Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 142 (2019)

Semi-Algebraic Proofs, IPS Lower Bounds and the τ-Conjecture: Can a Natural Number be Negative?

Yaroslav Alekseev* Dima Grigoriev† Edward A. Hirsch‡ Iddo Tzameret§

Abstract

We introduce the *binary value principle* which is a simple subset-sum instance expressing that a natural number written in binary cannot be negative, relating it to central problems in proof and algebraic complexity. We prove conditional superpolynomial lower bounds on the Ideal Proof System (IPS) refutation size of this instance, based on a well-known hypothesis by Shub and Smale about the hardness of computing factorials, where IPS is the strong algebraic proof system introduced by Grochow and Pitassi [*J. ACM*, 65(6):37:1–55, 2018]. Conversely, we show that short IPS refutations of this instance bridge the gap between sufficiently strong algebraic and semi-algebraic proof systems. Our results extend to unrestricted IPS the paradigm introduced in Forbes, Shpilka, Tzameret and Wigderson [*Theory Comput.*, 17:1–88, 2021] whereby lower bounds against subsystems of IPS were obtained using restricted algebraic circuit lower bounds, and demonstrate that the binary value principle captures the advantage of semi-algebraic over algebraic reasoning, for sufficiently strong systems. Specifically, we show the following:

Conditional IPS lower bounds: The Shub–Smale hypothesis [*Duke Math. J.*, 81:47-54, 1995] implies a superpolynomial lower bound on the size of IPS refutations of the binary value principle over the rationals defined as the unsatisfiable linear equation $\sum_{i=1}^{n} 2^{i-1}x_i = -1$, for Boolean x_i's. Further, the related and more widely known τ-conjecture [*Duke Math. J.*, 81:47-54, 1995] implies a superpolynomial lower bound on the size of IPS refutations of a variant of the binary value principle over the ring of rational functions. No prior conditional lower bounds were known for IPS or apparently weaker propositional proof systems such as Frege systems (though our lower bounds do not translate to Frege lower bounds since the hard instances are not Boolean formulas).

Algebraic vs. semi-algebraic proofs: Admitting short refutations of the binary value principle is necessary for any algebraic proof system to fully simulate any known semi-algebraic proof system, and for strong enough algebraic proof systems it is also *sufficient*. In particular, we introduce a very strong proof system that simulates all known semi-algebraic proof systems (and most other known concrete propositional proof systems), under the name Cone Proof System (CPS), as a semi-algebraic analogue of the Ideal Proof System: CPS establishes the unsatisfiability of collections of polynomial equalities and inequalities over the reals, by representing *sum-of-squares*

*Steklov Institute of Mathematics at St. Petersburg, St. Petersburg, Russia, and Chebyshev Laboratory at St. Petersburg State University. The research was supported in part by Russian Science Foundation (project 16-11-10123).

†CNRS, Mathematiques, Universite de Lille, Villeneuve d'Ascq, 59655, France. http://en.wikipedia.org/wiki/Dima_Grigoriev

‡Department of Computer Science, Technion, Haifa, Israel; partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC. Part of this work has been done when affiliated with Steklov Institute of Mathematics at St. Petersburg, St. Petersburg, Russia, supported in part by Russian Science Foundation (project 16-11-10123). https://edwardahirsch.github.io/edwardahirsch/

§Department of Computing, Imperial College London. iddo.tzameret@gmail.com. http://www.doc.ic.ac.uk/~itzamere. Some parts of this work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 101002742).

ISSN 1433-8092

proofs (and extensions) as algebraic circuits. We prove that IPS polynomially simulates CPS iff IPS admits polynomial-size refutations of the binary value principle (for the language of systems of equations that have no 0/1-solutions), over both $\mathbb{Z}$ and $\mathbb{Q}$.

1 Introduction

This work connects three separate objects of study in computational complexity: algebraic proof systems, semi-algebraic proof systems and algebraic circuit complexity. The connecting point is a subset-sum instance expressing that the value of a natural number given in binary is nonnegative. We will show that this instance captures the advantage of semi-algebraic reasoning over algebraic reasoning in the regime of sufficiently strong proof systems, and is expected to be hard even for very strong algebraic proof systems. We begin with a general discussion about proof complexity, and then turn to algebraic and semi-algebraic proofs, their inter-relations, and the connection between circuit lower bounds and proof-size lower bounds.

Narrowly construed, proof complexity can be viewed as a stratification of the NP vs. coNP question, whereby one aims to understand the complexity of stronger and stronger propositional proof systems as a gradual approach towards separating NP from coNP (and hence, also P from NP). This mirrors circuit complexity in which different circuit classes are analyzed in the hope to provide general super-polynomial circuit lower bounds. Broadly understood however, proof complexity serves as a way to study the computational resources required in different kind of reasoning, different algorithmic techniques and constraint solvers, as well as providing propositional analogues to weak first-order theories of arithmetic.

Algebraic proof systems have attracted immense amount of work in proof complexity, due to their simple nature, being a way to study the complexity of computer-algebra procedures such as the Gröbner basis algorithm, and their connection to different fragments of propositional proof systems with counting gates. Beginning with the fairly weak Nullstellensatz refutation system by Beame *et al* [4] and culminating in the very strong Ideal Proof System by Grochow and Pitassi [28], many algebraic proof systems and variants have been studied. In such systems one basically operates with polynomial equations over a field using simple algebraic derivation rules such as additions of equations and multiplication of an equation by a variable, where variables are usually meant to range over $\{0, 1\}$ values.

Impagliazzo, Pudlák and Sgall [33], following Razborov [51], showed that Polynomial Calculus, which is the standard dynamic algebraic proof system introduced in [13], requires exponential-size refutations (namely, those using an exponential number of monomials) for the simple symmetric unsatisfiable subset-sum instance $x_1 + \cdots + x_n = n + 1$. Note that refuting (that is, showing the unsatisfiability of) a linear equation $\sum_i \alpha_i x_i = \beta$ in which the variables x_i are Boolean, establishes that there is no subset of the α_i numbers that sums up to β, and hence is considered to be a refutation of a subset-sum instance. Forbes, Shpilka, Tzameret and Wigderson [19] showed that variants of this symmetric subset-sum instance are hard for different subsystems of the very strong IPS algebraic proof system, that is, when IPS refutations are written using various restricted algebraic circuit classes. Loosely speaking, IPS is a static Nullstellensatz refutation in which proof-size is measured by algebraic circuit complexity instead of sparsity (that is, monomial size). In other words, IPS proofs are written as algebraic circuits, and thus can tailor the advantage that algebraic circuits have over sparse polynomials (somewhat reminiscent to the way Extended Frege can tailor the full strength of Boolean circuits in comparison to resolution which operates merely with clauses).

The realm of semi-algebraic proof systems has emerged as an equally fruitful subject as algebraic proofs. Semi-algebraic proofs have been brought to the attention of complexity theory from optimization [39, 38] by the works of Pudlák [48] and Grigoriev and Vorobojov [27] (cf. [26]), and more

recently, through their connection to approximation algorithms with the work of Barak *et al.* [3] (see for example [42] and the new excellent survey by Fleming, Kothari and Pitassi [18]). While algebraic proofs derive polynomials in the ideal of a given initial set of polynomials, semi-algebraic proofs extend it to allow deriving polynomials also in the cone of the initial polynomials (informally a cone is an "ideal that preserves positive signs"), hence potentially utilizing a stronger kind of reasoning. In particular [3] considered the *sum-of-squares* (SoS) refutation system. What makes SoS important, for example to polynomial optimization, is the fact that the existence of a degree-d SoS certificate can be formulated as the feasibility of a semidefinite program (SDP), and hence can essentially be solved in polynomial time (though, see O'Donnel [41] and subsequently Raghavendra-Weitz [50] about cases in which the polynomial-time automatability of SoS does not apply).

Berkholz [5] showed interestingly that in the regime of *weak* proof systems, even static semi-algebraic proofs, such as SoS, can simulate dynamic algebraic proof systems such as Polynomial Calculus. Grigoriev [24] showed that in this weak regime semi-algebraic proofs are in fact strictly stronger (with respect to degrees and size) than algebraic proofs, where the separating instances are simple polynomials (for example, symmetric subset sum instances). However, the question of whether semi-algebraic systems can simulate stronger algebraic systems has not been considered before, to the best of our knowledge.

Another established tradition in proof complexity is to seek synergies between proofs and circuit lower bounds. In particular, *proofs-to-circuits* transformations in the form of feasible interpolation, and other close concepts have been pivotal in the search for proof complexity lower bounds, as well as in circuit lower bounds themselves (see Göös, Kamath, Robere and Sokolov [21] for a recent example). In fact, the conception of IPS itself was motivated by the attempt to show that very strong proof complexity lower bounds would result in algebraic complexity class separations such as $\mathsf{VP} \neq \mathsf{VNP}$ (see [28, 53] and the survey [47]). Li, Tzameret and Wang [36] as well as Forbes *et al.* [19] went in the other direction and showed that certain restricted algebraic circuit lower bounds imply size lower bounds on subsystems of IPS. In particular, [19] devised a simple framework by which lower bounds on (subsystems of) IPS refutations are reduced to algebraic circuit lower bounds. [19] used this framework to establish lower bounds on subsystems of IPS refutations of variants of symmetric subset-sum instances when the IPS refutations are written as read once algebraic branching programs and multilinear formulas. But lower bounds on the size of unrestricted IPS refutations were not known.

2 Preliminaries

2.1 Notation

For a natural number we let $[n] = \{1, \ldots, n\}$. Let R be a ring. Denote by $R[x_1, \ldots, x_n]$ the ring of multivariate polynomials with coefficients from R and variables $x_1, \ldots, x_n$. We usually denote by $\overline{x}$ the vector of variables $x_1, \ldots, x_n$. We treat polynomials as *formal* linear combination of monomials, where a monomial is a product of variables. Hence, when we talk about the *zero polynomial* we mean the polynomial in which the coefficients of all monomials are zero. Similarly, two polynomials are said to be *identical* if their monomials have the same coefficients. The *number of monomials* in a polynomial f is the number of monomials with nonzero coefficients denoted $|f|_{\#\text{monomials}}$. The *degree* of a multivariate polynomial (or total degree) is the maximal sum of variable powers in a monomial with a nonzero coefficient in the polynomial. We write $\text{poly}(n)$ to denote a polynomial growth in n, namely a function that is upper bounded by cn^c, for some constant c independent of n. Similarly, $\text{poly}(n_1, \ldots, n_s)$ for some constant s, means a polynomial growth that is at most $kn_1^{c_1} \cdots n_s^{c_s}$, for k and c_{ji}'s that are constants independent of $n_1, \ldots, n_s$.

For S a set of polynomials from $R[x_1, \ldots, x_n]$, we denote by $\langle S \rangle$ the *ideal generated by* S, namely the minimal set containing S such that if $f, g \in \langle S \rangle$ then also $\alpha f + \beta g \in \langle S \rangle$, for any $\alpha, \beta \in R$.

2.2 Algebraic Circuits

Algebraic circuits over some fixed chosen field or ring R compute polynomials in $R[x_1, \ldots, x_n]$ via addition and multiplication gates, starting from the input variables $\overline{x}$ and constants from the field. More precisely, an *algebraic circuit* C is a finite directed acyclic graph where edges are directed from leaves (that is, in-degree 0 nodes) towards the output nodes (that is out-degree 0 nodes). By default, there is a single output node. *Input nodes* are in-degree 0 nodes that are labeled with a variable from $x_1, \ldots, x_n$; every other in-degree zero node is labelled with a scalar element in R. All the other nodes have in-degree two (unless otherwise stated) and are labeled with either $+$ or $\times$. An in-degree 0 node is said to *compute* the variable or scalar that labels itself. A $+$ (or $\times$) gate is said to compute the addition (product, resp.) of the polynomials computed by its incoming nodes. The ***size*** of an algebraic circuit C is the number of nodes in it denoted $|C|$, and the *depth* of a circuit is the length of the longest directed path in it. Note that the size of a field coefficient in this setting is 1 irrespective of the value of the coefficient (this is called sometimes the "unit-cost" model). Sometimes it is important to consider the size of the coefficients appearing in the circuit (for instance, when we are concerned with the computational complexity of problems pertaining to algebraic circuits we need to have an efficient way to represent the circuits as bit strings). For this purpose it is standard to define a ***constant-free*** algebraic circuit to be an algebraic circuit in which the only constants used are $0, 1, -1$. Other constants must be built up using algebraic operations, which then count towards the size of the circuit. Constant-free algebraic circuit computes a polynomial over $\mathbb{Z}$, but when we allow for constant sub-circuits (and *only* for constant sub-circuits) to contain division gates (in Sect. 4) we can also compute polynomials over $\mathbb{Q}$ with constant-free circuits.

An algebraic circuit is said to be a *multi-output* circuit if it has more than one output node, namely, more than one node of out-degree zero. Given a single-output algebraic circuit $F(\overline{x})$ we denote by $\widehat{F}(\overline{x}) \in R[\overline{x}]$ the *polynomial* computed by $F(\overline{x})$, to distinguish at some points the circuit from the polynomial it computes. We define the *degree* of a circuit C (similarly a node) as the total degree of the polynomial $\hat{C}$ computed by C, denoted $\deg(C)$.

Algebraic Complexity Classes. We now recall some basic notions from algebraic complexity (for more details see [54, Sec. 1.2]). Over a ring R, VP_R (for "Valiant's P") is the class of families $f = (f_n)_{n=1}^{\infty}$ of formal polynomials f_n such that f_n has $\mathrm{poly}(n)$ input variables, is of $\mathrm{poly}(n)$ degree, and can be computed by algebraic circuits over R of $\mathrm{poly}(n)$ size. VNP_R (for "Valiant's NP") is the class of families g of polynomials g_n such that g_n has $\mathrm{poly}(n)$ input variables and is of $\mathrm{poly}(n)$ degree, and can be written as

$$g_n\left(x_1, \ldots, x_{\mathrm{poly}(n)}\right) = \sum_{\overline{e} \in \{0,1\}^{\mathrm{poly}(n)}} f_n(\overline{e}, \overline{x})$$

for some family $(f_n) \in \mathsf{VP}_R$. A major question in algebraic complexity theory is whether the permanent polynomial can be computed by algebraic circuits of polynomial size. Since the permanent is complete for VNP (under a suitable concept of algebraic reductions that are called p-projections), showing that no polynomial-size circuit can compute the permanent amounts to showing $\mathsf{VP} \neq \mathsf{VNP}$ (cf. [59, 60, 62]).

Similarly, we can consider the *constant-free* versions of VP and VNP: we denote by VP^0 and VNP^0 the class of polynomial-size and polynomial-degree *constant-free* algebraic circuits and the class of VNP polynomials as above in which the family of polynomials $(f_n) \in \mathsf{VP}^0$. In these definitions of VP^0

and VNP^0 we assume also that no division gate occur in the circuits, hence VP^0 and VNP^0 compute polynomials over $\mathbb{Z}$. We shall also consider in Sect. 4 constant-free circuits *over* $\mathbb{Q}$: these will be constant-free circuits in which constant sub-circuits (and *only* constant sub-circuits) may contain division gates.

2.3 The τ-Conjecture and Shub–Smale Hypothesis

Here we explain several important assumptions and conjectures that are known to lead to strong complexity lower bounds and complexity class separations, all of which are relevant to our work. See for example Smale's list of "mathematical problems for the next century" [56] for a short description and discussion about these problems. Recall Definition 2.1 of the τ-function.

When we focus on constant polynomials, that is, numbers $n \in \mathbb{Z}$, $\tau(n)$ is the minimal-size circuit that can construct n from 1 using additions, subtractions and multiplications (but not divisions; note that subtraction of a term A can be constructed by $-1 \cdot A$).

Definition 2.1 (τ-function [55])**.** *Let $f \in \mathbb{Z}[\overline{x}]$ be a multivariate polynomial over $\mathbb{Z}$. Then $\tau(f)$ is the minimal size of a constant-free algebraic circuit that computes f (that is, a circuit where the only possible constants that may appear on leaves are $1, 0, -1$).*

We say that a family of (possibly constant) polynomials $(f_n)_{n\in\mathbb{N}}$ is ***easy*** if there is a constant $c \in \mathbb{N}$ such that $\tau(f_n) \leq \log^c n$, for every $n > 2$, and ***hard*** otherwise.[1]

The following are some known facts about $\tau(\cdot)$:

- $(2^n)_{n\in\mathbb{N}}$ is *easy*. For instance, if n is a power of 2 then $\tau(2^n) = \log n + 3$, where log denotes the logarithm in the base 2. We start with 3 nodes to build $2 = 1 + 1$ and then by $\log n$ repeated squaring we arrive at $((2^2)^2)^2 \dots)^2 = 2^{2^{\log n}} = 2^n$.
- $(2^{2^n})_{n\in\mathbb{N}}$ is *hard*. This is clear from the straightforward upper bound on the largest integer that can be computed with k multiplication/addition/subtraction gates.
- A simple known upper bound on τ is this [17]: for every integer $m > 2$, $\tau(m) \leq 2 \log m$. This is shown by considering the binary expansion of m.
- For every integer m, the following lower bound is known $\tau(m) \geq \log \log m$ [17].

While $(2^n)_{n\in\mathbb{N}}$ is easy and $(2^{2^n})_{n\in\mathbb{N}}$ is hard, it is not known whether $(n!)_{n\in\mathbb{N}}$ is easy or hard, and as seen below, showing the hardness of $\tau(m_n \cdot n!)$, for every sequence $(m_n \cdot n!)_{n\in\mathbb{N}}$ with $m_n \in \mathbb{Z}$ any nonzero integers, has very strong consequences.

Blum, Shub and Smale [8] introduced an algebraic version of Turing machines that has access to a field K (Poizat observed that their model can be defined as algebraic circuits in which *selection* gates $s(z, x, y)$ can be used; where a selection gate outputs x in case $z = 0$ and y in case $z = 1$). In this model one can formalise and study a variant of the P vs. NP problem for languages solvable by polynomial-time machines with access to K, denoted P_K, versus nondeterministic polynomial-time machines with access to K, denoted NP_K.

