overline{u}).$$

By Proposition 2.23, there is a constant-depth, constant-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{m}) = 2^t z + 2^{k-1} x_{k-1}$ since there are only two Boolean variables in it after replacing each bit m_i with $\mathsf{arit}(\mathsf{cnf}(f_i^t(z, x_{k-1})))$.

Hence, by adding $\mathsf{VAL}(\overline{m}) = 2^t z + 2^{k-1} x_{k-1}$ and $\sum_{i=0}^{k-2} 2^i x_i + \mathsf{VAL}(\overline{m}) = \mathsf{VAL}(\overline{u})$ together, we have $\mathsf{VAL}(\overline{x}) + 2^t z = \mathsf{VAL}(\overline{u})$.

Now, we aim to show that

$$\mathsf{Addition}'(u_{k-1}, \cdots, u_0, \overline{w}, \overline{y}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{u}) = \mathsf{VAL}(y)$$

Same as the proof of Lemma 7.14, for $0 \le i \le k-2$, there is a constant-depth, constant-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$2^i \cdot (u_i + w_i) = 2^i y_i(u_{k-1}, \ldots, u_0, \overline{w}) + 2^{i+1}\mathsf{CARRY}_{i+1}(u_{k-1}, \ldots, u_0, \overline{w}) - 2^i \mathsf{CARRY}_i(u_{k-1}, \ldots, u_0, \overline{w}).$$

For the kth bit, we aim to prove

$$2^{k-1}(u_{k-1} + w_{k-1}) = 2^{k-1} y_{k-1}(u_{k-1}, \ldots, u_0, \overline{w}) - 2^{k-1}\mathsf{CARRY}_{k-1}(u_{k-1}, \ldots, u_0, \overline{w}),$$

which is

$$2^{k-1}(u_{k-1} + w_{k-1}) = 2^{k-1}(u_{k-1} \oplus w_{k-1} \oplus \mathsf{CARRY}_{k-1}(u_{k-1}, \ldots, u_0, \overline{w})) - 2^{k-1}\mathsf{CARRY}_{k-1}(u_{k-1}, \ldots, u_0, \overline{w}).$$

By replacing u_{k-1}, w_{k-1} and $\mathsf{CARRY}_{k-1}(u_{k-1}, \ldots, u_0, \overline{w})$ by constant-depth, $\mathsf{poly}(k)$-size formulas $U(\overline{x}, y)$, $W(\overline{x}, y)$ and $CARRY(\overline{x}, z)$ correspondingly, we get

$$2^{k-1}(U + W) = 2^{k-1}(U \oplus W \oplus CARRY) - 2^{k-1}CARRY.$$

The above equation holds over all Boolean assignments of $\overline{x}, z$ according to our discussion above. By Proposition 2.23, we get a depth-2, $O(2^k \cdot \mathsf{poly}(k))$-size $\mathsf{IPS}^{\mathsf{alg}}$ of the above equation. By adding the above equations, we have $\sum_{i=0}^{k-1} 2^i u_i + \mathsf{VAL}(\overline{w}) = \mathsf{VAL}(\overline{y})$.

Since each bit in $\overline{w}$ can be replaced by the corresponding $\mathsf{arit}(\mathsf{cnf}(g_i(u_k)))$, there is a constant-depth, constant-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{w}) = 2^k u_k$ since there is only one Boolean variable (i.e. u_k) in the formula after replacing. From $\sum_{i=0}^{k-1} 2^i u_i + \mathsf{VAL}(\overline{w}) = \mathsf{VAL}(\overline{y})$ and $\mathsf{VAL}(\overline{w}) = 2^k u_k$, we get $\mathsf{VAL}(\overline{u}) = \mathsf{VAL}(\overline{y})$.

Now, we can conclude that

$$\mathsf{Modular}^t(\overline{x}, z, \overline{y}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + 2^t z = \mathsf{VAL}(\overline{y}).$$

□

Proof of Lemma 7.13. Recall the Definition 7.5, $\mathsf{CNF\text{-}ADD}(\overline{x}, \overline{y}, \overline{z})$ is as follows:

- Addition step: $\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}})$
- Modular step: $\mathsf{Modular}^k(\mathsf{ADD}_{k-1}, \ldots, \mathsf{ADD}_0, \mathsf{ADD}_k, \overline{\mathsf{ADD}'})$
- Connection step: for each $0 \le i \le k-1$, we have $\mathsf{ADD}'_i = z_i$.

We aim to show

$$\mathsf{CNF\text{-}ADD}(\overline{x}, \overline{y}, \overline{z}) \left|\frac{O(k), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{z}).$$

By Lemma 7.14,

$$\mathsf{Addition}(\overline{x}, \overline{y}, \overline{\mathsf{ADD}}) \left|\frac{O(k), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{\mathsf{ADD}}).$$

By Lemma 7.15,

$$\mathsf{Modular}^k(\mathsf{ADD}_{k-1}, \ldots, \mathsf{ADD}_0, \mathsf{ADD}_k, \overline{\mathsf{ADD}'}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{\mathsf{ADD}}) = \mathsf{VAL}(\overline{\mathsf{ADD}'}).$$

Note that $\mathsf{VAL}(\mathsf{ADD}_{k-1}, \ldots, \mathsf{ADD}_0) + 2^k \mathsf{ADD}_k = \mathsf{VAL}(\overline{\mathsf{ADD}})$.