The following is a condition put forth by Shub and Smale [55] (cf. [56]) towards separating $\mathsf{P}_{\mathbb{C}}$ from $\mathsf{NP}_{\mathbb{C}}$, for $\mathbb{C}$ the complex numbers:

Shub–Smale Hypothesis ([55, 56])**.** *For every nonzero integer sequence $(m_n)_{n\in\mathbb{N}}$, the sequence $(m_n \cdot n!)_{n\in\mathbb{N}}$ is hard.*

[1] We put the condition $n > 2$ instead of $n \geq 1$, because unlike [55] we do not add the constant 2 to the constants available in the circuit. Therefore, to keep the same known upper bounds of τ we skip the cases $n = 1, 2$.

Shub and Smale, as well as Bürgisser, showed the following consequences of the Shub–Smale hypothesis:

Theorem 2.2 ([55, 9]). *1. If the Shub–Smale hypothesis holds then* $\mathsf{P}_{\mathbb{C}} \neq \mathsf{NP}_{\mathbb{C}}$.

2. If the Shub–Smale Hypothesis holds then $\mathsf{VP}^0 \neq \mathsf{VNP}^0$. *In other words, Shub–Smale Hypothesis implies that the permanent does not have polynomial size constant-free algebraic circuits over* $\mathbb{Z}$.

It is open whether the Shub–Smale hypothesis holds. What is known is that if Shub–Smale hypothesis does *not* hold then factoring of integers can be done in (nonuniform) polynomial time (cf. Blum *et al* [7, p.126] and [12]).

Another related important assumption in algebraic complexity is the τ-*conjecture*. Let $f \in \mathbb{Z}[x]$ be a univariate polynomial with integer coefficients, denote by $z(f)$ the number of distinct integer roots of f.

τ-Conjecture ([55, 56]). *There is a universal constant c, such that for every univariate polynomial $f \in \mathbb{Z}[x]$:*

$$(1 + \tau(f))^c \geq z(f)\,.$$

The consequences of the τ-conjecture are similar to the Shub–Smale Hypothesis:

Theorem 2.3 ([55, 9]). *If the τ-conjecture holds then both* $\mathsf{P}_{\mathbb{C}} \neq \mathsf{NP}_{\mathbb{C}}$ *and* $\mathsf{VP}^0 \neq \mathsf{VNP}^0$ *hold.*

2.4 Basic Proof Complexity

In the standard setting of propositional proof complexity, a *propositional proof system* [16] is a polynomial-time predicate $V(\pi, x)$ that verifies purported proofs π (encoded naturally in, say, binary) for propositional formulas x (also encoded naturally in binary), such that $\exists \pi\ (V(\pi, x) = \mathsf{true})$ iff x is a tautology.[2] Hence, a propositional proof system is a complete and sound proof system for propositional logic in which a proof can be checked for correctness in polynomial time (though, note that a proof π may be exponentially larger than the tautology x it proves).

When considering algebraic proof systems that operate with algebraic circuits, such as IPS, it is common to relax the notion of a propositional proof system, so to require that the relation $V(\pi, x)$ is in probabilistic polynomial time, instead of deterministic polynomial time (since polynomial identities can be verified in coRP, while not known to be verified in P).

Furthermore, the language that a given proof system proves, namely the set of instances that the proof system proves to be tautological, or always satisfied, can be different from the set of propositional tautologies. First, we can consider a propositional proof system to be a *refutation system* in which a proof establishes that the initial set of axioms (e.g., clauses) is *unsatisfiable*, instead of always satisfied (i.e., tautological). For most cases, considering a propositional proof system to be a refutation system preserves all properties of the proof system, and thus the notions of refutation and proofs are used as synonyms. Second, we can define a proof system to be complete and sound for languages different or larger than unsatisfiable propositional formulas. For instance, in algebraic proof systems we usually consider proof systems that are sound and complete for the language of unsatisfiable sets of polynomial equations.

[2] Historically, Cook and Reckhow [16] defined a propositional proof systems as a polynomial-time computable surjective mapping of bit strings (encoding purported proofs) *onto* the set of propositional tautologies (encoded as bit strings as well). This is equivalent to the definition of propositional proof systems we presented, up to polynomial factors.

For the purpose of comparing the relative complexity of different proof systems we have the concept of *simulation*: given two proof systems P, Q for the *same* language, we say that P ***simulates*** Q if there is a function f that maps Q-proofs to P-proofs of the same instances with at most a polynomial blow-up in size. If f can be computed in polynomial time, this is called a ***p-simulation***. If P and Q simulate each other we say that P and Q are *polynomially equivalent*. If P and Q are two proof systems for *different* languages, prima facie we cannot compare their strength via the notion of simulation. However, if both P and Q prove (or refute) propositional instances like formulas in conjunctive normal form, or Boolean tautologies, while encoding them in different ways (namely, they use different representations for essentially the same propositional formulas), we can fix a polynomial-time computable translation from one representation to the other. Under this translation we can consider P and Q to be proof systems for the *same* language, allowing us to use the notion of simulation between P and Q.

2.5 Algebraic Proofs

Grochow and Pitassi [28], following [45], suggested the following algebraic proof system which is essentially a Nullstellensatz proof system [4] written as an algebraic circuit (this was shown in [19]). A proof in the Ideal Proof System is given as a *single* polynomial. We provide below the *Boolean* version of IPS (which includes the Boolean axioms), namely the version that establishes the unsatisfiability over 0-1 of a set of polynomial equations. In what follows we follow the notation in [19]:

Definition 2.4 ((Boolean) Ideal Proof System (IPS), Grochow-Pitassi [28])**.** *Let $f_1(\overline{x}), \ldots, f_m(\overline{x}), p(\overline{x})$ be a collection of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$ over the field $\mathbb{F}$. An* ***IPS proof of*** *$p(\overline{x}) = 0$* ***from*** *$\{f_j(\overline{x}) = 0\}_{j=1}^m$, 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 $C(\overline{x}, \overline{y}, \overline{z}) \in \mathbb{F}[\overline{x}, y_1, \ldots, y_m, z_1, \ldots, z_n]$ such that (the equalities in what follows stand for formal polynomial identities[3]):*

1. *$C(\overline{x}, \overline{0}, \overline{0}) = 0$; and*
2. *$C(\overline{x}, f_1(\overline{x}), \ldots, f_m(\overline{x}), x_1^2 - x_1, \ldots, x_n^2 - x_n) = p(\overline{x})$.*

The ***size of the IPS proof*** *is the size of the circuit C. If C is assumed to be constant-free, we refer to the size of the proof as the* ***size of the constant-free IPS proof****. The variables $\overline{y}, \overline{z}$ are called the* placeholder variables *since they are used as placeholders for the axioms. An IPS proof $C(\overline{x}, \overline{y}, \overline{z})$ of $1 = 0$ from $\{f_j(\overline{x}) = 0\}_{j \in [m]}$ is called an* ***IPS refutation*** *of $\{f_j(\overline{x}) = 0\}_{j \in [m]}$ (note that in this case it must hold that $\{f_j(\overline{x}) = 0\}_{j=1}^m$ have no common solutions in $\{0,1\}^n$).*

Notice that the definition above adds the equations $\{x_i^2 - x_i = 0\}_{i=1}^n$, called the set of ***Boolean axioms*** denoted $\overline{x}^2 - \overline{x}$, to the system $\{f_j(\overline{x}) = 0\}_{j=1}^m$. This allows to refute over $\{0,1\}^n$ unsatisfiable systems of equations. Also, note that the first equality in the definition of IPS means that the polynomial computed by C is in the ideal generated by $\overline{y}, \overline{z}$, which in turn, following the second equality, means that C witnesses the fact that $p(\overline{x})$ is in the ideal generated by $f_1(\overline{x}), \ldots, f_m(\overline{x}), x_1^2 - x_1, \ldots, x_n^2 - x_n$. (the existence of this witness, for unsatisfiable set of polynomials (that is, $p(\overline{x}) = 1$), stems from Hilbert's Nullstellensatz [4]).

In order to use IPS as a propositional proof system (namely, a proof system for propositional tautologies), we need to fix the encoding of clauses as algebraic circuits.

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

Definition 2.5 (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 in this way a 0-1 assignment to a CNF is satisfying iff the assignment is satisfying all the equations in the algebraic translation of the CNF.

Therefore, using Definition 2.5 to encode CNF formulas, Boolean IPS is considered as a propositional proof system for the language of unsatisfiable CNF formulas, sometimes called ***propositional IPS***. We say that an IPS proof is an ***algebraic IPS*** proof, if we do not use the Boolean axioms $\overline{x}^2 - \overline{x}$ in the proof. *As a default when referring to IPS we mean the Boolean IPS version.*

2.5.1 Conventions and Notations for IPS Proofs

An IPS proof over a specific field or ring is sometimes denoted $\mathrm{IPS}_{\mathbb{F}}$ noting it is over $\mathbb{F}$. For two algebraic circuits F, G, we define the *size of the equation* $F = G$ to be the total circuit size of F and G, namely, $|F| + |G|$.

Let $\overline{\mathcal{F}}$ denote a set of polynomial equations $\{f_i(\overline{x}) = 0\}_{i=1}^m$, and let $C(\overline{x}, \overline{y}, \overline{z}) \in \mathbb{F}[\overline{x}, \overline{y}, \overline{z}]$ be an IPS proof of $f(\overline{x})$ from $\overline{\mathcal{F}}$ as in Definition 2.4. Then we write $C(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x})$ to denote the circuit C in which y_i is substituted by $f_i(\overline{x})$ and z_i is substituted by the Boolean axiom $x_i^2 - x_i$. By a slight abuse of notation we also call $C(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) = f(\overline{x})$ an IPS proof of $f(\overline{x})$ from $\overline{\mathcal{F}}$ and $\overline{x}^2 - \overline{x}$ (that is, displaying $C(\overline{x}, \overline{y}, \overline{z})$ *after* the substitution of the placeholder variables $\overline{y}, \overline{z}$ by the axioms in $\overline{\mathcal{F}}$ and $\overline{x}^2 - \overline{x}$, respectively).

For two polynomials $f(\overline{x}), g(\overline{x})$, *an IPS proof of* $f(\overline{x}) = g(\overline{x})$ from the assumptions $\overline{\mathcal{F}}$ is an IPS proof of $f(\overline{x}) - g(\overline{x}) = 0$ (note that in case $f(\overline{x})$ and $g(\overline{x})$ are identical as polynomials this is trivial to prove; see Fact A.1).

We denote by $C : \overline{\mathcal{F}} \vdash^s_{\mathrm{IPS}} p = 0$ (resp. $C : \overline{\mathcal{F}} \vdash^s_{\mathrm{IPS}} p = g$) the fact that $p = 0$ (resp. $p = g$) has an IPS proof $C(\overline{x}, \overline{y}, \overline{z})$ of size s from assumptions $\overline{\mathcal{F}}$. We may also suppress "$= 0$" and write simply $C : \overline{\mathcal{F}} \vdash^s_{\mathrm{IPS}} p$ for $C : \overline{\mathcal{F}} \vdash^s_{\mathrm{IPS}} p = 0$. Whenever we are only interested in claiming the existence of an IPS proof of size s of $p = 0$ from $\overline{\mathcal{F}}$ we suppress the C from the notation. Similarly, we can suppress the size parameter s from the notation. If F is a circuit computing a polynomial $\hat{F} \in \mathbb{F}[\overline{x}]$, then we can talk about *an IPS proof C of F from assumptions* $\overline{\mathcal{F}}$, in symbols $C : \overline{\mathcal{F}} \vdash_{\mathrm{IPS}} F$, meaning an IPS proof of $\hat{F}$. Accordingly, for two circuits F, F' such that $\hat{F} = \hat{F}'$, we may speak about an *an IPS proof C of F from assumptions* $\overline{\mathcal{F}}$ to refer to an IPS proof of F' from assumptions $\overline{\mathcal{F}}$.

2.6 Semi-Algebraic Proofs

The *Positivstellensatz* proof system, as defined by Grigoriev and Vorobojov [27], is a refutation system for establishing the unsatisfiability over the reals $\mathbb{R}$ of a system consisting of both polynomial equations $\overline{\mathcal{F}} = \{f_i(\overline{x}) = 0\}_{i\in I}$ and polynomial inequalities $\overline{\mathcal{H}} = \{h_j(\overline{x}) \geq 0\}_{j\in J}$, respectively. It is based on a restricted version of Krivine–Stengle's Positivstellensatz [35, 57]. In order to formulate it, we need to define the notion of a cone, as in [27], which serves as a non-negative closure of a set of polynomials, or informally the notion of a "positive ideal". Usually the cone is defined as the set closed under non-negative linear combinations of polynomials (cf. [6]), but following [27] we are going to use a more general formulation which is sometimes called *the SoS cone*.

Definition 2.6 (cone)**.** *Let $\overline{\mathcal{H}} \subseteq R[\overline{x}]$ be a set of polynomials over an ordered ring R. Then* the cone of $\overline{\mathcal{H}}$*, denoted* $\mathrm{cone}(\overline{\mathcal{H}})$*, is defined to be the smallest set $S \subseteq R[\overline{x}]$ such that:*

1. *$\overline{\mathcal{H}} \subseteq S$;*
2. *for any polynomial $s \in R[\overline{x}]$, $s^2 \in S$;*

3. *for any positive constant* $c > 0$, $c \in S$;

4. *if* $f, g \in S$, *then both* $f + g \in S$ *and* $f \cdot g \in S$.

Note that we have formulated the cone for any ordered ring (item item 3 would be superfluous for reals). This is because we are going to use this notion in the context of $\mathbb{Z}$ and $\mathbb{Q}$ (although Krivine–Stengle's Positivstellensatz does not hold for these rings, it is still possible to use Positivstellensatz refutations in the presence of the Boolean axioms, namely as a refutation system for instances unsatisfiable over 0-1 value).

Note also that every sum of squares (that is, every sum of squared polynomials $\sum_i s_i^2$) is contained in every cone. Specifically, cone($\emptyset$) contains every sum of squares.

Similar to the way the Nullstellensatz proof system [4] establishes the unsatisfiability of sets of polynomial equations based on the Hilbert's Nullstellensatz [29] from algebraic geometry, the Positivstellensatz proof system is based on Krivine–Stengle's Positivstellensatz from semi-algebraic geometry:

Theorem 2.7 (Positivstellensatz [35, 57], restricted version)**.** *Let* $\overline{\mathcal{F}} := \{f_i(\overline{x}) = 0\}_{i \in I}$ *be a set of polynomial equations and let* $\overline{\mathcal{H}} := \{h_j(\overline{x}) \geq 0\}_{j \in J}$ *be a set of polynomial inequalities, where all polynomials are from* $\mathbb{R}[x_1, \ldots, x_n]$. *There exists a pair of polynomials* $f \in \langle \{f_i(\overline{x})\}_{i \in I} \rangle$ *and* $h \in \text{cone}(\{h_j(\overline{x})\}_{j \in J})$ *such that* $f + h = -1$ *if and only if there is no assignment that satisfies both* $\overline{\mathcal{F}}$ *and* $\overline{\mathcal{H}}$.

The Positivstellensatz proof system is now natural to define. We shall distinguish between the *real* Positivstellensatz in which variables are meant to range over the reals and *Boolean* Positivstellensatz in which variables range over $\{0, 1\}$.

Definition 2.8 (real Positivstellensatz proof system (real PS) [27])**.** *Let* $\overline{\mathcal{F}} := \{f_i(\overline{x}) = 0\}_{i \in I}$ *be a set of polynomial equations and let* $\overline{\mathcal{H}} := \{h_j(\overline{x}) \geq 0\}_{j \in J}$ *be a set of polynomial inequalities, where all polynomials are from* $\mathbb{R}[x_1, \ldots, x_n]$. *Assume that* $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$ *have no common real solutions. A* Positivstellensatz refutation *of* $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$ *is a collection of polynomials* $\{p_i\}_{i \in I}$ *and* $\{s_{i,\zeta}\}_{i,\zeta}$ *(for* $i \in \mathbb{N}$, $\zeta \subseteq J$ *and* $I_\zeta \subseteq \mathbb{N}$*) in* $\mathbb{R}[x_1, \ldots, x_n]$ *such that the following formal polynomial identity holds:*

$$\sum_{i \in I} p_i \cdot f_i + \sum_{\zeta \subseteq J} \left(\prod_{j \in \zeta} h_j \cdot \left(\sum_{i \in I_\zeta} s_{i,\zeta}^2 \right) \right) = -1 \,. \tag{1}$$

The ***monomial size*** *of a Positivstellensatz refutation is the combined total number of monomials in* $\{p_i\}_{i \in I}$ *and* $\sum_{i \in I_\zeta} s_{i,\zeta}^2$, *for all* $\zeta \subseteq J$, *that is,* $\sum_{i \in I} |p_i|_{\#\text{monomials}} + \sum_{\zeta \subseteq J} \left| \sum_{i \in I_\zeta} s_{i,\zeta}^2 \right|_{\#\text{monomials}}$. [4]

Note that Grigoriev, Hirsch, and Pasechnik [26] defined the size of Positivstellensatz proofs slightly differently: they included in the size of proofs both the number of monomials and the size of the coefficients of monomials written in binary (while this does not matter for their lower bounds). This is more natural when considering Positivstellensatz as a propositional proof system (which is polynomially verifiable).

In order to use Positivstellensatz as a refutation system for collections of equations $\overline{\mathcal{F}}$ and inequalities $\overline{\mathcal{H}}$ that are unsatisfiable over 0-1 assignments, we need to include simple Boolean axioms. This is done in slightly different ways in different works (see for example [26, 2]). One way to do this, which is the way we follow, is the following:

[4] The definition of size measure for Positivstellensatz and SoS proofs is slightly less standard than degree measure (see discussion in [2]). We define the monomial size measure of Positivstellensatz proofs to count the monomials in p_i and $s_{i,\zeta}^2$, while ignoring the monomials in the initial axioms in $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$. This choice of definition is closer to the definition of size of IPS proofs, which ignores the size of the initial axioms.

Definition 2.9 ((Boolean) Positivstellensatz proof system (Boolean PS)). *A **Boolean Positivstellensatz proof** from a set of polynomial equations $\overline{\mathcal{F}}$, and polynomial inequalities $\overline{\mathcal{H}}$, is an algebraic Positivstellensatz proof in which the following **Boolean axioms** are part of the axioms: the polynomial equations $x_i^2 - x_i = 0$ (for all $i \in [n]$) are included in $\overline{\mathcal{F}}$, and the polynomial inequalities $x_i \geq 0,\ 1 - x_i \geq 0$ (for all $i \in [n]$) are included in $\overline{\mathcal{H}}$.*

In this way, $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$ have no common 0-1 solutions iff there exists a Boolean Positivstellensatz refutation of $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$. Eventually, to define the Boolean Positivstellensatz as a propositional proof system for the unsatisfiable CNF formula we consider CNF formulas to be encoded as polynomial equalities according to Definition 2.5. This version is sometimes called ***propositional Positivstellensatz***. *As a default when referring to Positivstellensatz we mean the Boolean Positivstellensatz version.*

In recent years, starting mainly with the work of Barak, Brandao, Harrow, Kelner, Steurer and Zhou [3], a special case of the Positivstellensatz proof system has gained much interest due to its application in complexity and algorithms (cf. [42]). This is the ***sum-of-squares*** proof system (**SoS**), which is defined as follows:

Definition 2.10 (sum-of-squares proof system). *A **sum-of-squares proof** (SoS for short) is a Positivstellensatz proof in which in eq. 1 in Definition 2.8 we restrict the index sets $\zeta \subseteq J$ to be* singletons*, namely $|\zeta| = 1$, hence, disallowing arbitrary products of inequalities within themselves. The real, Boolean and propositional versions of* SoS *are defined similar to Positivstellensatz.*

For most interesting cases SoS is also complete (and sound) by a result of Putinar [49].

2.6.1 Dynamic Positivstellensatz

Here we follow Grigoriev, Hirsch and Pasechnik [26] to define what is, to the best of our knowledge, the most general propositional Positivstellensatz- (or SoS-) based semi-algebraic proof system defined to date. It can be viewed as the generalization of (dynamic) Lovász–Schrijver proof systems [39, 38] that have been put in the context of propositional proof complexity by Pudlák [48], and constitutes essentially a dynamic version of propositional Positivstellensatz (the proof size is measured by the total number of monomials appearing in the proof).

The translation of propositional formulas here is different from the algebraic translation (Definition 2.5). For higher degree proof systems, Definition 2.5 and the definition that follows can be reduced to one another (within the proof system, as long as both translations can be written down efficiently); however, we provide Definition 2.11 for the sake of consistency with earlier work.

Definition 2.11 (semi-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 $\sum_{i\in P} x_i + \sum_{j\in N}(1 - x_j) \geq 1$.*

Notice that in this way a 0-1 assignment to a CNF formula is satisfying iff the assignment satisfies all the inequalities in the semi-algebraic translation of the CNF formula.

Definition 2.12 ($\mathrm{LS}^{\infty}_{*,+}$ [26]). *Consider a Boolean formula in conjunctive normal form and translate it into inequalities as in Definition 2.11. Take these inequalities as axioms, add the axioms $x \geq 0$, $1 - x \geq 0$, $x^2 - x \geq 0$, $x - x^2 \geq 0$ for each variable x. Allow also $h^2 \geq 0$ as an axiom, for any polynomial h of degree at most d. An $\mathrm{LS}^{d}_{*,+}$ proof of the original formula is a derivation of $-1 \geq 0$ from these axioms using the following rules:*

$$\frac{f \geq 0, \quad g \geq 0}{f + g \geq 0} \qquad \frac{f \geq 0}{\alpha f \geq 0} \text{ (for } \alpha \text{ a nonnegative integer)} \qquad \frac{f \geq 0, \quad g \geq 0}{f \cdot g \geq 0}.$$

In particular, $\mathrm{LS}^{\infty}_{*,+}$ *is such a proof without the restriction on the degree. Note that we have to write polynomials as sums of monomials (and not as circuits or formulas), so the verification of such proof is doable in deterministic polynomial-time (assuming field operations and field coefficients are deterministic polynomial-time computable).*

The proof of the following simulation follows by definition and we omit the details:

Proposition 2.13. $\mathrm{LS}^{\infty}_{*,+}$ *simulates Boolean Positivstellensatz.*

3 Overview of Results and Organisation

We consider the following subset-sum instance written as an unsatisfiable linear equation with large coefficients, expressing the fact that natural numbers written in binary cannot be negative:

Definition 3.1 (Binary Value Principle BVP_n)**.** *The* binary value principle *over the variables* $x_1, \ldots, x_n$*,* BVP_n *for short, is the following unsatisfiable (over* $\{0,1\}$ *assignments) linear equation:*

$$x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n = -1\,.$$

At times we use a more general principle denoted $\mathrm{BVP}_{n,M}$, which we call the *generalized binary value principle*: $x_1 + 2x_2 + 4x_3 + \cdots + 2^{n-1}x_n = -M$, for a positive integer M.

Note that, though simple, the binary value principle is not a direct translation of a Boolean formula, hence, similarly to [19] and other results on algebraic proofs (e.g., Razborov [51]), IPS lower bounds on this principle do not necessarily entail lower bounds for the usual (Boolean) Frege systems.

3.1 Lower Bounds

We prove two kinds of conditional super-polynomial lower bounds against IPS refutations. The first is over $\mathbb{Q}$ and $\mathbb{Z}$ and the second is over the field $\mathbb{Q}(y)$ of rational functions of univariate polynomials in the indeterminate y denoted $\mathbb{Q}[y]$ (see Definition 4.8). They are conditioned on two different conjectures from the same paper by Shub and Smale [55]. We start with the first lower bound.

Theorem (Thm. 4.5)**.** *Under the Shub and Smale hypothesis, there are no* $\mathrm{poly}(n)$*-size constant-free (Boolean) IPS refutations of the binary value principle* BVP_n *over* $\mathbb{Q}$*.*

This result can be viewed as generalizing to full IPS the proof method initiated by Forbes et *al.* [19] wherein proof complexity lower bound questions are reduced to algebraic circuit size lower bound questions: an IPS proof written as a circuit from a class $\mathcal{C}$ is obtained by showing that there are no small $\mathcal{C}$-circuits computing certain polynomials. Here, by "full IPS" we simply mean that instead of using circuit lower bounds to obtain lower bounds against *sub-systems* of IPS, we use a circuit lower bound, alas conditional, to obtain a lower bound against (general) IPS.

We stress that *this approach can only lead to conditional lower bounds for full (unrestricted)* IPS, as long as we do not have (explicit) super-polynomial lower bounds against general algebraic circuits, namely as long as we do not prove that VP^0 captures an extension of VNP^0 by divisions (see Sect. 4.2.4 below).[5]

[5] Though, it should be mentioned that in proof complexity even non-explicit lower bounds are not known, and will constitute a breakthrough in the field; hence moving from non-explicit (and thus *known*) circuit lower bounds to (possibly also non-explicit) proof complexity lower bounds cannot be ruled out entirely.

Rational field lower bounds. We consider IPS operating over the field of rational functions in the (new) indeterminate y, denoted $\mathbb{Q}(y)$ (Definition 4.8). This allows us to formulate a very interesting version of the binary value principle. Roughly speaking, this version expresses the fact that the BVP is "almost always" unsatisfiable. More precisely, consider the linear equation $\sum_{i=1}^{n} a_i x_i = y$, for integer a_i's, and y the new indeterminate. This equation is unsatisfiable for most y's, when y is substituted by an element from $\mathbb{Q}$. We show that once we have an IPS refutation over $\mathbb{Q}(y)$ of this equation we can substitute y by any rational number but a finite number of rational numbers and get a valid IPS refutation over $\mathbb{Q}$ of the original BVP. Thus *an IPS refutation over* $\mathbb{Q}(y)$ *of* $\sum_{i=1}^{n} a_i x_i = y$ *can be viewed as a single refutation for all but finitely many values of* $y \in \mathbb{Q}$.

We show that while for polynomially bounded coefficients a_i there are small $\mathbb{Q}(y)$-IPS refutations of $\sum_{i=1}^{n} a_i x_i = y$, for $\sum_{i=1}^{n} 2^{i-1} x_i = y$, there are no small refutations, assuming the τ-conjecture:

Theorem (Thm. 4.13)**.** *Suppose a system of polynomial equations* $F_0(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$, $F_i \in \mathbb{Q}(y)[x_1, \ldots, x_n]$, *where* $F_0(\overline{x}) = y + \sum_{i=1}^{i=n} 2^{i-1} x_i$ *and* $F_i(\overline{x}) = x_i^2 - x_i$, *has an IPS-LIN*$_{\mathbb{Q}(y)}$ *certificate* $H_0(\overline{x}), \ldots, H_n(\overline{x})$, *where each* $H_i(\overline{x})$ *can be computed by a* $\mathbb{Q}(y)[x_1, \ldots, x_n]$*-algebraic circuit of size* poly(n) *. Then, the* τ*-conjecture is false.*

Roughly speaking, the lower bound proof extracts denominators from the refutation and obtains a small circuit that has all n-bit nonnegative integers as its roots and thus cannot exist under the τ-conjecture.

We also raise an interesting question (Sect. 4.2.4) about the possibility that a lower bound on IPS refutations of BVP$_n$ implies a separation of $\mathsf{VNP}^0_{\mathbb{Q}}$ from $\mathsf{VP}^0_{\mathbb{Q}}$ (generalising the result of [28] showing that CNF formulas lower bounds in IPS implies $\mathsf{VNP}^0 \neq \mathsf{VP}^0$).

3.2 Algebraic versus Semi-Algebraic Proofs

In Sect. 5 we exhibit the importance of the binary value principle by showing that it captures in a manner made precise the strength of semi-algebraic reasoning in the regime of strong (to very strong) proof systems, and formally those systems that can efficiently reason about bit arithmetic. Note that already Frege system can reason about bit arithmetic (see [20] following [10]); however, this alone is not sufficient to simulate semi-algebraic systems: one needs also to be able to prove the BVP. Specifically, we show that short refutations of the binary value principle would bridge the gap between very strong algebraic reasoning captured by the ideal proof system and its semi-algebraic analogue that we introduce in this work, which we call the Cone Proof System (CPS for short; Definition 5.2).

In contrast to IPS where a short refutation for BVP$_n$ would imply strong computational consequences, the binary value principle is trivially refutable in CPS (as well as in SoS):

We show that IPS simulates CPS if there exist small IPS refutations of the binary value principle. This provides a characterisation of semi-algebraic reasoning in terms of the binary value principle.

The relative strength of proof systems. Figure 1 provides an illustrative picture of the relative strength of algebraic and semi-algebraic proof systems, which gives context to our results. Note that CPS is among the strongest concrete proof systems for Boolean tautologies to be formalized to date: it simulates IPS (Thm. 5.13) which is already very strong (note that *constant-free* IPS simulates Extended Frege [28]). Like IPS it can prove freely polynomial identities, and so it "subsumes" in this sense such identities (accordingly, CPS proofs needs the full power of coRP to be verified). It is unclear whether even ZFC (namely, Zermelo-Fraenkel set theory with the axiom of choice) can simulate CPS as a proof system for sets of polynomial equations over a field (it is not hard to show

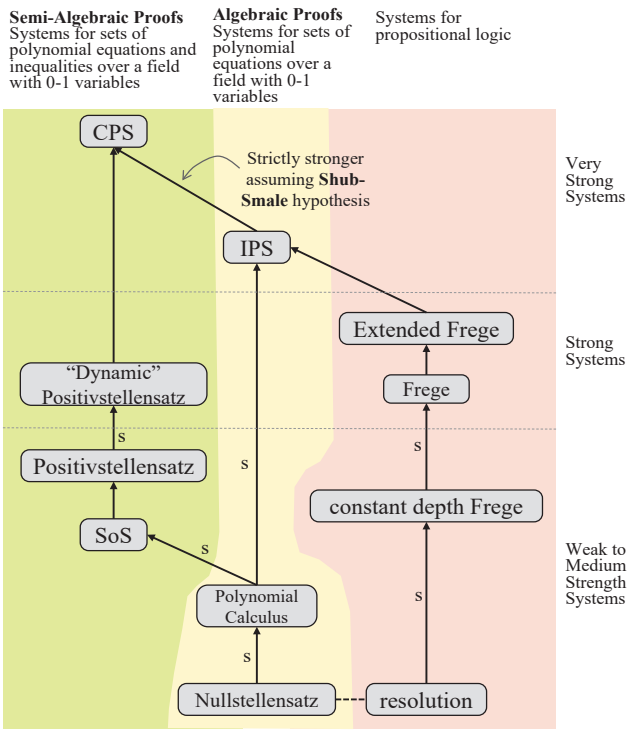

Figure 1: Relative strength of propositional proof systems (partial). An arrow $Q \to P$ means that P simulates Q. While $Q \overset{s}{\to} P$ means "strictly stronger", i.e., P simulates Q but Q does not simulate P. Dashed line $Q - - - P$ means that Q and P are incomparable: P cannot simulate Q and Q cannot simulate P. The three colored-shaded vertical blocks indicate proof systems for languages of increasing expressiveness (from right to left): systems for propositional logic, for polynomial equations with 0/1 variables (including encodings of propositional logic) and both polynomial equations and inequalities with 0/1 variables. The *informal* qualifications of strength mean roughly the following: weak systems are those we know super-polynomial lower bounds against, and their strength and limitations are quite well understood via feasible interpolation results and random CNFs lower bounds. Medium strength systems are those with some known lower bounds, but their strength is less well understood; e.g., feasible interpolation is not known for them. Strong systems are those with no known lower bounds. Very strong proof systems are those strong systems whose verification is done in coRP, and they can prove freely any polynomial identity (written as an algebraic circuit).

that this would imply that polynomial identity testing is in NP)[6]. Indeed, we are unaware of any concrete propositional proof system (even those that are merely coRP-verifiable) that can simulate CPS.

Grigoriev [24] showed that algebraic proofs like PC cannot simulate semi-algebraic proofs like SoS because symmetric subset-sum instances such as $x_1 + \cdots + x_n = -1$ require linear degrees (and exponential monomial size) [33], and Forbes *et al* [19] extended these lower bounds on symmetric subset-sum instances to stronger algebraic proof systems, namely to subsystems of IPS. Our work (Thm. 4.5) extends this gap further, showing that even the strongest algebraic proof system known to date IPS cannot fully simulate even a weak proof system like SoS, assuming Shub–Smale hypothesis.

Exponential size lower bounds for semi-algebraic proof systems are known since [26], and such bounds for propositional versions of static Lovász–Schrijver and constant degree Positivstellensatz systems were proved in [34]. Beame, Pitassi and Segerlind [44] started the study of lower bounds for semantic threshold systems, that include in particular tree-like Lovász-Schrijver systems. This line of research culminated in [22], where strong lower bounds were proved using critical block sensitivity, a notion introduced in [31].

[6]Indeed, if ZFC simulates CPS, and hence IPS, as a proof system for $\overline{F} \models p = 0$, where $\overline{F}$ is a set of polynomial equations over $\mathbb{Q}$ written as algebraic circuits and $\overline{F} \models p = 0$ means that the $\mathbb{Q}$ polynomial p written as an algebraic circuit is in the ideal generated by $\overline{F}$ (equivalently, that every $\mathbb{Q}$-assignment that nullifies $\overline{F}$ nullifies also p), then PIT is in NP. The proof proceeds as follows: let C be an algebraic circuit over $\mathbb{Q}$. Then there exists a size $|C|$ IPS proof of $\models C = 0$ iff C is identically zero. Hence, there exists a size poly($|C|$) ZFC proof of $\models C = 0$ iff C is identically zero. Since ZFC is verifiable in poly-time (we assume here some efficient encoding of circuits over $\mathbb{Q}$), we conclude that any algebraic circuit C over $\mathbb{Q}$ is identically zero iff it has a poly-time verifiable witness (where the witness is the ZFC proof of $\models C = 0$).

3.3 Relation to other Work

Bit arithmetic and semi-algebraic proofs. In Sect. 6 we show how to reason about the bits of polynomial expressions within algebraic proofs. Bit arithmetic in proof complexity was used before in Frege systems (see [20] following [10]). Independently of our work presented initially at [30], Impagliazzo, Mouli, and Pitassi [32] considered the possibility to *effectively* simulate *weak* semi-algebraic proofs using medium-strength algebraic proofs. They have considered expressing and reasoning with the bits of algebraic expressions, as we do in Sect. 6. However, their proof methods and results are fundamentally different from ours: first, they work in the weak proof systems regime, while we work in the strong systems regime. I.e., they aim to effectively simulate *weak* proof systems like constant degree sum-of-squares (in which polynomials are written as sum of monomials), while we aim to simulate *very strong* proof systems such as CPS (essentially, Positivstellensatz written as algebraic circuits). Second, they use a different way to express bits in their work. This is done in order to be able to reason about bits with bounded-depth algebraic circuits, while we do not need this mechanism. Third, they show only *effective* simulation and not simulation (namely, before the algebraic proofs can simulate a system of polynomial equations or inequalities, the equations and inequalities need to be pre-processed, that is, translated "off-line" to their bit-vector representation). Fourth, they do not consider the VAL function nor the binary value principle, while our work shows that essentially this is a necessary ingredient in a full simulation of strong semi-algebraic proof systems. In fact, we have the following:

Assuming the Shub–Smale hypothesis, our results *rule out the possibility that even a very strong algebraic proof system such as IPS simulates (in contrast to the weaker notion of an effective simulation) even a weak semi-algebraic proof system like constant degree* SoS *measured by monomial size.* In other words, assuming Shub–Smale hypothesis, we rule-out the possibility that the arguments in [32] (or any other argument) can yield a simulation of constant degree SoS by algebraic proofs operating with constant depth algebraic circuits (depth-d PC in [32][7]). It remains however open whether depth-d PC simulates constant degree SoS *for the language of unsatisfiable CNF formulas* or for unsatisfiable sets of linear equations with small coefficients.

Another relevant very recent work that considers bit arithmetic in proof complexity from the applied perspective of verification and SAT-solving is Liew *et al* [37]. That paper shows the advantage of cutting planes (a proof system that operates with integral inequalities) over algebraic proof systems (in the weak regime of Polynomial Calculus).

Subset-sum lower bounds in proofs complexity. Different instances of the subset sum problem have been considered as hard instances for algebraic proof systems. For example, Impagliazzo, Pudlák, and Sgall [33] provided an exponential lower bound on the size of refutations of the symmetric subset sum instance $x_1 + \cdots + x_n = n + 1$, for Boolean x_i's, in Polynomial Calculus. Grigoriev [24] proved that the version $\sum_{i=1}^{n} x_i = r$ for a non-integer $r \approx \frac{n}{2}$ requires linear degrees to refute in Positivstellensatz, and [26] later transformed this idea into an exponential-size lower bound for both Positivstellensatz and static high-degree Lovász–Schrijver proof systems. Moreover, as already mentioned, our lower bounds can be seen as an extension to the case of general IPS refutations of the approach introduced by Forbes *et al* [19].