Since for each $0 \le i \le k-1$, $\mathsf{ADD}'_i = z_i$. It is easy to show that $\mathsf{VAL}(\overline{\mathsf{ADD}'}) = \mathsf{VAL}(\overline{z})$. We can conclude that

$$\mathsf{CNF\text{-}ADD}(\overline{x}, \overline{y}, \overline{z}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) + \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{z}).$$

□

Lemma 7.16. *Let $\mathbb{F}_q$ be a finite field. Let $\overline{x} = x_{k-1}, \dots, x_0$, $\overline{y} = y_{k-1}, \dots, y_0$ and $\overline{z} = z_{k-1}, \dots, z_0$ be three k length binary representations. Then,*

$$\mathsf{CNF\text{-}MULT}(\overline{x}, \overline{y}, \overline{z}) \left|\frac{O(2^k \cdot \mathsf{poly}(k)), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) \cdot \mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{z})$$

Lemma 7.17. *Let $\mathbb{F}_q$ be a finite field. Let $\overline{x} = x_{k-1}, \dots, x_0$ and $\overline{y} = y_{k-1}, \dots, y_0$ be two k length binary representations. Let $\overline{s_{k-1}}, \dots, \overline{s_0}$ be k many binary representations with different length where $\overline{s_i} = s_{i,i+k-1}, \dots, s_{i,0}$ is a $i + k$ length binary representation for $0 \le i \le k-1$. Then,*

$$\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \dots, \overline{s_{k-1}}) \left|\frac{O(k^2), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) \cdot \mathsf{VAL}(\overline{y}) = \sum_{i=0}^{k-1} \mathsf{VAL}(\overline{s_i}).$$

In other words, from $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_{k-1}}, \dots, \overline{s_0})$, there is a $O(1)$-depth, $O(k+i)$ size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{x}) \cdot \mathsf{VAL}(\overline{y}) = \sum_{i=0}^{k-1} \mathsf{VAL}(\overline{s_i})$.

Proof of Lemma 7.17. Recall Definition 7.6, $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \cdots, \overline{s_{k-1}})$ includes

$$\mathsf{MULT}_i(\overline{x}, \overline{y}, \overline{s_i}) := \begin{cases} s_{i,j} = \mathsf{arit}(x_{j-i} \wedge y_i), & i \le j \le k-1+i \\ s_{i,j} = \mathsf{arit}(\bot), & 0 \le j < i. \end{cases}, \qquad 0 \le i \le k-1.$$

First, we aim to show that from each $\mathsf{MULT}_i(\overline{x}, \overline{y}, \overline{s_i})$, there is a $O(k+i)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\mathsf{VAL}(\overline{x}) \cdot 2^i y_i = \mathsf{VAL}(\overline{s_i}).$$

For each j such that $i \le j \le k-1+i$, from $s_{i,j} = x_{j-i} \wedge y_i$ and Boolean axioms, there is a $O(1)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $2^{j-i} x_{j-i} \cdot 2^i y_i = 2^j s_{i,j}$. For each j such that $0 \le j < i$, from $s_{i,j} = 0$, there is a $O(1)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $2^j s_j = 0$. By summing up all these equations, there is a $O(k+i)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\mathsf{VAL}(\overline{x}) \cdot 2^i y_i = \mathsf{VAL}(\overline{s_i}).$$

Again, by summing up all these equations for each MULT_i, there is a $O(k^2)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\sum_{j=0}^{k-1} 2^j x_j \times \sum_{i=0}^{k-1} 2^i y_i = \sum_{i=0}^{k-1} \sum_{w=0}^{k-1+i} 2^w s_{iw}.$$

which is $\mathsf{VAL}(\overline{x}) \cdot \mathsf{VAL}(\overline{y}) = \sum_{i=0}^{k-1} \mathsf{VAL}(\overline{s_i})$. ☐

Proof of Lemma 7.16. Recall the Definition 7.7, $\mathsf{CNF\text{-}MULT}(\overline{x}, \overline{y}, \overline{z})$ is defined as follows:

- Multiplication step: $\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \cdots, \overline{s_{k-1}})$.
- Modular step: for each $\overline{s_i}$ such that $1 \le i \le k-1$, we have the following i many modular formula:

$$\begin{aligned} &\mathsf{Modular}^k(s_{i,k-1}, \dots, s_{i,0}, s_{i,k}, \overline{u_{i,k}}) \\ &\mathsf{Modular}^{k+1}(\overline{u_{i,k}}, s_{i,k+1}, \overline{u_{i,k+1}}) \\ &\vdots \\ &\mathsf{Modular}^j(\overline{u_{i,j-1}}, s_{i,j}, \overline{u_{i,j}}), \qquad k+1 \le j \le i+k-1. \end{aligned}$$

All $\overline{u_{i,j}}$ is a k length binary representation.

- Addition step: Now for each $\overline{s_i}$, we have a k length binary representation $\overline{u_{i,i+k-1}}$ such that $\sum_{j=0}^{i+k-1} 2^j s_{i,j} = \sum_{j=0}^{k-1} 2^j u_{i,i+k-1,j}$ in the finite field $\mathbb{F}_q$. Then, using the CNF encoding of bit arithmetic for addition in a finite field as we defined above in Definition 7.5, we have the following:

$$\begin{aligned}
&\mathsf{CNF\text{-}ADD}(\overline{s_0}, \overline{u_{1,k}}, \overline{v_1})\\
&\mathsf{CNF\text{-}ADD}(\overline{v_1}, \overline{u_{2,k+1}}, \overline{v_2})\\
&\vdots\\
&\mathsf{CNF\text{-}ADD}(\overline{v_i}, \overline{u_{i+1,k+i}}, \overline{v_{i+1}}), \qquad 1 \le i \le k-2
\end{aligned}$$

  where each $\overline{v_i}$ is k length binary representation.

- Connection step: for each $0 \le j \le k-1$, we have $v_{k-1,j} = z_j$.

By Lemma 7.17,

$$\mathsf{MULT}(\overline{x}, \overline{y}, \overline{s_0}, \dots, \overline{s_{k-1}}) \left|\frac{O(k^2), O(1)}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x}) \cdot \mathsf{VAL}(\overline{y}) = \sum_{i=0}^{k-1} \mathsf{VAL}(\overline{s_i}).$$

By Lemma 7.15, for each $\overline{s_i}$, there is a $O(2^k \cdot \mathsf{poly}(k))$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\begin{aligned}
&\mathsf{VAL}(s_{i,k-1}, \dots, s_{i,0}) + 2^k s_{i,k} = \mathsf{VAL}(\overline{u_{i,k}})\\
&\mathsf{VAL}(\overline{u_{i,k}}) + 2^{k+1} s_{i,k+1} = \mathsf{VAL}(\overline{u_{i,k+1}})\\
&\mathsf{VAL}(\overline{u_{i,j-1}}) + 2^j s_{i,j} = \mathsf{VAL}(\overline{u_{i,j}}), \qquad k+1 \le j \le i+k-1.
\end{aligned}$$

By summing up the above equations, there is a $O(2^k \cdot \mathsf{poly}(k))$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{s_i}) = \mathsf{VAL}(\overline{u_{i,i+k-1}})$.

Therefore, there is a $O(2^k \cdot \mathsf{poly}(k))$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{s_i}) = \mathsf{VAL}(\overline{u_{i,i+k-1}}), 1 \le i \le k-1$.

By Lemma 7.13, given the formulas in the addition step:

$$\begin{aligned}
&\mathsf{CNF\text{-}ADD}(\overline{s_0}, \overline{u_{1,k}}, \overline{v_1})\\
&\mathsf{CNF\text{-}ADD}(\overline{v_1}, \overline{u_{2,k+1}}, \overline{v_2})\\
&\vdots\\
&\mathsf{CNF\text{-}ADD}(\overline{v_i}, \overline{u_{i+1,k+i}}, \overline{v_{i+1}}), \qquad 1 \le i \le k-2,
\end{aligned}$$

there is a $O(2^k \cdot \mathsf{poly}(k))$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\begin{aligned}
&\mathsf{VAL}(\overline{s_0}) + \mathsf{VAL}(\overline{u_{1,k}}) = \mathsf{VAL}(\overline{v_1})\\
&\mathsf{VAL}(\overline{v_i}) + \mathsf{VAL}(\overline{u_{i+1,k+i}}) = \mathsf{VAL}(\overline{v_{i+1}}), \qquad 1 \le i \le k-2.
\end{aligned}$$

By adding them together,

$$\mathsf{VAL}(\overline{s_0}) + \sum_{i=0}^{k-1} \mathsf{VAL}(\overline{u_{i,i+k-1}}) = \mathsf{VAL}(\overline{v_{k-1}}).$$

Since we have already proved that $\mathsf{VAL}(\overline{s_i}) = \mathsf{VAL}(\overline{u_{i,i+k-1}})$ for each i such that $1 \le i \le k-1$, by summing up them together with $\mathsf{VAL}(\overline{s_0})$,

$$\begin{aligned}\sum_{i=0}^{k-1}\mathsf{VAL}(\overline{s_0}) &= \mathsf{VAL}(\overline{s_0}) + \sum_{i=1}^{k-1}\mathsf{VAL}(\overline{s_i})\\ &= \mathsf{VAL}(\overline{s_0}) + \sum_{i=0}^{k-1}\mathsf{VAL}(\overline{u_{i,i+k-1}})\\ &= \mathsf{VAL}(\overline{v_{k-1}}).\end{aligned}$$

By Connection step, for each $o \leq j \leq k-1$, $v_{k-1,j} = z_j$. There is a $O(k)$-size, $O(1)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $\mathsf{VAL}(\overline{v_{k-1}}) = \mathsf{VAL}(\overline{z})$.

Now, we can conclude that

$$\mathsf{CNF\text{-}MULT}(\overline{x},\overline{y},\overline{z}) \left|\frac{\;O(2^k\cdot \mathsf{poly}(k)),O(1)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{VAL}(\overline{x})\cdot\mathsf{VAL}(\overline{y}) = \mathsf{VAL}(\overline{z})$$

□

Lemma 7.18 (Translating from extended CNFs to circuit equations)**.** *Let $\mathbb{F}_q$ be a finite field and let k be $2^{k-1} < q < 2^k$, and let $C(\overline{x})$ be a circuit of depth Δ over $\overline{x}$ variables. Then, the following hold*

$$\mathsf{ecnf}(C(\overline{x})=0) \left|\frac{\;O(2^k\cdot \mathsf{poly}(k)|C|),O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. C(\overline{x}) = 0$$

To prove Lemma 7.18, we will first show that from the Extended CNFs of each node g with t many children in $C(\overline{x})$, there is a $O(1)$-depth, $O(2^k\cdot\mathsf{poly}(k)t)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the SLP formula of g. Then, we will show that from $\mathsf{SLP}(C(\overline{x})=0)$, there is a $O(\Delta)$-depth, $O(2^k\cdot\mathsf{poly}(k)|C|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the circuit equation $C(\overline{x}) = 0$.

Proof of Lemma 7.18. First, we aim to show that from the Extended CNFs of each node g with t many children in $C(\overline{x})$, there is a $O(1)$-depth, $O(2^k\cdot\mathsf{poly}(k)t)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the SLP formula of g. We start with the $+$ node case.

Suppose we have the following CNF formulas:

$$\begin{aligned}&\mathsf{CNF\text{-}ADD}(\overline{u_1},\overline{u_2},\overline{v_1^g})\\ &\mathsf{CNF\text{-}ADD}(\overline{u_{i+2}},\overline{v_i^g},\overline{v_{i+1}^g}),\quad 1\leq i\leq t-3\\ &\mathsf{CNF\text{-}ADD}(\overline{u_t},\overline{v_{t-2}^g},\overline{g})\end{aligned}$$

which are the CNF encoding of the node g expressing that $\sum_{i=0}^{t} u_i = g$.