The work of Part and Tzameret [43] established unconditional exponential lower bounds on the size of resolution over linear equations refutations of the binary value principle, over any sufficiently large field $\mathbb{F}$, denoted $\mathrm{Res}(\mathrm{lin}_{\mathbb{F}})$. The proof techniques in [43] are completely different from the current work, but these results demonstrate that using instances with large coefficients in proof complexity provides new insight into the complexity of proof systems.

[7] Here we use the fact that IPS simulates depth-d PC.

3.4 Subsequent Work

After the publication of the current work, Alekseev [1] showed unconditionally that the binary value principle does not have small refutations in a strong algebraic proof system (PC with extension variables) when refutation size is measured by bit-size (in contrast to algebraic circuit size; namely, the coefficients appearing in the refutation must be of super-exponential magnitude). Furthermore, Govindasamy, Hakoniemi and Tzameret [23] showed unconditionally that there are no polynomial-size constant-depth IPS refutations of a simple variant of the subset-sum principle (when the IPS refutations are multilinear).

4 Conditional IPS Lower Bounds

4.1 IPS Lower Bounds under Shub–Smale Hypothesis

Here we provide a super-polynomial conditional lower bound on the size of (Boolean) IPS refutations of the binary value principle over the rationals based on the Shub–Smale Hypothesis (Sect. 2.3).

The conditional lower bound is first established for constant-free IPS proofs over $\mathbb{Z}$ and then we extract a lower bound over $\mathbb{Q}$ as a corollary using Cor. 4.4 below. Notice that we can consider IPS proofs also over rings, and not only fields, however it might make the proof system incomplete. In the case of (Boolean) IPS over $\mathbb{Z}$ in order to keep the completeness it suffices to assume that *refutations* are proofs of any nonzero constant polynomial rather than of 1 (cf. [11, Definition 2.1]):

Definition 4.1 ($\mathrm{IPS}_{\mathbb{Z}}$)**.** *An* $\mathrm{IPS}_{\mathbb{Z}}$ proof of $g(\overline{x}) \in \mathbb{Z}[\overline{x}]$ from a set of assumptions $\overline{\mathcal{F}} \subseteq \mathbb{Z}[\overline{x}]$ *is an IPS proof of* $g(\overline{x})$ *from* $\overline{\mathcal{F}}$*, as in Definition 2.4, where* $\mathbb{F} = \mathbb{Z}$ *and all the constants in the IPS proof are from* $\mathbb{Z}$*. An* $\mathrm{IPS}_{\mathbb{Z}}$ refutation of $\overline{\mathcal{F}}$ *is a proof of* M*, for* $M \in \mathbb{Z} \setminus \{0\}$*. (The definition is similar for the Boolean and algebraic IPS versions.)*

It is easy to see that this system is complete (by multiplying a refutation over $\mathbb{Q}$ by the greatest common denominator; we show a more efficient way below in Prop. 4.3).

We will need to define a constant-free circuit over $\mathbb{Q}$ (to define rational numbers we use division by constants; not to be confused with circuits with division by polynomials that compute rational functions).

Definition 4.2 (circuits and proofs over $\mathbb{Q}$)**.** *A* constant-free circuit over $\mathbb{Q}$ *is a constant-free algebraic circuit as in Sect. 2.2 that has an additional division gate* $\div$*, where* $u \div v$ *means that the polynomial computed by* u *is divided by the polynomial computed by* v*. Moreover, we require that for every division gate* $u \div v$ *the sub-circuit* v *contains* no variables *and computes a nonzero constant. A* constant-free IPS proof over $\mathbb{Q}$ *is an* IPS *proof written with a constant-free circuit over* $\mathbb{Q}$*.*

Note that circuits over $\mathbb{Q}$ may contain nested division gates. For example, $\frac{(\frac{x_1}{(1+1)\cdot(1+1)}\cdot x_2)}{(1+1)\cdot(\frac{1}{1+1+1})}$ which formally is $(x_1 \div ((1+1)\cdot(1+1))\cdot x_2) \div (1+1)\cdot(1+(1\div(1+1+1+1)))$. The following proposition is proved by a simple induction on the circuit size, using sufficiently many products to cancel out the denominators in the circuit over $\mathbb{Q}$, turning it into a circuit over $\mathbb{Z}$.

The following proposition is proved roughly along the lines of Valiant [61] (and is also reminiscent of Strassen's [58] argument to turn a circuit with division gates to a circuit division with only a single division gate).

Proposition 4.3 (from $\mathbb{Q}$-circuit to $\mathbb{Z}$-circuit)**.** *Let* C *be a size-*s *constant-free circuit over* $\mathbb{Q}$ *computing a polynomial* $q \in \mathbb{Q}[\overline{x}]$*. Then there exists a size* $\leq 4s$ *constant-free circuit (without division gates) over* $\mathbb{Z}$ *computing* $M \cdot q$*, for some* $M \in \mathbb{Z} \setminus \{0\}$*, with* $\tau(M) \leq 4s$*.*

Proof: We choose any topological order $g_1, g_2, \ldots, g_i, \ldots, g_{|C|}$ on the gates of the constant-free circuit C over $\mathbb{Q}$ (that is, if g_j has a directed path to g_k in C then $j < k$) and proceed by induction on $|C|$ to eliminate rationals from the circuit (identifying the gate g_i with the sub-circuit of C for which g_i is its root, and denoting by $\widehat{g}_i$ the polynomial computed by g_i).

Induction statement: Let $g_1, \ldots, g_s$ be the topologically ordered gates of a constant-free circuit C over $\mathbb{Q}$, where $s = |C|$. Then, there exists (division-free) constant-free circuits over $\mathbb{Z}$ consisting of the corresponding topologically ordered gates $g_{11}, \ldots, g_{1a_1}, g_{21}, \ldots, g_{2a_2}, \ldots, g_{s1}, \ldots, g_{sa_s}$, such that for *every* $i \leq s$:

1. $a_i \leq 4$ and g_{ia_i} is a constant-free and division-free circuit computing the polynomial $M_i \cdot g_i$ over $\mathbb{Z}$, for some nonzero integer M_i (again, identifying the gate g_{ia_i} with the sub-circuit for which it is a root);

2. The integer M_i is constructed as a part of the circuit (except for the trivial case $M_i = 1$). More precisely, there exists a division-free constant-free (sub-)circuit $g_{j\ell}$, for $j \leq i, \ell \leq 4$ that computes M_i. In particular $\tau(M_i) \leq 4i$.

Base case: g_i is a variable or a constant in $\{-1, 0, 1\}$. Hence, we put $g_{i1} := g_i$, $a_i = 1$, and $M_i = 1$.

Induction step: In the case of a binary gate $g_i = g_j \circ g_\ell$, for $\circ \in \{\times, +, \div\}$ (where $j, \ell < i$), by induction hypothesis we already have division-free constant-free circuits g_{ja_j} and $g_{\ell a_\ell}$ computing the polynomials $M_j g_j$ and $M_\ell g_\ell$, respectively, for some integers M_j, M_ℓ that are also computed as part of the circuit.

Case 1: g_i is a division gate computing g_j/g_ℓ, where, by definition of circuits over $\mathbb{Q}$, g_ℓ is a constant-free circuit computing a nonzero constant.

By induction hypothesis item 1 we have already constructed the two division-free and constant-free circuits g_{ja_j} and $g_{\ell a_\ell}$, where

$$\widehat{g}_{ja_j} = M_j g_j \quad \text{and} \quad \widehat{g}_{\ell a_\ell} = M_\ell g_\ell \,,$$

and $M_j g_j$ is a polynomial over $\mathbb{Z}$, for some nonzero integer M_j, and $M_\ell g_\ell$ is an integer number for some nonzero integer M_ℓ (g_ℓ can be rational).

By induction hypothesis item 2, M_j and M_ℓ are already computed by some division-free gates in the circuit. We thus put $a_i = 2$ and

$$g_{i1} := M_j \cdot g_{\ell a_\ell} = M_j M_\ell g_\ell \quad \text{and} \quad g_{i2} := M_\ell \cdot g_{ja_j} = M_\ell M_j g_j \,,$$

(that is, g_{i1} is a product gate that connects to the two previously constructed gates computing the two integers M_j and $g_{\ell a_\ell}$).

Letting $M_i = M_j M_\ell g_\ell$, we get that g_{i1} is a division-free circuit computing the integer M_i, and g_{i2} is a division-free circuit computing the polynomial $M_j M_\ell g_\ell \cdot (g_j/g_\ell) = M_i \cdot g_i$.

Case 2: $g_i = g_j \cdot g_\ell$. In this case $a_i = 2$ and $M_i = M_j M_\ell$, and we put $g_{i2} := g_{ja_j} \cdot g_{\ell a_\ell}$ and $g_{i1} := M_i \cdot M_j$, where M_i, M_j are two integers that are already computed (with a constant-free division-free and variable-free sub-circuits).

Case 3: $g_i = g_j + g_\ell$. In this case $a_i = 4$, $M_i = M_j M_\ell$, and we put $g_{i4} := M_\ell \cdot g_{ja_j} + M_j \cdot g_{\ell a_\ell}$, namely, we add three gates g_{i2}, g_{i3}, g_{i4} (two products, both of which connects to previous gates, and one addition to add these two products). Finally, we put $g_{i1} := M_i \cdot M_j$, where M_i, M_j are two integers that are computed already (with a constant-free division-free sub-circuits). □

An immediate corollary of Prop. 4.3 is:

Corollary 4.4 (from $\mathrm{IPS}_{\mathbb{Q}}$ to $\mathrm{IPS}_{\mathbb{Z}}$)**.** *Boolean $\mathrm{IPS}_{\mathbb{Z}}$ simulates Boolean $\mathrm{IPS}_{\mathbb{Q}}$, in the following sense: if there exists a size-s constant-free Boolean IPS proof over $\mathbb{Q}$ from $\overline{\mathcal{F}}$ of H, for $\overline{\mathcal{F}}$ a set of assumptions written as constant-free algebraic circuits over $\mathbb{Z}$ and H a constant-free algebraic circuit over $\mathbb{Z}$, then there exists a size $\leq 4s$ constant-free Boolean $\mathrm{IPS}_{\mathbb{Z}}$ proof of $M \cdot H$, for some $M \in \mathbb{Z} \setminus \{0\}$, such that $\tau(M) \leq 4s$.*

Theorem 4.5. *Under the Shub and Smale Hypothesis, there are no* poly(n)*-size constant-free (Boolean) IPS refutations of the binary value principle* BVP_n *over* $\mathbb{Q}$*.*

Proof: Given Cor. 4.4, it suffices to prove the statement for constant-free (Boolean) $\mathrm{IPS}_{\mathbb{Z}}$.

We proceed to prove the contrapositive. Suppose that the binary value principle $1+\sum_{i=1}^{i=n} 2^{i-1}x_i = 0$ has a constant-free $\mathrm{IPS}_{\mathbb{Z}}$ refutation (using only the boolean axioms) of size poly(n). We will show that there is a sequence of nonzero natural numbers c_m such that $\tau(c_m m!) \leq (\log m)^c$, for all $m \geq 2$, where c is a constant independent of m. In other words, we will show that $(c_m m!)_{m=1}^{\infty}$ is easy.

Assume that $C(\overline{x}, y, \overline{z})$ is the polynomial-size constant-free Boolean $\mathrm{IPS}_{\mathbb{Z}}$ refutation of $1 + \sum_{i=1}^{i=n} 2^{i-1}x_i = 0$ (here we only have a single placeholder variable y for the single non-Boolean axiom, that is, the binary value principle). For simplicity, denote $G(\overline{x}) = 1 + \sum_{i=1}^{i=n} 2^{i-1}x_i$, $F_i(\overline{x}) = x_i^2 - x_i$, and $\overline{F}(\overline{x}) = \overline{x}^2 - \overline{x}$.

We know that there exists an integer constant $M \neq 0$ such that

$$C\left(\overline{x}, G(\overline{x}), \overline{F}(\overline{x})\right) = M\,. \tag{2}$$

For every integer $0 \leq k < 2^n$ we denote by $\overline{b}_k := (b_{k1}, \ldots, b_{kn}) \in \{0,1\}^n$ its (positive, standard) binary representation, that is, $k = \sum_{i=1}^{i=n} b_{ki}2^{i-1}$. Then, $F_i(\overline{b}_k) = 0$ and $G(\overline{b}_k) = 1 + k$, for all $1 \leq i \leq n$, $0 \leq k < 2^n$. Hence, by eq. 2:

$$C(b_{k1}, \ldots, b_{kn}, 1+k, \overline{0}) = M, \quad \text{for every integer } 0 \leq k < 2^n. \tag{3}$$

Claim 4.6. *M is divisible by every prime number less than 2^n.*

Proof of claim: For a fixed $0 \leq k < 2^n$ and its binary representation $b_{k1}, \ldots, b_{kn}$, consider $g(y) = C(b_{k1}, \ldots, b_{kn}, y, \overline{0})$ as a univariate polynomial in $\mathbb{Z}[y]$. Then, $g(1+k) = M$ by eq. 3, and $g(0) = 0$ holds since $C(b_{k1}, \ldots, b_{kn}, 0, \overline{0}) = 0$, by the definition of IPS. Because $g(0) = 0$ and g is not identically 0, we know that $g(y) = y \cdot g^{\star}(y)$, for some $g^{\star}(y) \in \mathbb{Z}[y]$, meaning that $g(1+k) = (1+k) \cdot g^{\star}(1+k) = M$. Since $g^{\star}(y)$ is an integer polynomial, this implies that M is a multiple of $1+k$.

Overall, this argument shows that for every $1 \leq p \leq 2^n$, M is divisible by p, and in particular M is divisible by every prime number less than 2^n. ■Claim

Note that once we substitute the all-zero assignment $\overline{0}$ into eq. 2, we obtain a constant-free algebraic circuit of size poly(n) with no variables computing M, thus $\tau(M) =$ poly(n). Then we can compute M^{2^n} using a constant-free algebraic circuit of size poly(n) by taking M to the power 2, n many times (that is, using n repeated squaring), yielding $\tau(M^{2^n}) =$ poly(n).

Claim 4.7. *The exponent of every prime factor in $(2^n)!$ is at most 2^n.*

Proof of claim: We show that for every number $k \in \mathbb{N}$, the power of every prime factor of $k!$ is at most k. Let $p_1^{t_1} \cdots p_r^{t_r}$ be the prime factorisation of $k!$, namely $k! = p_1^{t_1} \cdots p_r^{t_r}$ where each p_i is a prime number and $p_i \neq p_j$, for all $i \neq j$. To compute t_i we consider the k factors $k, (k-1), \ldots, 1$,

in $k! = k \cdot (k-1) \cdots 1$, out of which only each p_ith number is divisible by p_i, hence only $\lfloor \frac{k}{p_i} \rfloor$ numbers are divisible by p_i. Consider now only these $\lfloor \frac{k}{p_i} \rfloor$ numbers in $k!$ which are divisible by p_i, and write them as $p_i \cdot \lfloor \frac{k}{p_i} \rfloor, p_i \cdot (\lfloor \frac{k}{p_i} \rfloor - 1), \ldots, p_i \cdot 1$. Now we need once again to factor out the p_i products in $\lfloor \frac{k}{p_i} \rfloor, \lfloor \frac{k}{p_i} \rfloor - 1, \ldots, 1$. Hence, as before, we conclude that in these $\lfloor \frac{k}{p_i} \rfloor$ numbers only $\lfloor \frac{\lfloor \frac{k}{p_i} \rfloor}{p_i} \rfloor \leq \lfloor \frac{k}{p_i^2} \rfloor$ are divisible by p_i. Continuing in a similar fashion we obtain the equation $t_i \leq \lfloor \frac{k}{p_i} \rfloor + \lfloor \frac{k}{p_i^2} \rfloor + \lfloor \frac{k}{p_i^3} \rfloor + \cdots \leq \frac{k}{p-1}$. ■Claim

Consider the poly(n)-size circuit for M^{2^n} that exists by assumption. Since M is divisible by every prime number between 1 and 2^n, and since every prime factor of $(2^n)!$ is clearly at most 2^n, we get that M^{2^n} is divisible by the 2^n-th power of *each* prime factor of $(2^n)!$. By Claim 4.7 the power of every prime factor of $(2^n)!$ is at most 2^n, and so M^{2^n} is divisible by $(2^n)!$. We conclude that there are nonzero numbers $c_n \in \mathbb{N}$ such that the sequence $\{c_n \cdot (2^n)!\}_{n=1}^{\infty}$ is computable by a sequence of constant-free algebraic circuits of size poly(n), that is, $\tau(c_n \cdot (2^n)!) \leq n^c$ for some constant c independent of n. It remains to show that not only the multiples of factorials *of powers of* 2 are easy, but also the multiples of factorials of *all* natural numbers are easy.

For every natural number m, let $n \in \mathbb{N}$ be such that $2^{n-1} \leq m \leq 2^n$. Because $(2^n)!$ is clearly divisible by $m!$, there exists some $c_m \in \mathbb{N}$, such that $c_n \cdot (2^n)! = c_m \cdot m!$, where c_n is the natural number for which we have showed the existence of poly(n)-size constant-free circuit computing $c_n \cdot (2^n)!$. Hence, this same circuit also computes $c_m \cdot m!$, meaning that $\tau(c_m \cdot m!) \leq n^b \leq (\log(2m))^b \leq (\log m)^c$, for some constants b and c independent of m. □

Why does an IPS lower bound on BVP not lead to Extended Frege lower bounds? Given that IPS (of possibly exponential degree) simulates Extended Frege (EF) [28, 47], it is interesting to consider why our conditional IPS lower bound for the BVP does not imply a conditional EF lower bound. Simply put, the answer is that the BVP is not a propositional tautology (or a direct translation of one), and moreover we do not know how to efficiently derive in IPS any propositional contradiction from the BVP (note that if we could efficiently derive in IPS a propositional contradiction, for example an unsatisfiable CNF formula [encoded as polynomials in the standard way], we would immediately get a conditional Extended Frege lower bound).