By Lemma 7.13, from there is a $O(kt)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\begin{aligned}&\mathsf{VAL}(\overline{u_1}) + \mathsf{VAL}(\overline{u_2}) = \mathsf{VAL}(\overline{v_1^g})\\ &\mathsf{VAL}(\overline{u_{i+2}} + \mathsf{VAL}(\overline{v_i^g}) = \mathsf{VAL}(\overline{v_{i+1}^g}),\quad 1\leq i\leq t-3\\ &\mathsf{VAL}(\overline{u_t}) + \mathsf{VAL}(\overline{v_{t-2}^g}) = \mathsf{VAL}(\overline{g}).\end{aligned}$$

By Definition 7.10, for $\overline{u_i}$, $\overline{v_i^g}$ and $\overline{g}$, $\mathsf{ecnf}(C(\overline{x})=0)$, $\mathsf{ecnf}(C(\overline{x})=0)$ includes $\mathsf{VAL}(\overline{u_i}) = u_i$, $\mathsf{VAL}(\overline{v_i^g}) = v_i^g$ and $\mathsf{VAL}(\overline{g}) = g$, which are the binary value formulas for them.

Hence, from the Extended CNF encoding of the node g, there is a $O(kt)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of the following SLPs:

$$\begin{aligned}
&u_1 + u_2 = v_1^g \\
&u_{i+2} + v_i^g = v_{i+1}^g \quad 1 \le i \le t-3 \\
&u_t + v_{t-2}^g = g
\end{aligned}$$

By summing up all the SLP formulas,

$$u_1 + u_2 + \cdots + u_t = g.$$

For the $\times$ nodes case, suppose we have the following CNF formulas:

$$\begin{aligned}
&\mathsf{CNF\text{-}MULT}(\overline{u_1}, \overline{u_2}, \overline{v_1^g}) \\
&\mathsf{CNF\text{-}MULT}(\overline{u_{i+2}}, \overline{v_i^g}, \overline{v_{i+1}^g}), \quad 1 \le i \le t-3 \\
&\mathsf{CNF\text{-}MULT}(\overline{u_t}, \overline{v_{t-2}^g}, \overline{g})
\end{aligned}$$

which are the CNF encoding of the node g expressing that $\prod_{i=0}^{t} u_i = g$.

By Lemma 7.16, from the CNF encoding of the node g, there is a $O(2^k \cdot \mathsf{poly}(k)t)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of

$$\begin{aligned}
&\mathsf{VAL}(\overline{u_1}) \times \mathsf{VAL}(\overline{u_2}) - \mathsf{VAL}(\overline{v_1^g}) = 0 \\
&\mathsf{VAL}(\overline{u_{i+2}}) \times \mathsf{VAL}(\overline{v_i^g}) - \mathsf{VAL}(\overline{v_{i+1}^g}) = 0, \quad 1 \le i \le t-3 \\
&\mathsf{VAL}(\overline{u_t}) \times \mathsf{VAL}(\overline{v_{t-2}^g}) - \mathsf{VAL}(\overline{g}) = 0.
\end{aligned}$$

By Definition 7.10, $\mathsf{ecnf}(C(\overline{x}) = 0)$ includes the binary value formulas for those binary representations.

Hence, from the Extended CNF encoding of the node g, there is a $O(kt)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of the following SLPs:

$$\begin{aligned}
&u_1 \times u_2 - v_1^g = 0 \\
&u_{i+2} \times v_i^g - v_{i+1}^g = 0, \quad 1 \le i \le t-3 \\
&u_t \times v_{t-2}^g - g = 0
\end{aligned}$$

Then, by the following depth-2 $O(t)$-size formula

$$(u_1 \times u_2 - v_1^g) \times \prod_{j=3}^{t} u_j + \cdots + (u_{i+2} \times v_i^g - v_{i+1}^g) \times \prod_{j=i+3}^{t} u_j + \cdots + (u_t \times v_{t-2}^g - g) = 0,$$

there is a $O(2^k \cdot \mathsf{poly}(k)t)$-size, $O(1)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $u_1 \times \cdots \times u_t - g = 0$.

For the output node, by the axioms $g_{out,i} = q_i$ and binary value formulas for $\overline{g_{out}}$ and $\overline{q}$, $g_{out} = \mathsf{VAL}(g_{out}) = \mathsf{VAL}(\overline{q}) = 0$ can be easily derived.

Therefore, from $\mathsf{ecnf}(C(\overline{x}) = 0)$, there is a $O(1)$-depth, $O(2^k \cdot \mathsf{poly}(k)|C|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of SLP formulas of $C(\overline{x}) = 0$.

Now, we aim to show that from these SLP formulas, there is $O(\Delta)$-depth, $O(2^k \cdot \mathsf{poly}(k)|C|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of the circuit equation $C(\overline{x}) = 0$. We prove this by induction that for each node g, there is a $O(\mathsf{Depth}(g))$-depth, $O(2^k \cdot \mathsf{poly}(k)|C_g|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of a circuit equation $C_g(\overline{x}) - g = 0$:

- Base case: Since we have the binary value formulas for all the leaves, this is immediate.
- Addition case: suppose we have a SLP formula $u_1+u_2+\cdots+u_t-g=0$ where for each u_i, there is a $O(\mathsf{Depth}(u_i))$-depth, $O(2^k\cdot\mathsf{poly}(k)|C_{u_i}|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of a circuit equation $C_{u_i}(\overline{x})-u_i=0$. Then, by summing up all the circuit equations $C_{u_i}-u_i$ together with $u_1+u_2+\cdots+u_t=g$, there is a $O(\mathsf{Depth}(g))$-depth, $O(2^k\cdot\mathsf{poly}(k)|C_g|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of a circuit equation $\sum_{i=1}^t C_{u_i}-g=0$ as this proof adds one depth and one node.
- Multiplication case: suppose we have a SLP formula $u_1\times u_2\times\cdots\times u_t-g=0$ where for each u_i, there is a $O(\mathsf{Depth}(u_i))$-depth, $O(2^k\cdot\mathsf{poly}(k)|C_{u_i}|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of a circuit equation $C_{u_i}(\overline{x})-u_i=0$. Then, computes $(\sum_{i=1}^t((C_{u_i}-u_i)\times\prod_{j=1}^{i-1}C_{u_j}\prod_{l=i+1}^t u_l))+(u_1\times u_2\times\cdots\times u_t-g)$, which adds two depths and two nodes. This gives a $O(\mathsf{Depth}(g))$-depth, $O(2^k\cdot\mathsf{poly}(k)|C_g|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of a circuit equation $\prod_{i=0}^t C_{u_i}-g=0$.
- Output node: given SLP formula $g_{out}=0$ and circuit equations $C_{g_{out}}-g_{out}=0$, it is easy to get $C_{g_{out}}=0$.

Now, we can conclude that

$$\mathsf{ecnf}(C(\overline{x})=0)\,\Big|\!\frac{\;O(2^k\cdot\mathsf{poly}(k)|C|),O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\; C(\overline{x})=0$$

□

Lemma 7.19. *Let $\mathbb{F}_q$ be a finite field, and let $C(\overline{x})$ be a circuit of depth Δ over $\overline{x}$ variables. Then, the following hold*

$$\mathsf{SLP}(C(\overline{x})),C(\overline{x})=0\,\Big|\!\frac{\;O(|C|),O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\;\mathsf{SLP}(C(\overline{x})=0)$$

Proof of Lemma 7.19. We aim to show that given $\mathsf{SLP}(C(\overline{x}))$ and $C(\overline{x})$, there is a $O(\Delta)$-depth, $O(|C|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ proof of $g_{out}=0$. By the proof of Lemma 7.19, given $\mathsf{SLP}(C(\overline{x}))$, there is a $O(|C|)$-size, $O(\Delta)$-depth $\mathsf{IPS}^{\mathsf{alg}}$ proof of $C_{g_{out}}-g_{out}=0$ where $C_{g_{out}}$ is exactly the same as $C(\overline{x})$. $g_{out}=0$ can be obtained by a simple subtraction. □

Proposition 7.20. *Let $\overline{x}=x_{k-1},\ldots,x_0$ be a k-length binary representation, where x is an algebraic variable.*

$$\{x_i^2-x_i=0:\quad 0\le i\le k-1\},$$
$$x=\mathsf{VAL}(\overline{x}),$$
$$x=0\,\Big|\!\frac{\;O(2^k\cdot\mathsf{poly}(k)),O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\;\{x_i=q_i:\quad 0\le i\le k-1\}$$

Proposition 7.20 follows from Proposition 2.23.

Lemma 7.21 (Translating from circuit equations and addition axioms to extended CNFs)**.** *Let $\mathbb{F}_q$ be a finite field and let k be $2^{k-1}<q<2^k$, and let $C(\overline{x})$ be a circuit of depth Δ over $\overline{x}$ variables. Then, the following holds*

$$\{x_i^2-x_i=0: x_i \textit{ is a binary variable in } \mathsf{ecnf}(C(\overline{x})=0)\},$$
$$\textit{All formulas in } \mathsf{ecnf}(C(\overline{x})=0) \textit{ except the connection formula for the output node,}$$
$$\mathsf{SLP}(C(\overline{x})),$$
$$C(\overline{x})=0\,\Big|\!\frac{\;O(2^k\cdot\mathsf{poly}(k)|C|),O(1)\;}{\mathsf{IPS}^{\mathsf{alg}}}\;\mathsf{ecnf}(C(\overline{x})=0).$$

Proof of Lemma 7.21. Note that the only formulas required to derive is the connection formula for the output node:

$$\{g_{out,i} = q_i : \quad 0 \le i \le k-1\}.$$

By Lemma 7.19, there is $O(\Delta)$-depth, $O(|C|)$-size $\mathsf{IPS}^{\mathsf{alg}}$ of $g_{out} = 0$. Together with additional axioms and Boolean axioms we already have, we have all the axioms needed in Proposition 7.20. By Proposition 7.20, there is a $O(1)$-depth, $O(2^k \cdot \mathsf{poly}(k))$ $\mathsf{IPS}^{\mathsf{alg}}$ proof of $g_{out,i} = q_i$. □

Lemma 7.22 (Translating between extended CNFs and circuit equations)**.** *Let $\mathbb{F}_q$ be a finite field and let k be $2^{k-1} < q < 2^k$, and let $C(\overline{x})$ be a circuit of depth Δ over $\overline{x}$ variables. Then, the following holds*

$$\mathsf{ecnf}(C(\overline{x}) = 0) \left|\frac{\;O(2^k \cdot \mathsf{poly}(k)|C|), O(\Delta)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. C(\overline{x}) = 0$$

and

$$\{x_i^2 - x_i = 0 : x_i \text{ is a binary variable in } \mathsf{ecnf}(C(\overline{x}) = 0)\},$$
$$\textit{All formulas in } \mathsf{ecnf}(C(\overline{x}) = 0) \textit{ except the connection formula for the output node,}$$
$$\mathsf{SLP}(C(\overline{x})),$$
$$C(\overline{x}) = 0 \left|\frac{\;O(2^k \cdot \mathsf{poly}(k)|C|), O(1)\;}{\mathsf{IPS}^{\mathsf{alg}}}\right. \mathsf{ecnf}(C(\overline{x}) = 0).$$

For an instance of size s, we take the characteristics of the finite field to be a prime number between n^3 and $(n+1)^3$ that must exist for sufficiently large instances according to [Che10]. Let k be the number of bits needed for the binary representation of field elements in $\mathbb{F}_q$. In other words, $2^{k-1} \le q < 2^k$. We let $N = \sum_{j=0}^{l} \binom{n^2+j-1}{j} = 2^{O(n^2+l)}$ be the number of different monomials over n^2 variables and degree at most l.