Notice furthermore, that we can encode the BVP as a propositional tautology stating that the carry-save addition of the n numbers in the BVP has sign-bit 0 (and hence the addition is positive), but the problem is that there is no apparent way to efficiently derive in IPS this encoding from the BVP principle itself, *because from a polynomial equation like $f = 0$ we cannot in general efficiently derive in IPS that the sign-bit of f is zero*, as we now explain.

One can think of the following translation of the BVP into a propositional tautology: we consider the addition of n numbers $2^{i-1}x_i$, for $x_i \in \{0,1\}$ and $i = 1, \ldots, n$. Each $2^{i-1}x_i$ is written as a bit-vector $\mathbf{v}_i$ of at most n bits, in the two's complement notation. Each bit in $\mathbf{v}_i$ can be written as a polynomial-size Boolean circuit in the single Boolean variable x_i. Using carry-save addition we can construct a polynomial-size in n Boolean circuit C computing the sign-bit of the addition of these n bit vectors $\sum_{i=1}^{n} \mathbf{v}_i$ (this is done as in Sect. 6). Now, the BVP can be encoded propositionally as the contradiction $C \equiv \top$ (namely, the sign-bit of the addition of bit-vectors addition is logically equivalent to *true* meaning that the sign is negative; note that this is indeed a contradiction). Although we can think of the above natural propositional formulation of BVP, there is no apparent way to efficiently in IPS derive this propositional formulation from BVP.

4.2 IPS over Rational Functions and the τ-Conjecture

Here we deal with IPS operating over the field of rational functions in the (new) indeterminate y. This will allow us to formulate an interesting version of the binary value principle. Roughly speaking, this version expresses the fact that the BVP is "almost always" unsatisfiable. More precisely, consider the equation $\sum_{i=1}^{n} 2^{i-1}x_i = y$. This equation is unsatisfiable for most y's, when y is substituted by an element from $\mathbb{Q}$. In the setting of IPS refutations over the field of rational functions in the indeterminate y, refuting $\sum_{i=1}^{n} 2^{i-1}x_i = y$ would correspond to refuting $\sum_{i=1}^{n} 2^{i-1}x_i = M$, for all $M \in \mathbb{Q}$ but a finite set of numbers from $\mathbb{Q}$ (see below).

We shall prove a super-polynomial lower bound on $\sum_{i=1}^{n} 2^{i-1}x_i = y$, over the fields of rational functions in the indeterminate y, subject to the τ-conjecture.

Definition 4.8 ($\mathbb{Q}$-rational functions)**.** *Denote by $\mathbb{Q}(y)$ the field of* $\mathbb{Q}$-rational functions in y*, that is, all functions $f(y) : \mathbb{Q} \to \mathbb{Q}$ such that there exist $P(y) \in \mathbb{Q}[y]$ and nonzero $Q(y) \in \mathbb{Q}[y]$ with $f(y) = \frac{P(y)}{Q(y)}$.*

In particular, in this system one can consider refutations of $\sum_{i=1}^{n} 2^{i-1}x_i + y = 0$, where x_i are Boolean variables (the Boolean axioms $x_i^2 - x_i = 0$ are included in the initial axioms). In this section we will be using the concept of a *linear* IPS refutation (proved to be polynomially equivalent to general IPS, at least in the unit-cost model where each coefficient appearing in an algebraic circuit is considered to contribute only 1 to the overall size of the circuit), defined in Forbes *et al* [19]:

Definition 4.9 ([19])**.** *An* IPS-LIN$_{\mathbb{Q}(y)}$-certificate *of the unsatisfiability of a system of polynomial equations $F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_m(\overline{x}) = 0$ is a set of polynomials $(H_1(\overline{x}), \ldots H_m(\overline{x}))$, where each $H_i(\overline{x}) \in \mathbb{Q}(y)[x_1, \ldots, x_n]$, such that $F_1(\overline{x}) \cdot H_1(\overline{x}) + \cdots + F_m(\overline{x}) \cdot H_m(\overline{x}) = 1$ (as a formal polynomial equation).*

We assume that the F_j's include the Boolean axioms $x_i^2 - x_i$ for every variable x_i. The system is complete for this case, as discussed in the next subsection.

Note that once we have an IPS-LIN$_{\mathbb{Q}(y)}$-certificate of a system of equations that include the Boolean axioms and the equation $\sum_i x_i a_i = y$, we can substitute for y any constant except for the finite number of roots of the denominators of H_i's and get a valid IPS-LIN$_{\mathbb{Q}}$ refutation. *Thus an IPS-LIN$_{\mathbb{Q}(y)}$-certificate can be viewed as a single proof for all but finitely many values of y.*

To show this concept is meaningful, we first show a short IPS-LIN$_{\mathbb{Q}(y)}$ proof of $\sum_{i=1}^{n} a_i x_i = y$ for small scalars a_i. Then we demonstrate a lower bound for $a_i = 2^{i-1}$ modulo the τ-conjecture.

We start with precise definitions of the complexity of IPS-LIN$_{\mathbb{Q}(y)}$-proofs and related completeness issues.

4.2.1 Computational complexity of elements in $\mathbb{Q}(y)$

To compute elements of $\mathbb{Q}(y)$, we extend the definition of a constant-free circuit by allowing the use of gates for y. The definition of a constant-free circuit over $\mathbb{Q}(y)$ thus mimics Definition 4.2, but we allow now the constant y in addition to $-1, 0, 1$ (notice that y is indeed a constant in terms of the polynomials in $\mathbb{Q}(y)[x_1, \ldots, x_n]$).

Note that the system we consider is complete for the Boolean case, that is, for every unsatisfiable (over $\{0, 1\}$) set of polynomial equations involving coefficients in $\mathbb{Q}(y)$ that contains the Boolean axiom $x^2 - x = 0$ for every variable x, there is an IPS-LIN$_{\mathbb{Q}(y)}$ certificate. Indeed, the set of equations remains unsatisfiable in the algebraic closure of $\mathbb{Q}(y)$ (since every solution must satisfy $x^2 - x = 0$), and thus by Hilbert's Nullstellensatz the linear system that has H_i's coefficients as variables and expresses that the H_i's form a valid certificate, has a solution. Since the coefficients of this linear system are in $\mathbb{Q}(y)$, so must be (some) solution.

Remark 4.10. *Forbes et al. [19] proved that IPS is polynomially equivalent to IPS-LIN when the scalars are given for free (that is, do not count towards the proof size). We believe that a similar transformation can be made in the constant-free model to establish the equivalence between* $IPS_{\mathbb{Q}(y)}$ *and* $IPS\text{-}LIN_{\mathbb{Q}(y)}$*, however, we did not verify this.*

4.2.2 Upper Bound

Proposition 4.11. *Suppose we have a system of polynomial equations* $F_0(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$*,* $F_i \in \mathbb{Q}(y)[x_1, \ldots, x_n]$*, where* $F_0(\overline{x}) = y + \sum_{i=1}^{i=n} a_i x_i$*,* $a_i \in \mathbb{N}$ *and* $F_i(\overline{x}) = x_i^2 - x_i$*. Then, there is an* $IPS\text{-}LIN_{\mathbb{Q}(y)}$ *certificate of this system consisting of* $H_0(\overline{x}), \ldots, H_{n+1}(\overline{x})$*, where each* $H_i(\overline{x})$ *can be computed by a constant-free algebraic circuit over* $\mathbb{Q}(y)$ *of size* $\text{poly}(a_1 + \cdots + a_n)$*.*

Proof: We will construct our proof by induction on n. In each step of our induction we will use the following notation:

- For each $0 \leq k \leq n$, $0 \leq t \leq \sum_{i=k+1}^{n} a_i$, $t, k \in \mathbb{Z}$ we define $G_{k,t}(\overline{x}) = y + t + a_1 x_1 + \ldots + a_k x_k$.
- In the induction step we will build the collection of certificates $H_{k+1,0,t}(\overline{x}), \ldots, H_{k+1,n,t}(\overline{x})$ for the systems of polynomial equations $G_{k+1,t}(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$ for each $0 \leq t \leq \sum_{i=k+1}^{n} a_i$.

Base case: suppose $G_{0,t}(\overline{x}) = y + t$, $t \in \mathbb{N}, t \leq \sum_{i=1}^{n} a_i$. Then we can take $H_{0,0,t}(\overline{x}) = \frac{1}{y+t}$ and $H_{0,i,t} = 0$ where $1 \leq i \leq n$, $i \in \mathbb{N}$ as an $\text{IPS-LIN}_{\mathbb{Q}(y)}$ certificate for a system of polynomial equations $G_{0,t}(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$.

Induction step: suppose we have already built certificates $H_{k,0,t}(\overline{x}), \ldots, H_{k,n,t}(\overline{x})$ for the systems of polynomial equations $G_{k,t}(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$. Now we will build certificates $H_{k+1,0,t}(\overline{x}), \ldots, H_{k+1,n,t}(\overline{x})$ for the systems of polynomial equations $G_{k+1,t}(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$ where $G_{k+1,t}(\overline{x}) = y + t + a_1 x_1 + \ldots a_{k+1} x_{k+1}$, $t \in \mathbb{Z}$, $0 \leq t \leq \sum_{i=k+2}^{n} a_i$. There are the following cases:

1. If $i > k+1$, then we will take $H_{k+1,i,t}(\overline{x}) = 0$.
2. If $i = k+1$, then we will take $H_{k+1,i,t}(\overline{x}) = a_{k+1}(H_{k,0,t}(\overline{x}) - H_{k,0,t+a_{k+1}}(\overline{x}))$.
3. If $0 \leq i < k+1$, then we will take $H_{k+1,i,t}(\overline{x}) = x_{k+1} H_{k,i,t+a_{k+1}}(\overline{x}) + (1 - x_{k+1}) H_{k,i,t}(\overline{x})$.

The main idea of this construction is the case analysis for $x_{k+1} = 0$, $x_{k+1} = 1$, that is,

$$(y + t + a_1 x_1 + \ldots + a_{k+1} x_{k+1}) x_{k+1} - a_{k+1}(x_{k+1}^2 - x_{k+1}) = (y + t + a_{k+1} + a_1 x_1 + \ldots a_k x_k) x_{k+1}$$

and

$$(y + t + a_1 x_1 + \ldots + a_{k+1} x_{k+1})(1 - x_{k+1}) + a_{k+1}(x_{k+1}^2 - x_{k+1}) = (y + t + a_1 x_1 + \ldots a_k x_k)(1 - x_{k+1}).$$

which means that (using the induction hypothesis)

$$\begin{aligned}((y + t + a_1 x_1 + \ldots a_{k+1} x_{k+1}) x_{k+1} - a_{k+1}(x_{k+1}^2 - x_{k+1})) H_{k,0,t+a_{k+1}}(\overline{x}) + \\ + (x_1^2 - x_1) x_{k+1} H_{k,1,t+a_{k+1}}(\overline{x}) + \ldots + (x_k^2 - x_k) x_{k+1} H_{k,k,t+a_{k+1}}(\overline{x}) = x_{k+1}\end{aligned}$$

and

$$\begin{aligned}((y + t + a_1 x_1 + \ldots a_{k+1} x_{k+1})(1 - x_{k+1}) + a_{k+1}(x_{k+1}^2 - x_{k+1})) H_{k,0,t}(\overline{x}) + \\ + (x_1^2 - x_1)(1 - x_{k+1}) H_{k,1,t}(\overline{x}) + \ldots + (x_k^2 - x_k)(1 - x_{k+1}) H_{k,k,t}(\overline{x}) = 1 - x_{k+1}\end{aligned}$$

Summing up the equations for both cases, due to the fact that $(1 - x_{k+1}) + x_{k+1} = 1$ we get $G_{k+1,t}H_{k+1,0,t} + \sum_{i=1}^{n} F_i H_{k+1,i,t} = 1$.

In each step of our induction we create no more than poly$(a_1 + \cdots + a_n)$ new gates computing algebraic circuits for $\mathbb{Q}(y)[x_1, \ldots, x_n]$-polynomials $H_{k,0,t}(\overline{x}), \ldots, H_{k,n,t}(\overline{x})$ for each $0 \leq t \leq \sum_{i=k+1}^{n} a_i$. Thus, we can take $H_0(\overline{x}) = H_{n,0,0}(\overline{x}), \ldots, H_n(\overline{x}) = H_{n,n,0}(\overline{x})$ to conclude our proof. □

4.2.3 Lower Bound

Lemma 4.12. *Suppose we have a constant-free circuit C over $\mathbb{Q}(y)$ of size M computing a polynomial in $\mathbb{Q}(y)[x_1, \ldots, x_n]$ that is a rational function $f(y, x_1, \ldots, x_n)$. Then there are two constant-free circuits over $\mathbb{Z}$ of size less than $4M$ computing polynomial functions $P(y, x_1, \ldots, x_n) \in \mathbb{Z}[y, x_1, \ldots, x_n]$ and $Q(y) \in \mathbb{Z}[y]$ such that $f(y, x_1, \ldots, x_m) = \frac{P(y,x_1,\ldots,x_n)}{Q(y)}$.*

Proof: Similar to Prop. 4.3, the proof of this lemma is similar to that given by Valiant in [61].

Consider any topological order $g_1, \ldots, g_M$ on the gates of C. We will gradually rewrite our circuit starting from g_1. Assume that we have already done the job for $g_1, \ldots, g_k$, that is, for each $i \leq k$ there are appropriate algebraic circuits for polynomial functions $P_i(y, x_1, \ldots, x_n) \in \mathbb{Z}[y, x_1, \ldots, x_n]$ and $Q_i(y) \in \mathbb{Z}[y]$ such that $g_i = \frac{P_i}{Q_i}$. We now augment these circuits to compute the polynomials for g_k.

Here are all possible cases:

1. g_{k+1} is a variable x_j, then $P_{k+1} = x_j$, $Q_{k+1} = 1$.
2. g_{k+1} is a constant from $\mathbb{Q}(y)$ (that is, $0, -1, 1, y$), then P_{k+1} computes this constant, and $Q_{k+1} = 1$.
3. $g_{k+1} = \frac{g_i}{g_j}$, where $i, j \leq k$. In this case $P_i \in \mathbb{Q}(y)$ because of the structure of our $\mathbb{Q}(y)[x_1, \ldots, x_n]$-circuit. Then $Q_{k+1} = Q_i P_j$ and $P_{k+1} = P_i Q_j$ and sizes of the circuits for P_{k+1} and Q_{k+1} are less than $4 \cdot (k + 1)$.
4. $g_{k+1} = g_i \cdot g_j$, where $i, j \leq k$. Then $Q_{k+1} = Q_i Q_j$ and $P_{k+1} = P_i P_j$ and sizes of the circuits for P_{k+1} and Q_{k+1} are less than $4 \cdot (k + 1)$.
5. $g_{k+1} = g_i + g_j$. Then $P_{k+1} = P_i Q_j + P_j Q_i$ and $Q_{k+1} = Q_i Q_j$ and sizes of the circuits for P_{k+1} and Q_{k+1} are less than $4 \cdot (k + 1)$.

We can conclude our proof by taking P_M and Q_M as P and Q, respectively. □

Theorem 4.13. *Suppose a system of polynomial equations $F_0(\overline{x}) = F_1(\overline{x}) = F_2(\overline{x}) = \cdots = F_n(\overline{x}) = 0$, $F_i \in \mathbb{Q}(y)[x_1, \ldots, x_n]$, where $F_0(\overline{x}) = y + \sum_{i=1}^{i=n} 2^{i-1} x_i$ and $F_i(\overline{x}) = x_i^2 - x_i$, has an IPS-LIN$_{\mathbb{Q}(y)}$ certificate $H_0(\overline{x}), \ldots, H_n(\overline{x})$, where each $H_i(\overline{x})$ can be computed by a* poly(n)*-size constant-free algebraic circuit over $\mathbb{Q}(y)$. Then, the τ-conjecture is false.*

Proof: Based on the above lemma, we can say that there are polynomials $P_i(y, x_1, \ldots, x_n) \in \mathbb{Z}[y, x_1, \ldots, x_n]$ and $Q_i(y) \in \mathbb{Z}[y]$ such that $H_i = \frac{P_i}{Q_i}$ for every i. Also we know that

$$(y + x_1 + \cdots + 2^{n-1} x_n)\frac{P_0}{Q_0} + (x_1^2 - x_1)\frac{P_1}{Q_1} + \cdots + (x_n^2 - x_n)\frac{P_n}{Q_n} = 1$$

So we can derive that

$$(y + x_1 + \cdots + 2^{n-1}x_n)P_0 \prod_{j=1}^{n} Q_j + (x_1^2 - x_1)P_1Q_0 \prod_{j=2}^{n} Q_j + \cdots + (x_n^2 - x_n)P_n \prod_{j=0}^{n-1} Q_j = \prod_{j=0}^{n} Q_j \quad (4)$$

Denote $\prod_{j=0}^{n} Q_j$ by $Q(y)$. From the above lemma we know that there is a constant-free circuit over $\mathbb{Z}$ of size poly(n) for $Q(y)$. Furthermore, for any integer y such that $0 \geq y > -2^n$, there are values for x_i (namely, the bit expansion of $-y$) such that the left hand side of eq. 4 is zero, and hence $Q(y) = 0$. However, it contradicts the τ-conjecture. □

4.2.4 Some remarks on IPS over fields of rational functions and VNP with division

Consider BVP_n as a particular case of the general subset sum equality $\sum a_i x_i + c = 0$. We can write the polynomial coefficients in an IPS (or Nullstellensatz) refutation of this equality over the field $\mathbb{Q}(\overline{w})$ as follows (where $g(\overline{x})$ is the polynomial coefficient of BVP_n in the refutation and the $g_i(\overline{x})$'s are the polynomial coefficients of the Boolean axioms for x_i in the refutation):

$$\begin{aligned}
g(\overline{x}) &= \sum_{\bar{w}\in\{0,1\}^n} (1 - x_1 + 2x_1w_1 - w_1)\cdots(1 - x_n + 2x_nw_n - w_n)/(a_1w_1 + \cdots + a_nw_n + c),\\
g_i(\overline{x}) &= a_i \sum_{\overline{w}\in\{0,1\}^n} (1 - 2w_i)(1 - x_1 + 2x_1w_1 - w_1)\cdots\\
&\quad (1 - x_{i-1} + 2x_{i-1}w_{i-1} - w_{i-1})(1 - x_{i+1} + 2x_{i+1}w_{i+1} - w_{i+1})\cdots\\
&\quad (1 - x_n + 2x_nw_n - w_n)/(a_1w_1 + \cdots + a_nw_n + c).
\end{aligned}$$

Note that g is the inverse of $a_1x_1 + \cdots + a_nx_n + c$ in the quotient ring $\mathbb{Q}(\overline{w})[\overline{x}]/(x_1^2 - x_1, \ldots, x_n^2 - x_n)$ and that after multiplying out the expression $g(a_1x_1 + \cdots + a_nx_n + c) + g_1(x_1^2 - x_1) + \cdots + g_n(x_n^2 - x_n)$ in the ring $\mathbb{Q}[\overline{x}](\overline{w})$, we obtain a multilinear expression in $x_1, \ldots, x_n$, meaning that it equals 1.