- $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$: circuit equations expressing that there is a constant-depth universal circuit for size s and depth Δ circuits that compute the Permanent polynomial of dimension n, which means there are n^2 many variables, over degree l.
  - Type: circuit equations;
  - Number of variable: $K_{s,\Delta}$ which is $\mathsf{poly}(s, \Delta)$;
  - Size: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\varphi^{\mathsf{cnf}}_{n,s,l,\Delta}$: the CNF encoding of $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ based on definition Definition 7.9
  - Type: CNF formulas;
  - Number of variable: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$;
  - Size: $O(k \cdot 2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\varphi^{\mathsf{ecnf}}_{n,s,l,\Delta}$: the Extended CNF encoding of $\mathsf{VNP} = \mathsf{VAC}^0(n, s, l, \Delta)$ based on definition Definition 7.10
  - Type: Extended CNF formulas;
  - Number of variable: $O(2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$;
  - Size: $O(k \cdot 2^{(n+l)l} \cdot \mathsf{poly}(s, \Delta) \cdot N)$.
- $\mathsf{IPS}_{\mathsf{ref}}(t, \Delta, l, \overline{\mathcal{F}})$: circuit equations expressing that there exists a constant-depth universal circuit for size t and depth Δ circuits that computes the IPS refutation of $\mathcal{F}$ over degree l.

  - Type: circuit equations;
  - Number of variable: $K_{t,\Delta}$ which is $\mathsf{poly}(t,\Delta)$;
  - Size: $O(2^{(n+l)l} \cdot \mathsf{poly}(t,\Delta) \cdot |\mathcal{F}| \cdot N)$.

- $\varphi^*_{n,s,l,\Delta}$: the Extended CNF encoding $\varphi^{\mathsf{ecnf}}_{n,s,l,\Delta}$ together with additional extension axioms for $\mathsf{IPS}_{\mathsf{ref}}(s,\Delta,l,\mathcal{F})$ includes:

  $\{x_i^2 - x_i = 0 : x_i$ is a binary variable in $\mathsf{ecnf}(\mathsf{IPS}_{\mathsf{ref}}(s,\Delta,l,\mathcal{F}))\}$,
  All formulas in $\mathsf{ecnf}(\mathsf{IPS}_{\mathsf{ref}}(s,\Delta,l,\mathcal{F}))$ except the connection formula for the output node,
  $\mathsf{SLP}(\mathsf{IPS}_{\mathsf{ref}}(s,\Delta,l,\mathcal{F}))$ excepts the SLP formulas for output nodes

  - Type: Extended CNF formulas;
  - Number of variable: $O(q \cdot 2^{(n+l)l} \cdot \mathsf{poly}(s,\Delta) \cdot N)$;
  - Size: $O(k \cdot 2^{(n+l)l} \cdot \mathsf{poly}(s,\Delta) \cdot N)$.

- $\Phi_{t,l,\Delta',n,s,\Delta}$: the CNF encoding of $\mathsf{IPS}_{\mathsf{ref}}(t,\Delta',l,\varphi^*_{n,s,l,\Delta})$ expressing that IPS refutes $\varphi^*_{n,s,l,\Delta}$ in size t, depth Δ' and degree l.

  - Type: CNF formulas;
  - Number of variable: $K_{t,\Delta}$ which is $\mathsf{poly}(t,\Delta)$;
  - Size: $O(k \cdot 2^{(n+l)l} \cdot \mathsf{poly}(t,\Delta) \cdot |\mathcal{F}| \cdot N)$.

By replacing the use of Lemma 2.35 and Lemma 2.36, which is the translation lemma for fixed finite fields, with the above Lemma 7.22, which is the translation lemma for polynomial-size finite fields, in the proof of Theorem 5.3, we can get the corollary below. We fix $l : \mathbb{N} \to \mathbb{N}$ to be a (monotone) size function $l(n) = n^\epsilon$ for some constant ϵ.

Corollary 7.23 (Main theorem in polynomial-size finite fields)**.** *The* CNF *family* $\{\Phi_{t,l,\Delta',n,s,\Delta}\}_n$ *does not have polynomial-size* IPS *refutations infinitely often over* $\mathbb{F}_q$*, for every prime* q *such that* $q < (|\varphi^*_{n,s,l,\Delta}|+1)^3$*, in the following sense: for every constant* Δ *there exists a constant* c_1*, a constant* Δ' *such that for every sufficiently large constant* c_2 *and every constant* Δ'' *and every constant* c_0*, for infinitely many* $n, t(n), s(n) \in \mathbb{N}$*,* $t(n) > |\varphi^*_{n,s,l,\Delta}|^{c_1}$ *and* $n^{c_1} < s(n) < n^{c_2}$*,* $\Phi_{t,l,\Delta',n,s,\Delta}$ *has no* IPS *refutation of size at most* $|\Phi_{t,l,\Delta',n,s,\Delta}|^{c_0}$ *and depth at most* Δ''*.*

Corollary 7.23 is a formal statement of Theorem 1.7 in the Introduction, which can be obtained from it by setting l, s, t to appropriate polynomial functions of n, and by setting $d = \Delta, d' = \Delta', d'' = \Delta''$.

References

[ABRW04] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM J. Comput.*, 34(1):67–88, 2004. (A preliminary version appeared in Proceedings of the 41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000)). 1.4

[ABSRW02] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM J. Comput.*, 31(4):1184–1211 (electronic), 2002. 2.19

[AF22] Robert Andrews and Michael A. Forbes. Ideals, determinants, and straightening: proving and using lower bounds for polynomial ideals. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, page 389–402, New York, NY, USA, 2022. Association for Computing Machinery. 1.2, 1.4

[AGHT24] Yaroslav Alekseev, Dima Grigoriev, Edward A. Hirsch, and Iddo Tzameret. Semialgebraic proofs, IPS lower bounds, and the τ-conjecture: Can a natural number be negative? *SIAM J. Comput.*, 53(3):648–700, 2024. 1.3.2, 2.3, 2.3

[Ajt88] Miklós Ajtai. The complexity of the pigeonhole principle. In *Proceedings of the IEEE 29th Annual Symposium on Foundations of Computer Science*, pages 346–355, 1988. 1.1

[Ajt94] Miklós Ajtai. The independence of the modulo p counting principles. *Electronic Colloquium on Computational Complexity, ECCC*, (Report no.: TR94-014), December 1994. 1.1

[BDS24] CS Bhargav, Sagnik Dutta, and Nitin Saxena. Improved lower bound, and proof barrier, for constant depth algebraic circuits. *ACM Transactions on Computation Theory*, 16(4):1–22, 2024. 2.1, 2.12

[BIK$^+$96a] Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. *Proc. London Math. Soc. (3)*, 73(1):1–26, 1996. 1.1, 1.2

[BIK$^+$96b] Samuel R. Buss, Russell Impagliazzo, Jan Krajíček, Pavel Pudlák, Alexander A. Razborov, and Jiří Sgall. Proof complexity in algebraic systems and bounded depth Frege systems with modular counting. *Computational Complexity*, 6(3):256–298, 1996. 1.1, 1.2

[BLRS25] Amik Raj Behera, Nutan Limaye, Varun Ramanathan, and Srikanth Srinivasan. New bounds for the ideal proof system in positive characteristic. In *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, Aarhus, Denmark, July 2025. To appear. 1.4

[BP98] Paul Beame and Toniann Pitassi. Propositional proof complexity: past, present, and future. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, (65):66–89, 1998. 1, 1.1

[BPR97] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. *The Journal of Symbolic Logic*, 62(3):708–728, 1997. 1.4

[BPR00] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for Frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000. 1.4

[Bus87] Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *The Journal of Symbolic Logic*, (52):916–927, 1987. 1.3.2

[CEI96] Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 174–183, New York, 1996. ACM. 1.1, 2.18

[Che10] Yuan-You Fu-Rui Cheng. Explicit estimate on primes between consecutive cubes. *The Rocky Mountain Journal of Mathematics*, pages 117–153, 2010. 7.1

[Coo85] Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64(1-3):2–21, 1985. 6.1

[CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979. 1, 1.2, 2.15

[EGLT25] Tal Elbaz, Nashlen Govindasamy, Jiaqi Lu, and Iddo Tzameret. Lower bounds against the ideal proof system in finite fields. *arXiv preprint arXiv:2506.17210*, 2025. 1.2, 1.4, 2.4, 2.35, 2.36

[For24] Michael A Forbes. Low-depth algebraic circuit lower bounds over any field. In *39th Computational Complexity Conference (CCC 2024)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2024. 1.2, 1.3, 1.3.1, 1.4, 2.1, 2.13, 2.14, 7

[Fri79] Harvey Friedman. On the consistency, completeness and correctness problems. Unpublished, 1979. 2

[FSTW21] Michael A. Forbes, Amir Shpilka, Iddo Tzameret, and Avi Wigderson. Proof complexity lower bounds from algebraic circuit complexity. *Theory Comput.*, 17:1–88, 2021. 1.2

[GGL$^+$25] Michal Garlik, Svyatoslav Gryaznov, Jiaqi Lu, Rahul Santhanam, and Iddo Tzameret. Metamathematics of algebraic circuit lower bounds. Manuscript, 2025. 1.3.1, 1.3, 1.4, 6.1, 6.1, 6.3, 6.5, 6.6, 6.2, 6.11

[GGPS23] Nicola Galesi, Joshua A. Grochow, Toniann Pitassi, and Adrian She. On the algebraic proof complexity of tensor isomorphism. In *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:35. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023. 6.1

[GGRT25] Michal Garlik, Svyatoslav Gryaznov, Hanlin Ren, and Iddo Tzameret. The weak rank principle: Lower bounds and applications. Manuscript, 2025. 1.5, 1.3.1, 6.1, 6.8, 6.10

[GHT22] Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Tzameret. Simple hard instances for low-depth algebraic proofs. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 188–199. IEEE, 2022. 1.2, 1.4

[Goe90] Andreas Goerdt. Cutting plane versus Frege proof systems. In Egon Börger, Hans Kleine Büning, Michael M. Richter, and Wolfgang Schönfeld, editors, *Computer Science Logic, 4th Workshop, CSL '90, Heidelberg, Germany, October 1-5, 1990, Proceedings*, volume 533 of *Lecture Notes in Computer Science*, pages 174–194. Springer, 1990. 1.3.2

[GP18] Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing: The ideal proof system. *J. ACM*, 65(6):37:1–37:59, 2018. 1.2, 1.3, 1.3, 1.3, 1.3.1, 1.3.1, 1.3.1, 1.3.3, 1.3.3, 2.20, 2.21, 2.3, 2.24, 2.25, 5.6

[Hak85] Armin Haken. The intractability of resolution. *Theoret. Comput. Sci.*, 39(2-3):297–308, 1985. 1.1

[Hås90] Johan Håstad. Tensor rank is NP-complete. *Journal of algorithms*, 11(4):644–654, 1990. 6.1

[HLT24] Tuomas Hakoniemi, Nutan Limaye, and Iddo Tzameret. Functional lower bounds in algebraic proofs: Symmetry, lifting, and barriers. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 1396–1404, New York, NY, USA, 2024. Association for Computing Machinery. Full version in ECCC https://eccc.weizmann.ac.il/report/2024/079/. 1.2, 1.4

[IMP20] Russell Impagliazzo, Sasank Mouli, and Toniann Pitassi. The surprising power of constant depth algebraic proofs. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 591–603. ACM, 2020. 1.3.2

[KB09] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 6.1

[KPW95] Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle. *Random Structures & Algorithms*, 7(1):15–39, 1995. 1.1

[Kra97] Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *The Journal of Symbolic Logic*, 62(2):457–486, 1997. 1.4

[Kra04a] Jan Krajíček. Diagonalization in proof complexity. *Fundamenta Mathematicae*, 182:181–192, 2004. 1.3