Notice also that we cannot prima facie conclude that such an IPS refutation is computable in $\text{VNP}_{\mathbb{Q}(\overline{w})}$, namely, we do not know how to compute g and the g_i's in $\text{VNP}_{\mathbb{Q}(\overline{w})}$, because we used division by polynomials to compute g and the g_i's. However, we can consider defining VNP^0_{div} as the class of polynomials represented in the form $\sum_{\bar{w}\in\{0,1\}^n} f(\overline{x}, \overline{w})/g(\bar{w})$ for $f, g \in \text{VP}_{\mathbb{Q}}$. Hence, we can compute g and the g_i's in VNP^0_{div}. *Therefore, if* $\text{VNP}^0_{div} \subseteq \text{VP}^0_{\mathbb{Q}}$*, IPS admits polynomial-size refutations of* BVP_n (here $\text{VP}^0_{\mathbb{Q}}$ is the class of polynomials that can be computed by polynomial-size constant-free circuits that allow dividing by constants similarly to Definition 4.2). In other words, a super-polynomial lower bound against IPS refutations over $\mathbb{Q}$ implies a separation of VNP^0_{div} from $\text{VP}^0_{\mathbb{Q}}$, and assuming that $\text{VNP}^0_{div} \subseteq \text{VNP}^0_{\mathbb{Q}}$, such a lower bound implies $\text{VNP}^0_{\mathbb{Q}} \neq \text{VP}^0_{\mathbb{Q}}$. As of now we do not know whether $\text{VNP}^0 = \text{VP}^0$ yields polynomial-size constant-free IPS refutations of BVP_n.

5 The Cone Proof System

Here we define a very strong semi-algebraic proof system under the name Cone Proof System (CPS for short). Similarly to other semi-algebraic systems, CPS establishes that a collection of polynomial equations $\overline{\mathcal{F}} := \{f_i = 0\}_i$ and polynomial inequalities $\overline{\mathcal{H}} := \{h_i \geq 0\}_i$ over the integers, the rational numbers or the reals, are unsatisfiable over 0-1 assignments (or over ring-valued assignments, when desired (Definition 5.2)). In the spirit of the Ideal Proof System (IPS) of Grochow and Pitassi [28] we are going to define a refutation in CPS as a *single* algebraic circuit. Specifically, a CPS refutation is a circuit C that computes a polynomial that results from *positive-preserving* operations such as addition and product applied between the inequalities $\overline{\mathcal{H}}$ and themselves, as well as the use

of nonnegative scalars and arbitrary squared polynomials. In order to simulate in CPS the free use of *equations* from $\overline{\mathcal{F}}$ we incorporate in the set of inequalities $\overline{\mathcal{H}}$ the inequalities $f_i \geq 0$ and $-f_i \geq 0$ for each $f_i = 0$ in $\overline{\mathcal{F}}$ (we show that this enables one to add freely products of the polynomial f_i in CPS proofs, namely working in the ideal of $\overline{\mathcal{F}}$; see Sect. 5.1.1).

We need to formalise the concept of a cone as an algebraic circuit. For this we first introduce the notion of a *squaring gate*: let C be a circuit and v be a node in C. We call v a ***squaring gate*** if v is a product gate whose two incoming edges are emanating from the *same* node. Therefore, if we denote by w the single node that has two outgoing edges to the squaring gate v, then v computes w^2 (that is, the square of the polynomial computed at node w).

The following is a definition of a circuit computing polynomials in the cone of the $\overline{y}$ variables:

Definition 5.1 ($\overline{y}$-conic circuit)**.** *Let R be an ordered ring. We say that an algebraic circuit C computing a polynomial over $R[\overline{x}, \overline{y}]$ is a **conic circuit with respect to** $\overline{y}$, or **$\overline{y}$-conic** for short, if for every negative constant or a variable $x_i \in \overline{x}$, that appears as a leaf u in C, the following holds: every path p from u to the output gate of C contains a squaring gate.*

Informally, a $\overline{y}$-conic circuit is a circuit in which we assume that the $\overline{y}$-variables are nonnegative, and any other input that may be negative (that is, a negative constant or an $\overline{x}$-variable) must be part of a squared sub-circuit. Here are examples of $\overline{y}$-conic circuits (over $\mathbb{Z}$): y_1, $y_1 \cdot y_2$, $3 + 2y_1$, $(-3)^2$, x_1^2, $(3 \cdot -x_1 + 1)^2$, $(x_1 y_2 + y_1)^2$, $y_1 + \cdots + y_n$. On the other hand, -1, x_1, $x_1 \cdot y_2$, $-1 \cdot y_1 + 4$ are examples of non $\overline{y}$-conic circuits.

Note that if the $\overline{y}$-variables of a $\overline{y}$-conic circuit are assumed to take on non-negative values, then a $\overline{y}$-conic circuit computes only non-negative values. It is evident that $\overline{y}$-conic circuits can compute all and only polynomials that are in the cone of the $\overline{y}$ variables. In other words, if $\overline{y}$ are the variables $y_1, \ldots, y_m$, then there exists a $\overline{y}$-conic circuit $C(\overline{x}, \overline{y})$ that computes the polynomial $p(\overline{x}, \overline{y})$ iff $p(\overline{x}, \overline{y}) \in \text{cone}(y_1, \ldots, y_m) \subseteq R[\overline{x}, \overline{y}]$. Similarly, if $\overline{f}(\overline{x})$ is a sequence of polynomials $f_1(\overline{x}), \ldots, f_m(\overline{x})$, then there exists a $\overline{y}$-conic circuit $C(\overline{x}, \overline{y})$ such that $C(\overline{x}, \overline{f}(\overline{x})) = p(\overline{x})$ iff $p(\overline{x})$ computes a polynomial in $\text{cone}\left(\overline{f}(\overline{x})\right) \subseteq R[\overline{x}]$.

Deciding if a given circuit is $\overline{y}$-conic is in deterministic polynomial-time (see Claim 5.10). In what follows, we will write "conic" instead of "$\overline{y}$-conic" where the meaning of $\overline{y}$ is clear from the context.

We start by defining the most general version of CPS over an ordered ring.

Definition 5.2 (CPS over an ordered ring R; R-CPS)**.** *Consider a collection of polynomial inequalities $\overline{\mathcal{H}} = \{h_i(\overline{x}) \geq 0\}_{i=1}^{\ell}$ ("assumptions"), where all polynomials are from $R[x_1, \ldots, x_n]$, with R and ordered ring. We will denote the collection of inequalities and the collection of polynomials h_i by the same letter without further notice. An **R-CPS proof of** $p(\overline{x})$ **from** $\overline{\mathcal{H}}$, showing that $\overline{\mathcal{H}}$ semantically imply the polynomial inequality $p(\overline{x}) \geq 0$, is an algebraic circuit $C(\overline{x}, \overline{y})$ computing a polynomial in $R[\overline{x}, y_1, \ldots, y_\ell]$, such that:*

1. *$C(\overline{x}, \overline{y})$ is a $\overline{y}$-conic circuit; and*
2. *$C(\overline{x}, \overline{\mathcal{H}}) = p(\overline{x})$,*

where item 2 above is a formal polynomial identity in which the left hand side means that we substitute $h_i(\overline{x})$ for y_i, for $i = 0, \ldots, \ell$.

*The **size** of an R-CPS proof is the size of the circuit C (see Remark 5.3 item 3). The variables $\overline{y}$ are the* placeholder variables *since they are used as placeholders for the axioms. A CPS proof of a negative constant from $\overline{\mathcal{H}}$ is called an **R-CPS refutation of** $\overline{\mathcal{H}}$. In the case that R is a field we assume that this constant is -1.*

Remark 5.3. *1. CPS should be thought of as a way to derive valid polynomial inequalities from a set of polynomial equations and inequalities from $\mathbb{R}[\overline{x}]$. Loosely speaking, it is a circuit representation of the Positivstellensatz proof system (Definition 2.8), though in CPS the assumptions (more precisely, placeholder variables) may have powers greater than one. That is, whereas eq. 1 is* multilinear *in the h_i variables, CPS is not.*

2. *In the definition of R-CPS we do not use equations (and unlike the Positivstellensatz we do not make a derivation in the corresponding* ideal*). However, we are not losing any power doing this. Let $\overline{\mathcal{F}}$ be a set of equations, and denote $-\overline{\mathcal{F}} = \{-f\}_{f\in\overline{\mathcal{F}}}$ (similarly to inequalities, we denote a collection of polynomial equations and a collection of the corresponding polynomials by the same letter). First, observe that the assignments satisfying the set of* inequalities *$\overline{\mathcal{H}}, \overline{\mathcal{F}}, -\overline{\mathcal{F}}$ are exactly the assignments satisfying the equalities $\overline{\mathcal{F}}$ and inequalities $\overline{\mathcal{H}}$. We show in Thm. 5.13 that if we encode equalities in $\overline{\mathcal{F}}$ as pairs of inequalities in $\overline{\mathcal{F}}$, $-\overline{\mathcal{F}}$, we can derive any polynomial in the* ideal *generated by $\overline{\mathcal{F}}$ (and not merely in the cone of $\overline{\mathcal{F}}$), as required for equations (and similar to the definition of SoS), with at most a polynomial increase in size (when compared to IPS).* ***Therefore, we will sometimes speak about R-CPS refutations for sets of both equations and inequalities.***

3. *Note that we have defined the size of an R-CPS proof as the size of the circuit C. This can be taken to be the circuit-size in the unit-cost model in which coefficients are of size 1 over any ordered ring R, or using the constant-free circuit model when working over $\mathbb{Q}$ or $\mathbb{Z}$.*

Note that similar to other semi-algebraic proof systems devoid of the Boolean axioms R-CPS is not necessarily complete (while its soundness is shown in Prop. 5.8). We will consider its particular complete cases. In the case where R is a real closed field its completeness follows from the simulation of Positivstellensatz (Thm. 5.16). We will also consider "Boolean" CPS, where the assumptions ensure 0-1 solutions (Definition 5.4). Boolean CPS is complete for $R = \mathbb{Q}$ and even for $R = \mathbb{Z}$.

We now define the Boolean Cone Proof System. *By default, when referring to CPS we will be speaking about the Boolean version* and hence may suppress the work "Boolean".

Definition 5.4 ((Boolean) Cone Proof System (CPS))**.** *This is R-CPS as in Definition 5.2 that in addition the assumptions $\overline{\mathcal{H}}$ contain for every variable x, the inequalities (note that the first two inequalities mean that implicitly we work with the equality $x^2 - x = 0$):*

$$\begin{aligned} x^2 - x &\geq 0 \\ x - x^2 &\geq 0 \\ x &\geq 0 \\ 1 - x &\geq 0. \end{aligned}$$

Remark 5.5. *To derive polynomials in the ideal of $\overline{\mathcal{F}}$ we need to be able to multiply f_i and $-f_i$ (from $\overline{\mathcal{H}}$) by* any *(positive) polynomial in the $\overline{x}$ variables. There are two ways to achieve this in Boolean CPS: the first, is to use the Boolean axiom $x_i \geq 0$ in $\overline{\mathcal{H}}$. This allows to product f_i and $-f_i$ by any polynomial in the $\overline{x}$-variables. The second way, the one we use in Prop. 5.12 to show that CPS simulates IPS, is different and does not necessitate the addition of the axiom $x_i \geq 0$ to $\overline{\mathcal{H}}$. Since the second way does not use the Boolean axiom $x_i \geq 0$ in $\overline{\mathcal{H}}$ we can use it in CPS over an ordered ring, hence allowing the derivation of polynomials in the ideal of $\overline{\mathcal{F}}$ within the latter proof system.*

In order to refute propositional formulas in conjunctive normal form (CNF) in CPS we use the algebraic translation of CNFs (Definition 2.5), which is expressed as a set of polynomial equalities. We

show in Prop. 5.19 that CPS can efficiently translate CNF formulas written as polynomial equalities to the standard way in which CNF formulas are written as polynomial *inequalities*.

To exemplify a proof in CPS we provide the following simple proposition:

Proposition 5.6. *CPS admits a linear size refutation of the binary value principle* BVP_n.

Proof: To simplify notation we put $S := \sum_{i=1}^{n} 2^{i-1} \cdot x_i + 1$. The set of assumptions includes $S = 0$ and the Boolean assumptions. Then by the definition of CPS the conic circuit can use the following axioms:

$$\overline{\mathcal{H}} := \Big\{ x_1 \geq 0, \dots,\ x_n \geq 0,\ \ -S \geq 0,\ S \geq 0,\ x_1^2 - x_1 \geq 0, \dots, x_n^2 - x_n \geq 0, \\ -(x_1^2 - x_1) \geq 0, \dots,\ \ -(x_n^2 - x_n) \geq 0,\ 1 - x_1 \geq 0, \dots,\ 1 - x_n \geq 0 \Big\}.$$

Therefore, the CPS refutation of the binary value principle is defined as the following $\overline{y}$-*conic* circuit:

$$C(\overline{x}, \overline{y}) := \left(\sum_{i=1}^{n} 2^{i-1} \cdot y_i \right) + y_{n+1}, \tag{5}$$

where the placeholder variables $y_1, y_2, \dots, y_{4n+2}$ correspond to the axioms in $\overline{\mathcal{H}}$ in the order they appear above (note that most y_i's do not appear in eq. 5, because the corresponding axioms are not needed for the proof). Observe indeed that $C(\overline{x}, \overline{\mathcal{H}}) = C(\overline{x}, x_1, \dots, x_n, -S, \dots) = (\sum_{i=1}^{n} 2^{i-1} \cdot x_i) + (-S) = -1$. □

Observing the CPS refutation in eq. 5 we see that it is in fact already an SoS refutation:

Corollary 5.7. SoS *admits a linear monomial size refutation of the binary value principle* BVP_n.

5.1 Basic Properties of CPS and Simulations

CPS is a very strong proof system. In fact, of all proof systems with randomized polynomial-time verification, given concretely (namely, as a circuit or a sequence of circuits[8]), to the best of our knowledge CPS is the strongest to have been defined to this date. CPS simulates IPS as shown below, while we show that IPS simulates CPS only under the condition that there are short IPS refutations of the binary value principle.

Now we show that soundness and completeness of CPS hold over the the same rings for which Positivestellensatz is sound and complete, whereas, similar to IPS, the probabilistically polynomial-time verifiability of CPS reduces to polynomial identity testing.

Proposition 5.8 (Soundness and completeness)**.** *Boolean CPS is sound and complete over every ordered ring, and* $\mathbb{F}$*-CPS is sound and complete for every real closed field* $\mathbb{F}$*. More precisely, let* R *be an ordered ring, and* $\mathbb{F}$ *be a real closed field. Then, given a set of polynomial inequalities* $\overline{\mathcal{H}}$*, where all polynomials are from* $R[x_1, \dots, x_n]$ *(resp.,* $\mathbb{F}[x_1, \dots, x_n]$*) there exists a CPS (resp.,* $\mathbb{F}$*-CPS) refutation of* $\overline{\mathcal{H}}$*, iff there is no* $\{0, 1\}$ *assignment (resp.,* $\mathbb{F}$*-assignment) satisfying* $\overline{\mathcal{H}}$*.*

[8] Though one should be aware of "non-standard" "concrete" proof systems that extend Extended Frege with additional axiom schemes expressing for instance the reflection principle for some very strong proof system, or the correctness of certain computations (e.g., that a given circuit computes correctly a certain function).

Proof: The soundness of Boolean CPS and $\mathbb{F}$-CPS is clear: assume that $C(\overline{x}, \overline{y})$ is a CPS refutation of $\overline{\mathcal{H}}$. Assume by a way of contradiction that $\overline{\alpha}$ is a 0-1 assignment to $\overline{x}$ in that satisfies $\overline{\mathcal{H}}$ in the case of CPS (or an $\mathbb{F}$-assignment to $\overline{x}$ in the case of $\mathbb{F}$-CPS). The circuit $C(\overline{x}, \overline{y})$ is $\overline{y}$-conic and hence $C(\overline{\alpha}, \overline{\mathcal{H}}(\overline{\alpha}))$ is non-negative assuming that the inputs to the $\overline{y}$ variables (that is, $\overline{\mathcal{H}}(\overline{\alpha})$) are non-negative. Since $\overline{\alpha}$ satisfies $\overline{\mathcal{H}}$ we know that indeed $h_i(\overline{\alpha}) \geq 0$, for every $h_i(\overline{x}) \in \overline{\mathcal{H}}$. Therefore, $\widehat{C}(\overline{\alpha}, \overline{\mathcal{H}}(\overline{\alpha})) \geq 0$, which contradicts our assumption that $C(\overline{\alpha}, \overline{\mathcal{H}}(\overline{\alpha})) = -1$.

The completeness of Boolean CPS follows e.g., from the completeness of propositional Positivstellensatz and its simulation below (Thm. 5.16). The completeness of $\mathbb{F}$-CPS stems from Thm. 2.7 (which holds for every real closed field). □

Proposition 5.9. *A constant-free CPS proof over $\mathbb{Z}$ or $\mathbb{Q}$ can be checked for correctness in probabilistic polynomial time.*

Proof: Similar to IPS, we can verify condition 2 in Definition 5.2, that is $C(\overline{x}, \overline{\mathcal{H}}) = p(\overline{x})$, in probabilistic polynomial-time (formally, in coRP). Note that to check condition 2 we need to be able to do polynomial identity testing, which is in coRP for constant-free circuits in $\mathbb{Z}$. For $\mathbb{Q}$ we can use Prop. 4.3 to turn the identity to an identity over $\mathbb{Z}$.