[Kra04b] Jan Krajíček. Dual weak pigeonhole principles, pseudo-surjective functions, and provability of circuit lower bounds. *Journal of Symbolic Logic*, 69(1):265–286, 2004. 1.3, 1.3.1, 1.4

[Kra09] Jan Krajíček. A proof complexity generator. In *Proc. from the 13th International Congress of Logic, Methodology and Philosophy of Science (Beijing, August 2007)*, Studies in Logic and the Foundations of Mathematics. King's College Publications, London, 2009. 6.1

[Kra11] Jan Krajíček. On the proof complexity of the Nisan-Wigderson generator based on a hard NP ∩ coNP function. *J. Math. Log.*, 11(1):11–27, 2011. 5

[Kra19] Jan Krajíček. *Proof complexity*, volume 170. Cambridge University Press, 2019. 1

[LST25] Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. *Journal of the ACM*, 72(4):1–35, 2025. 1.2, 1.3, 1.3, 1.3.1, 1.3.3, 1.4, 2.11, 2.14, 6.1, 7

[Mul86] Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 338–339, 1986. 6.1

[PBI93] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *computational complexity*, 3(2):97–140, 1993. 1.1

[Pit97] Toniann Pitassi. Algebraic propositional proof systems. In Neil Immerman and Phokion G. Kolaitis, editors, *Descriptive Complexity and Finite Models, Proceedings of a DIMACS Workshop 1996, Princeton, New Jersey, USA, January 14-17, 1996*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 215–244, Providence, RI, 1997. American Mathematical Society. 1.2

[PS19] Jan Pich and Rahul Santhanam. Why are proof complexity lower bounds hard? In *60th Annual IEEE Symposium on Foundations of Computer Science FOCS 2019, November 9-12, 2019, Baltimore, Maryland USA*, 2019. 1.4

[PS24] Ján Pich and Rahul Santhanam. Learning algorithms versus automatability of frege systems. *Journal of Mathematical Logic*, 2024. 1.4, 7

[Pud86] Pavel Pudlak. On the length of proofs of finitistic consistency statements in first order theories. *Studies in Logic and the Foundations of Mathematics*, 120:165–196, 1986. 2

[Pud87] Pavel Pudlak. Improved bounds to the length of proofs of finite consistency statements. *Contemporary Mathematics*, 65:309–331, 1987. 2

[Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, Sept. 1997. 1.4

[Raz87] Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987. 1.1

[Raz95a] Alexander A. Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Clote, P., Remmel, J., eds. Feasible Mathematics II. Progress in Computer Science and Applied Logic*, volume 13, pages 344–86. Birkhauser, 1995. 1.3, 1.3, 1.4

[Raz95b] Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izv. Ross. Akad. Nauk Ser. Mat.*, 59(1):201–224, 1995. 1.3, 1.3

[Raz96] Alexander A. Razborov. Lower bounds for propositional proofs and independence results in bounded arithmetic. In Friedhelm Meyer auf der Heide and Burkhard Monien, editors, *Automata, Languages and Programming, 23rd International Colloquium, ICALP96, Paderborn, Germany, 8-12 July 1996, Proceedings*, volume 1099 of *Lecture Notes in Computer Science*, pages 48–62. Springer, 1996. 1.3.1, 1.4

[Raz98] Alexander A. Razborov. Lower bounds for the polynomial calculus. *Comput. Complexity*, 7(4):291–324, 1998. 1.3, 1.3

[Raz04] Ran Raz. Resolution lower bounds for the weak pigeonhole principle. *J. ACM*, 51(2):115–138, 2004. 1.3

[Raz10] Ran Raz. Elusive functions and lower bounds for arithmetic circuits. *Theory of Computing*, 6(1):135–177, 2010. 3, 3.1, 3.3, 6.2

[Raz15] Alexander A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, 181:415–472, 2015. 1.3, 1.3, 1.3.1, 5, 1.4

[Raz16a] Alexander Razborov. Propositional proof complexity: Fifteen (or so) years after. Talk at "A Celebration of Mathematics and Computer Science. Celebrating Avi Wigderson's 60th Birthday October 5 - 8, 2016". `https://youtu.be/7LfW6VTW8zo?t=2722`, 2016. 5

[Raz16b] Alexander A. Razborov. Guest column: Proof complexity and beyond. *SIGACT News*, 47(2):66–86, 2016. 1.1, 1.3.1

[Raz21] Alexander Razborov. P, NP and proof complexity. Talk at "SAT and Foundations of Mathematics", Simons Institute. `https://youtu.be/xx4mxcqAl5A?t=2333`, 2021. 5

[Shi16] Yaroslav Shitov. How hard is the tensor rank? *arXiv preprint arXiv:1611.01559*, 2016. 6.1

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the Annual ACM Symposium on the Theory of Computing 1987*, pages 77–82, 1987. 1.1

[ST25] Rahul Santhanam and Iddo Tzameret. Iterated lower bound formulas: A diagonalization-based approach to proof complexity. *SIAM Journal on Computing*, pages 313–349, 2025. Preliminary version appeared in Proceedings of the *53rd Annual ACM Symposium on Theory of Computing (STOC 2021)*. 1.2, 1.3, 1.3, 1.3, 1.3, 1.3.2, 1.3.3, 1.4, 2.4, 2.27, 2.28, 2.29, 2.30, 2.32, 2.33, 2.4, 3, 3.2, 4.1, 4.2, 4, 5.1, 6.2, 7.1

[SU04] Michael Soltys and Alasdair Urquhart. Matrix identities and the pigeonhole principle. *Arch. Math. Log.*, 43(3):351–358, 2004. 6.1

[Val79] Leslie G. Valiant. Completeness classes in algebra. In *Proceedings of the 11th Annual ACM Symposium on the Theory of Computing*, pages 249–261. ACM, 1979. 2.3, 2.4, 2.6, 2.9, 5

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il