For condition 1 in Definition 5.2 we need to check that C is a $\overline{y}$-conic circuit, which can be done in P (in fact NL) via the following claim:

Claim 5.10. *There is a non-deterministic logspace (thus polynomial-time) algorithm to determine if a circuit $C(\overline{x}, \overline{y})$ is a $\overline{y}$-conic circuit or not (over any ring)*

Proof of claim: We say that a directed path from a leaf u in C holding either a negative constant or an $\overline{x}$ variable to the output gate of C is *bad* if the path does not contain any squaring gate.

For each leaf u in C holding either a negative constant or an $\overline{x}$ variable we can determine the following property in NL: there *exists* a bad path from u to the output gate of C. This algorithm is in NL simply because nondeterministically we can go along a directed path from u to the output gate and check that no squaring gate was encountered along the way (we only need to record the current node and the current length of the path so to know when to terminate). This means that the complement problem of deciding that there does *not* exist a bad path from u to the output gate is in coNL, which is equal to NL by the Immerman–Szelepcsényi Theorem.

Our algorithm thus enumerates the leaves and checks that each of the leaves holding negative constants do not possess any bad path to the output gate. ■Claim □

Similar to IPS, as a corollary we get the following:

Corollary 5.11. *If constant-free CPS over $\mathbb{Z}$ or $\mathbb{Q}$ is p-bounded (namely, admits polynomial-size refutations for every unsatisfiable CNF formula) then* coNP *is in* MA*, yielding in particular the polynomial hierarchy collapse to the third level (cf. [45, 28]).*

5.1.1 CPS Simulates IPS

We now show that Boolean CPS (Definition 5.4) simulates Boolean IPS for the language of $\{0,1\}$-unsatisfiable sets of polynomial *equations* over any ordered ring. Similarly, $\mathbb{Q}$-CPS simulates IPS over $\mathbb{Q}$. We translate an input equality $f_i(\overline{x}) = 0$ into a pair of inequalities $f_i(\overline{x}) \geq 0$ and $-f_i(\overline{x}) \geq 0$. Note that an IPS proof is written as a general algebraic circuit (computing an element of an ideal), while a CPS proof is written as a more restrictive algebraic circuit, namely as a $\overline{y}$-conic circuit (computing an element of a cone). This means that in CPS a priori we cannot multiply an inequality by an arbitrary polynomial. We thus demonstrate how to do it when we have opposite-sign inequalities. For this purpose, we represent an arbitrary polynomial as the difference of two nonnegative expressions.

Proposition 5.12 (minus gate normalisation)**.** *Let $G(\overline{x})$ be an algebraic circuit computing a polynomial in the $\overline{x}$ variables over $\mathbb{Q}$. Then, there is an algebraic circuit of the form $G_P(\overline{x}) - G_N(\overline{x})$ computing the same polynomial as $G(\overline{x})$ where G_P and G_N are $\emptyset$-conic. The size of G_P, G_N is at most linear in the size of G.*

Proof: This is somewhat reminiscent of Strassen's conversion of a circuit with division gate to a circuit with only a single division gate at the top [58]. We are going to break inductively each node into a pair of nodes computing the positive and negative parts of the polynomial computed in that node. Formally, we define the circuits G_P, G_N (that may have common nodes) by induction on the size of G as follows:
Case 1: $G = x_i$, for $x_i \in \overline{x}$. Then, $G_P := \frac{1}{2}(x_i^2 + 1), G_N := \frac{1}{2}(x_i - 1)^2$.
Case 2: $G = \alpha$, for α a constant in the ring. Then

$$G_P := \alpha,\ G_N := 0, \quad \text{if } \alpha \geq 0;$$
$$G_P := 0,\ G_N := \alpha, \quad \text{if } \alpha < 0.$$

Case 3: $G = F + H$. Then, $G_P := F_P + H_P$ and $G_N := F_N + H_N$.
Case 4: $G = F \cdot H$. Then, $G_P := F_P \cdot H_P + F_N \cdot H_N$ and $G_N := F_P \cdot H_N + F_N \cdot H_P$.

The size of both G_P, G_N is $O(|G|)$, namely linear in the size of G. This is because we only add constantly many new nodes in G_P, G_N for any original node in G; note that since we construct a new *circuit* computing the same polynomial as G, we can re-use nodes computed already, in case 4: for example, F_P is the same node used in G_P and G_N (hence, indeed, the number of new added nodes for every original node in G is constant). □

Theorem 5.13. *$\mathbb{Q}$-CPS simulates algebraic IPS as a proof system for the language of unsatisfiable sets of polynomial equations over $\mathbb{Q}$. In other words, there exists a constant c such that for any polynomial $p(\overline{x})$ and a set of polynomial equations $\overline{\mathcal{F}}$ over $\mathbb{Q}$, if $p(\overline{x})$ has an IPS proof of size s from $\overline{\mathcal{F}}$ then there is a CPS proof of $p(\overline{x})$ from $\overline{\mathcal{F}}$ of size at most s^c. Furthermore, Boolean CPS simulates Boolean IPS (for any ordered ring).*

Remark 5.14. *It is easy to see that fractional $\mathbb{Q}$ coefficients are not needed in the case of Boolean systems, as Case 1 in Prop. 5.12 above simplifies to $G_P := x_i, G_N := 0$ when x_i's are nonnegative. This is the reason Boolean CPS simulates Boolean IPS over any ordered ring.*

Specifically, if $\overline{\mathcal{F}}$ is a set of polynomial equations with no 0-1 satisfying assignments and suppose that there is an IPS refutation of $\overline{\mathcal{F}}$ with size s, then there is a CPS refutation of $\overline{\mathcal{F}}$ with size at most s^c.

Proof of Thm. 5.13. We are going to simulate both the Boolean and the algebraic versions of IPS. The proof in both cases is the same.

Assume that $C(\overline{x}, \overline{y})$ is the IPS proof of $p(\overline{x})$ from $\overline{\mathcal{F}} = \{f_i(\overline{x}) = 0\}_{i=1}^{\ell}$, of size s, and let $\overline{y} = \{y_1, \ldots, y_\ell\}$ be the placeholder variables for the equations in $\overline{\mathcal{F}}$. We assume for simplicity that if we simulate the *Boolean* version of IPS the Boolean axioms $\overline{x}^2 - \overline{x}$ are also part of $\overline{\mathcal{F}}$ (while if we simulate the *algebraic* version of IPS these axioms are not part of $\overline{\mathcal{F}}$). We use the following claim which is proved by a standard process that factors out the $\overline{y}$ variables one by one:

Claim 5.15. *Let $C(\overline{x}, \overline{y})$ be a circuit of size s, where $\overline{y} = \{y_1, \ldots, y_\ell\}$ and such that $C(\overline{x}, \overline{0}) = 0$. Then C can be written as a sum of circuits with only a polynomial increase in size as follows: $C(\overline{x}, \overline{y}) = \sum_{i=1}^{\ell} y_i \cdot C_i(\overline{x}, \overline{y})$.*

Proof of claim: We proceed by a standard process to factor out the $\overline{y}$ variables one by one. Beginning with y_1 we get:

$$C(\overline{y},\overline{x}) = y_1 \cdot (C(1,\overline{y}',\overline{x}) - C(0,\overline{y}',\overline{x})) + C(0,\overline{y}',\overline{x}),$$

where $\overline{y}'$ denotes the vector of variables $(y_2,\ldots,y_\ell)$. In a similar manner we factor out the variable y_2 from $C(0,\overline{y}',\overline{x})$. Continuing in a similar fashion we conclude the claim. Notice that the size of the resulting circuit is $O(|C|^2)$, and that in the final iteration of the construction we factor out y_ℓ from $C(\overline{0},y_\ell,\overline{x})$ it must hold that $C(\overline{0},y_\ell,\overline{x}) = y_1 \cdot (C(\overline{0},1,\overline{x}) - C(\overline{0},0,\overline{x})) + C(\overline{0},0,\overline{x}) = y_1 \cdot C(\overline{0},1,\overline{x})$, because by assumption $C(\overline{0},0,\overline{x}) = 0$. ■Claim

By this claim we have

$$\begin{aligned} C(\overline{x},\overline{y}) &= \sum_{i=1}^{\ell} y_i \cdot C_i(\overline{x},\overline{y}) \\ &= \sum_{i=1}^{\ell} y_i \cdot C_{i,P}(\overline{x},\overline{y}) - \sum_{i=1}^{\ell} y_i \cdot C_{i,N}(\overline{x},\overline{y}), \end{aligned} \tag{6}$$

where $C_{i,P}(\overline{x},\overline{y}), C_{i,N}(\overline{x},\overline{y})$ are the positive and negative parts of $C_i(\overline{x},\overline{y})$, respectively, that exist by Prop. 5.12, written as circuits in which no negative constants occur (we do not need to distinguish between the variables $\overline{x}$ and $\overline{y}$ here).

We wish to construct now a CPS refutation of $\overline{\mathcal{F}}$. Our corresponding set of inequalities $\overline{\mathcal{H}}$ will consist of $f_i(\overline{x}) \geq 0, -f_i(\overline{x}) \geq 0$, for every $i \in [\ell]$. In total, $|\overline{\mathcal{H}}| = 2\ell$. Accordingly, our CPS refutation of $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$, will have 2ℓ placeholder variables for the axioms in $\overline{\mathcal{H}}$ denoted as follows: $\overline{y}_P$ are the ℓ placeholder variables $y_{i,P}$ corresponding to $f_i(\overline{x}) \geq 0$, $i \in [l]$, $\overline{y}_N$ are the ℓ placeholder variables $y_{i,N}$ corresponding to $-f_i(\overline{x}) \geq 0$, $i \in [l]$.

Since $C_{i,P}$ and $C_{i,N}$ are $\emptyset$-conic circuits,

$$\sum_{i=1}^{\ell} y_{i,P} \cdot C_{i,P}(\overline{x},\overline{y}_P,\overline{y}_N) + \sum_{i=1}^{\ell} y_{i,N} \cdot C_{i,N}(\overline{x},\overline{y}_P,\overline{y}_N)$$

is a $(\overline{y}_P,\overline{y}_N)$-conic circuit. It constitutes a CPS proof of $p(\overline{x})$ from the assumptions $f_i(\overline{x}) \geq 0, -f_i(\overline{x}) \geq 0$, for $i \in [\ell]$ of size linear in $|C|$ (as before, we denote by $\overline{\mathcal{F}}$ the vector $f_1(\overline{x}),\ldots,f_\ell(\overline{x})$):

$$\begin{aligned} \sum_{i=1}^{\ell} f_i(\overline{x}) \cdot C_{i,P}(\overline{x},\overline{\mathcal{F}}) + \sum_{i=1}^{\ell} (-f_i(\overline{x})) \cdot C_{i,N}(\overline{x},\overline{\mathcal{F}}) & \\ = \sum_{i=1}^{\ell} f_i(\overline{x}) \cdot \Big(C_{i,P}(\overline{x},\overline{\mathcal{F}}) - C_{i,N}(\overline{x},\overline{\mathcal{F}})\Big) & \\ = \sum_{i=1}^{\ell} f_i(\overline{x}) \cdot C_i(\overline{x},\overline{\mathcal{F}}) = C(\overline{x},\overline{\mathcal{F}}) = p(\overline{x}). & \end{aligned}$$

□

5.1.2 CPS Simulates Positivstellensatz and SoS

We now turn to simulation results. Recall that the size of CPS (and similarly, IPS) is defined either as a constant-free circuit-size over $\mathbb{Q}$ or $\mathbb{Z}$, or as a circuit-size in the unit-cost model (Remark 5.3 item 3). The following theorem is immediate from the definitions.

Theorem 5.16. *CPS over an ordered ring R simulates Positivstellensatz (and hence also* SoS*) over R.*

Proof: This follows immediately from the fact that CPS is a circuit representation of the second big sum in eq. 1. More formally, let $\overline{\mathcal{F}} := \{f_i(\overline{x}) = 0\}_{i\in I}$ be a set of polynomial equations and let $\overline{\mathcal{H}} := \{h_j(\overline{x}) \geq 0\}_{j\in J}$ be a set of polynomial inequalities, where all polynomials are from $\mathbb{R}[x_1, \ldots, x_n]$. Consider the following Positivstellensatz refutation of $\overline{\mathcal{F}}$, $\overline{\mathcal{H}}$, where $\{p_i\}_{i\in I}$ and $\{s_{i,\zeta}\}_{i,\zeta}$ (for $i \in \mathbb{N}$ and $\zeta \subseteq J$) are collections of polynomials in $\mathbb{R}[x_1, \ldots, x_n]$:

$$\sum_{i\in I} p_i \cdot f_i + \sum_{\zeta\subseteq J} \left(\prod_{j\in\zeta} h_j \cdot \left(\sum_{i\in I_\zeta} s_{i,\zeta}^2 \right) \right) = -1\,. \tag{7}$$

The size of the Positivstellensatz refutation is the combined total number of monomials in $\{p_i\}_{i\in I}$ and $\sum_{i\in I_\zeta} s_{i,\zeta}^2$, for all $\zeta \subseteq J$ (see Definition 2.8).

By definition every $f_i(\overline{x}) = 0 \in \overline{\mathcal{F}}$ has corresponding two inequalities in $\overline{\mathcal{H}}$, $f_i(\overline{x}) \geq 0$ and $-f_i(\overline{x}) \geq 0$. Let the variables $\overline{y}$ (to be used as placeholder variables) be partitioned into three disjoint parts: $\overline{y} = \{y_i\}_{i\in I} \uplus \{y_{i'}\}_{i'\in I'} \uplus \{y_j\}_{j\in J}$, where $\{y_i\}_{i\in I}$ are the placeholder variables for $\{f_i(\overline{x}) \geq 0\}_{i\in I}$ in $\overline{\mathcal{H}}$, $\{y_{i'}\}_{i'\in I'}$ are the placeholder variables for $\{-f_i(\overline{x}) \geq 0\}_{i\in I}$ in $\overline{\mathcal{H}}$ and $\{y_j\}_{j\in J}$ are the placeholder variables for $\{h_j(\overline{x}) \geq 0\}_{j\in J}$ in $\overline{\mathcal{H}}$. Assume also that for every $i \in I$, $p_{i,P}$ is the sum of all non-negative monomials in p_i and $p_{i,N}$ is the sum of all negative monomials in p_i. Define

$$C(\overline{x}, \overline{y}) := \sum_{i\in I} p_{i,P} \cdot y_i + \sum_{i'\in I'} p_{i,N} \cdot y_i + \sum_{\zeta\subseteq J} \left(\prod_{j\in\zeta} y_j \cdot \left(\sum_i s_{i,\zeta}^2 \right) \right),$$

where each of the three big sums is written as a sum of monomials.

Hence, $C(\overline{x}, \overline{\mathcal{H}}) = -1$ by eq. 7 and the size of $C(\overline{x}, \overline{y})$ is linear in $\sum_{i\in I} |p_i|_{\#\text{monomials}} + \sum_{\zeta\subseteq J} \sum_{i\in I_\zeta} \left|s_{i,\zeta}^2\right|_{\#\text{monomials}}$. □

Corollary 5.17. *Boolean CPS simulates* SoS *and Positivstellensatz for inputs that include the Boolean axioms.*

5.1.3 CPS Simulates $\text{LS}^{\infty}_{*,+}$ for CNFs Written as Inequalities

CPS can simulate the strong semi-algebraic proof system as defined in Definition 2.12.

Theorem 5.18. *Boolean CPS simulates* $\text{LS}^{\infty}_{*,+}$ *(that is, "dynamic Positivstellensatz" from Definition 2.12).*

Recall that CPS uses the algebraic translation of CNFs (Definition 2.5) as equations while earlier semi-algebraic systems historically used the semi-algebraic translation of CNFs (Definition 2.11) as inequalities. We will show below that one can be efficiently converted into the other. Modulo this proposition the proof of Thm. 5.18 is almost trivial.

Proof sketch of Thm. 5.18. It suffices to observe that the derivation rules (adding and multiplying two inequalities, taking a square of an arbitrary polynomial) are the same as the rules of constructing the conic circuit. Therefore, following the $\text{LS}^{\infty}_{*,+}$ proof we construct a conic circuit that, given the axioms on the input, computes -1. □

Proposition 5.19 ([26], Lemmas 3.1 and 3.2)**.** *There is a polynomial-size propositional CPS proof that starts from the algebraic translation of a clause as the two inequalities* $\prod_{i\in P}(1-x_i)\cdot\prod_{j\in N}x_j \geq 0$ *and* $-\left(\prod_{i\in P}(1-x_i)\cdot\prod_{j\in N}x_j\right) \geq 0$*, and derives the semi-algebraic translation of the clause* $\sum_{i\in P}x_i+\sum_{j\in N}(1-x_j)-1\geq 0$*, and vice versa.*

Recall that CPS works with inequalities, whereas equalities $f=0$ in $\overline{\mathcal{F}}$ are interpreted as the two inequalities $f\geq 0$ and $-f\geq 0$ in $\overline{\mathcal{H}}$. Hence, Prop. 5.19 suffices to show that a clause given as an equality in $\overline{\mathcal{F}}$ can be translated efficiently in CPS to its semi-algebraic translation.

Proof of Prop. 5.19. Such efficient proofs were shown in [26, Lemmas 3.1 and 3.2] for a much weaker semi-algebraic proof system. The only difference of a proof in CPS is that we are allowed to write arbitrary formulas without multiplying out brackets. We provide a full proof in CPS in the direction that we use, for the sake of being self-contained.

We proceed by induction on the number of variables in the clause.

Base case: We start with one of the (algebraic) clauses x_1 or $1-x_1$. In the former case, we start from $-x_1$ which is in $\overline{\mathcal{H}}$ by the definition of CPS, and we need to derive $(1-x_1)-1$, which is equal to $-x_1$, hence we are done. In the latter case, we start from $-(1-x_1)$ which is x_1-1, hence we are done again.

Induction step:
Case 1: We start from the clause $(1-x_n)\cdot\prod_{i\in P}(1-x_i)\cdot\prod_{i\in N}x_i$ as a given equation (namely, in $\overline{\mathcal{F}}$; formally, the two corresponding inequalities are in $\overline{\mathcal{H}}$), and we need to derive $x_n+\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1$ in CPS. We consider the two cases $x_n=0$ and $x_n=1$, and then use reasoning by Boolean cases in CPS. Reasoning by Boolean cases in propositional CPS is doable in polynomial-size by Prop. A.5 which states this for IPS and since propositional CPS simulates IPS by Thm. 5.13 for the language of polynomial equations $\overline{\mathcal{F}}$ (in our case the initial clauses are indeed given as equations, and thus CPS simulates IPS' reasoning by Boolean cases).

In case $x_n=0$, $(1-x_n)\cdot\prod_{i\in P}(1-x_i)\cdot\prod_{i\in N}x_i=\prod_{i\in P}(1-x_i)\cdot\prod_{i\in N}x_i$, from which, by induction hypothesis we can derive in CPS with a polynomial-size proof $\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1$. Since $x_n=0$ we can add x_n to this expression obtaining $x_n+\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1$, and we are done.

In case $x_n=1$, we have $x_n+\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1=1+\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1=\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)$. But $\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)$ is easily provable in propositional CPS because we have the axioms $x_i\geq 0$ and $1-x_i\geq 0$ in $\overline{\mathcal{H}}$, for every variable x_i, by definition.
Case 2: We start from the clause $x_n\cdot\prod_{i\in P}(1-x_i)\cdot\prod_{i\in N}x_i$, and we need to derive $(1-x_n)+\sum_{i\in P}x_i+\sum_{i\in N}(1-x_i)-1$ in CPS. This is similar to Case 1 above with the two Boolean sub-cases $x_n=0$ and $x_n=1$ flipped. □

6 Reasoning about Bits within Algebraic Proofs

In what follows we define a number of circuits implementing arithmetic in the two's complement notation (see below for the details). Namely, we will define the following polynomial-size circuits:

$\mathrm{BIT}_i(f)$**:** if f is a circuit in the variables $\overline{x}$ then $\mathrm{BIT}_i(f)$ computes the ith bit of the integer computed by f (as a function of the input variables $\overline{x}$ where the variables $\overline{x}$ range over 0-1 values).

$\overline{\mathrm{BIT}}(f)$**:** a multi-output operation that computes the bit vector of f (as a function of the input variables $\overline{x}$ where the variables $\overline{x}$ range over 0-1 values).

$\overline{\text{ADD}}(\overline{y},\overline{z})$**:** a multi-output carry-lookahead circuit that computes the bit vector of the sum of $\overline{y}$ and $\overline{z}$.

$\text{ADD}_i(\overline{y},\overline{z})$**:** the circuit that computes the ith output bit in the carry-lookahead circuit $\overline{\text{ADD}}(\overline{y},\overline{z})$.

$\text{CARRY}_i(\overline{y},\overline{z})$**:** the carry for bit i when adding two bit vectors $\overline{y},\overline{z}$.

$\overline{\text{PROD}}(\overline{y},\overline{z})$**:** the multi-output circuit computing binary multiplication of two bit vectors $\overline{y}$ and $\overline{z}$.

$\overline{\text{PROD}}_+(\overline{y},\overline{z})$**:** the multi-output circuit computing binary multiplication of two nonnegative bit vectors $\overline{y}$ and $\overline{z}$.

$\text{VAL}(\overline{z})$**:** the valuation function that converts $\overline{z}$ encoding an integer in the two's complement representation to its integer value (see below).

$\overline{\text{ABS}}(\overline{x})$**:** The multi-output circuit computing the two's complement binary representation of the absolute value of an input integer $\overline{x}$ given in two's complement.

We construct the BIT_i function by induction on the size of f. In general this cannot be done for algebraic variables, but in our case we are assuming that the variables $x_1,\ldots,x_n$ are Boolean variables, and this allows us to carry out the constructions below, yielding a circuit of size which is polynomial in the size of the algebraic circuit of f where ring scalars are encoded in binary.

6.1 Basic Two's Complement Arithmetic

Two's Complement is a standard way of representing integers in computers, in particular, it allows to treat positive and negative integers in exactly the same way when, for example, adding them. Buss [10] and Goerdt [20] considered binary arithmetic as carried out in Frege proof system (see also the textbook by Lu [40]). However, Buss and Goerdt work used the more restrictive computational model of (Boolean) formulas, while we work with (algebraic) circuits. Moreover, we shall work with non-Boolean polynomials like VAL in Lemma 6.9 (that is, polynomials that can evaluate to non Boolean values, and these polynomials have no direct translation in Frege). Therefore, we prove our construction from scratch in IPS.

In the two's complement scheme the value represented by the bit string $\overline{w}\in\{0,1\}^k$ is determined by a function from $\{0,1\}^k$ to $\mathbb{Z}$ as follows:

Definition 6.1 (the binary value operation VAL)**.** *Given a bit vector $w_0\cdots w_{k-1}$ of variables, denoted $\overline{w}$, ranging over* 0-1 *values, define the following algebraic circuit with integer coefficients*[9]*:*

$$\text{VAL}(\overline{w}) := \sum_{i=0}^{k-2} 2^i\cdot w_i \; - 2^{k-1}\cdot w_{k-1}.$$

The most significant bit (msb) w_{k-1} is called the ***sign bit*** *of $\overline{w}$.*

Thus, $\text{VAL}(\overline{w})=\sum_{i=0}^{k-2}2^i\cdot w_i$ in case the sign bit $w_{k-1}=0$ (hence, $\overline{w}$ encodes a positive integer), and $\text{VAL}(\overline{w})=\sum_{i=0}^{k-2}2^i\cdot w_i \; - 2^{k-1}$, in case the sign bit $w_{k-1}=1$ (hence, $\overline{w}$ encodes a negative integer).

We will represent the integers computed inside the original algebraic circuit by *variable length* bit vectors (that is, different bit vectors may have different lengths). For each gate in a given circuit

[9] We assumed that algebraic circuits have fan-in two, hence VAL is written as a logarithmic depth circuit of addition gates (and product gates at the bottom of the circuit).

we will assign a number that is sufficiently large to store the bit vector of the integer it computes when the input variables range over 0-1 values; this number will be called the ***syntactic length*** of the gate (or equivalently, of the circuit whose output gate is this gate). The syntactic length of a gate is not necessarily the minimal number of bits needed to store a number, since we will find it convenient to use slightly more bits than is actually required at times. For instance, the product of two t-bit binary numbers can be stored with only $2t-1$ bits, but we will use $2t+3$ bits for a product. Given the syntactic length of algebraic gates such as $+, \times$ as functions of the syntactic length of their incoming edges, we can compute by induction on circuit size the syntactic length of any given gate in a circuit. It will be straightforward that the syntactic length of a constant-free (integer algebraic) circuit that has s gates and multiplicative depth D (that is, the longest directed path goes through at most D multiplications) is at most $O(s2^D)$ (imagine repeated squaring of 2 as the worst case), and it is at most $O(sd)$ for a constant-free circuit that has syntactic degree d (that is, it would compute a polynomial of total degree at most d if all constants -1 are replaced by 1; surely, $d \leq 2^D$).

When we need to make an operation over integers of different bit length, *we pad the shorter one* (in the two's complement scheme, a number is always padded by its sign bit, and it is immediate to see that such padding preserves the value of the number as computed by VAL).

We will use the Boolean connectives $\wedge, \vee, \oplus$, which stand for AND, OR and XOR, respectively. In order to use Boolean connectives inside algebraic circuits, we define the arithmetization of connectives as follows:

Definition 6.2 (arithmetization operation $\mathrm{arit}(\cdot)$)**.** *For a variable y_i, $\mathrm{arit}(y_i) := y_i$. For the truth values false $\bot$ and true $\top$ we put $\mathrm{arit}(\bot) := 0$ and $\mathrm{arit}(\top) := 1$. For logical connectives we define $\mathrm{arit}(A \wedge B) := \mathrm{arit}(A) \cdot \mathrm{arit}(B)$, $\mathrm{arit}(A \vee B) := 1 - (1 - \mathrm{arit}(A)) \cdot (1 - \mathrm{arit}(B))$, and for the* XOR *operation we define $\mathrm{arit}(A \oplus B) := \mathrm{arit}(A) + \mathrm{arit}(B) - 2 \cdot \mathrm{arit}(A) \cdot \mathrm{arit}(B)$.*

In this way, for every Boolean circuit $F(\overline{x})$ with n variables and a Boolean assignment $\overline{\alpha} \in \{0,1\}^n$, $\mathrm{arit}(F)(\overline{\alpha}) = 1$ iff $F(\overline{\alpha}) = \mathsf{true}$.

In what follows, we sometimes omit $\mathrm{arit}(\cdot)$ from our formulas and simply write $\wedge$, $\vee$, $\oplus$ meaning the corresponding polynomials or algebraic circuits.

Addition is done with a carry lookahead adder as follows:

Definition 6.3 (CARRY_i, ADD_i, $\overline{\mathrm{ADD}}$)**.** *When we use an adder for vectors of different size, we pad the extra bits of the shorter one by its sign bit. Suppose that we have a pair of length-t vectors of variables $\overline{y} = (y_0, \ldots, y_{t-1}), \overline{z} = (z_0, \ldots, z_{t-1})$ of the same size. We first pad the two vectors by a single additional bit $y_k = y_{k-1}$ and $z_t = z_{t-1}$, respectively (this is the way to deal with a possible overflow occurring while adding the two vectors). Define*

$$\mathrm{CARRY}_i(\overline{y}, \overline{z}) \;:=\; \begin{cases} (y_{i-1} \wedge z_{i-1}) \vee ((y_{i-1} \vee z_{i-1}) \wedge \mathrm{CARRY}_{i-1}(\overline{y}, \overline{z})), & i = 1, \ldots, t; \\ 0\,, & i = 0\,, \end{cases} \tag{8}$$

and

$$\mathrm{ADD}_i(\overline{y}, \overline{z}) := y_i \oplus z_i \oplus \mathrm{CARRY}_i(\overline{y}, \overline{z})\,,\ i = 0, \ldots, t.$$

Finally, define

$$\overline{\mathrm{ADD}}(\overline{y}, \overline{z}) := \mathrm{ADD}_t(\overline{y}, \overline{z}) \cdots \mathrm{ADD}_0(\overline{y}, \overline{z}) \tag{9}$$

(that is, $\overline{\mathrm{ADD}}$ is a multi-output circuit with $t+1$ output bits).

It is worth noting that by Definition 6.3 we have (where the equality means that the polynomials are identical, though the circuits for them is different):

$$\mathrm{CARRY}_i(\overline{y},\overline{z}) = \begin{cases} \bigvee_{r<i}\left(y_r \wedge z_r \wedge \bigwedge_{k=r+1}^{i-1}(y_k \vee z_k)\right), & i = 1,\ldots,t; \\ 0, & i = 0\,. \end{cases} \tag{10}$$

Let s be a bit, and denote by $\mathbf{e}(s)$ the bit vector in which all bits are s (that is, $\mathbf{e}(s) = s\cdots s$) and where the length of the vector is understood from the context. In the two's complement scheme inverting a positive number is a two-step process: first flip its bits (that is, XOR with the all-1 vector) and then add 1 to the result. Hence, in what follows, to *invert a negative* number, and extract its absolute value, we first subtract 1 and then flip its bits:

Definition 6.4 (absolute value operation $\overline{\mathrm{ABS}}$)**.** *Let $\overline{x}$ be a t-bit vector representing an integer in two's complement. Let s be its sign bit, and let $\overline{m} = \mathbf{e}(s)$ be the t-bit vector all of which bits are s. Define $\overline{\mathrm{ABS}}(\overline{x})$ as the multi-output circuit that outputs $t+1$ bits*[10] *as follows (where $\oplus$ here is bitwise* XOR*):*

$$\overline{\mathrm{ABS}}(\overline{x}) := \overline{\mathrm{ADD}}(\overline{x},\overline{m}) \oplus \overline{m}.$$

For the sake of the clarity of the proof, we compute the product of two t-bit numbers in the two's complement notation somewhat less efficiently than it is usually done: we define the product of nonnegative numbers in the standard way, apply it to the absolute values of the numbers and then apply the appropriate sign bit. This way we get a slightly greater number of bits than needed to keep the value.

To define the multiplication of two t-bit integers in the two's complement notation we first define an unsigned multiplication operator $\overline{\mathrm{PROD}}_+$ which is easy to implement. It takes two non-negative integers (that is, their sign bit is zero, and this assumption is required for the circuit to work correctly), and performs the standard non-negative multiplication by performing $i = 0,\ldots,t-1$ iterations, where the ith iteration consists of multiplying the first integer by the single i-th bit of the second integer, and then padding this product by i zeros to the right.

Definition 6.5 (product of two nonnegative numbers in binary $\overline{\mathrm{PROD}}_+$)**.** *Let $\overline{a}$ and $\overline{b}$ be two t-bit integers where the sign bit of both $\overline{a},\overline{b}$ is zero. We define t iterations $i = 0,\ldots,t-1$; the result of the ith iteration is defined as the $(t+i)$-length vector $\overline{s}_i = s_{i,t+i-1}s_{i,t+i-2}\cdots s_{i,0}$ where*

$$\begin{aligned} s_{ij} &:= a_{j-i} \wedge b_i, && \text{for } i \le j \le t-1+i, \\ s_{ij} &:= 0 && \text{for } 0 \le j < i. \end{aligned}$$

(Note that we use the sign bits a_{t-1}, b_{t-1} in this process although we assume it is zero; this is done in order to preserve uniformity with other parts of the construction.) The product of the two nonnegative t-bit numbers is defined as the sequential addition of all the results in all iterations:

$$\overline{\mathrm{PROD}}_+(\overline{a},\overline{b}) := \overline{\mathrm{ADD}}\left(\overline{s}_{t-1}, \overline{\mathrm{ADD}}\left(\overline{s}_{t-2},\ldots,\overline{\mathrm{ADD}}\left(\overline{s}_1,\overline{s}_0\right)\right)\ldots\right).$$

The number of output bits of $\overline{\mathrm{PROD}}_+$ is formally $2t$ including the sign bit.

[10] Note that since the largest (in absolute value) negative number that can be represented by a t-bit binary vector in the two's complement scheme is 2^{t-1}, while the largest positive number that can be represented in such a way is only $2^{t-1}-1$, we need to store the absolute number of a t-bit integer in the two's complement scheme using $t+1$ bits.

Definition 6.6 (product of two numbers in binary $\overline{\text{PROD}}$)**.** *Let $\overline{y}$ and $\overline{z}$ be two t-bit integers in the two's complement notation. Define the product of $\overline{y}$ and $\overline{z}$ by first multiplying the absolute values of the two numbers and then applying the corresponding sign bit:*

$$\overline{\text{PROD}}(\overline{y},\overline{z}) := \overline{\text{ADD}}\left(\overline{\text{PROD}}_{+}\left(\overline{\text{ABS}}(\overline{y}),\overline{\text{ABS}}(\overline{z})\right)\oplus\overline{m},s\right),$$

where $s = y_{t-1}\oplus z_{t-1}$ and $\overline{m} = \mathbf{e}(s)$, with y_{t-1}, z_{t-1} the sign bits of $\overline{y},\overline{z}$ as bit vectors in the two's complement notation, respectively.

Note that the number of bits that $\overline{\text{PROD}}$ outputs is $2t+3$: given a t-bit number, its ABS *is of size $t+1$ (including the zero sign bit), the nonnegative product $\overline{\text{PROD}}_{+}$ of $\text{ABS}(\overline{x})$ and $\text{ABS}(\overline{y})$ has size $2(t+1)$, bitwise XOR does not change the length, and adding s augments the result by one more bit.*

Note that given the bit vectors $\overline{x},\overline{y}$ of length t, the size of the circuit for $\text{CARRY}_i(\overline{x},\overline{y})$ is $O(t)$ by eq. 8, for $\text{ADD}_i(\overline{x},\overline{y})$ it is $O(t)$ as well, and for $\overline{\text{ADD}}(\overline{x},\overline{y})$ it is still $O(t)$ because in eq. 9 we can *re-use* CARRY_i. The size of $\overline{\text{ABS}}(\overline{x})$ is also $O(t)$ (this is addition and linear-size bitwise XOR) and finally $\overline{\text{PROD}}(\overline{x},\overline{y})$ is of size $O(t^2)$: we perform an addition of $O(t)$ bit vectors of size $O(t)$ each.

6.2 Extracting Bits and the Main Binary Value Lemma

We are now ready to define the algebraic circuit $\overline{\text{BIT}}$, in which BIT_i is the ith bit, that extracts the bit vector of the output of a given algebraic circuit (as a function of the input variables, where the variables are considered to range over 0-1).

Definition 6.7 (the bit vector extraction operation $\overline{\text{BIT}}$)**.** *Let F be an algebraic circuit with t its syntactic length. Assume that $0 \le i \le t-1$. We define $\text{BIT}_i(F)$ to be the circuit computing the ith bit of F recursively as follows (note that BIT_i is a circuit, that is, in the induction step of the construction we may* re-use *the same nodes more than once).*
***Case 1**: $F = x_j$ for an input x_j. Then, $\text{BIT}_0(F) := x_j$, $\text{BIT}_1(F) := 0$ (in this case there are just two bits).*
***Case 2**: $F = \alpha$, for $\alpha \in \mathbb{Z}$. Then, BIT_i is defined to be the ith bit of α in two's complement notation, with at most t bits (i.e., $i < t$).*
***Case 3**: $F = G + H$. Then $\overline{\text{BIT}}(F) := \overline{\text{ADD}}(\overline{\text{BIT}}(G),\overline{\text{BIT}}(H))$, and $\text{BIT}_i(F)$ is defined to be the ith bit of $\overline{\text{BIT}}(F)$.*
***Case 4**: $F = G \cdot H$. Then $\overline{\text{BIT}}(F) := \overline{\text{PROD}}(\overline{\text{BIT}}(G),\overline{\text{BIT}}(H))$, and $\text{BIT}_i(F)$ is defined to be the ith bit of $\overline{\text{BIT}}(F)$.*

Recall that in the latter two cases the shorter number is padded to match the length of the longer number by copying the sign bit before applying $\overline{\text{ADD}}$ or $\overline{\text{PROD}}$.

Note that both $|\text{BIT}_i(F)|$ and $|\overline{\text{BIT}}(F)|$ have size $O(t^2\cdot|F|)$ (for t the syntactic length of F). To understand this upper bound, observe that every node in the circuit for $\overline{\text{BIT}}(F)$ belongs to either a sub-circuit computing the ith bit of $\overline{\text{ADD}}(\overline{x},\overline{y})$ (i.e., $\text{ADD}_i(\overline{x},\overline{y})$) or to a sub-circuit computing the ith bit of $\overline{\text{PROD}}(\overline{x},\overline{y})$, for some $0 \le i \le t$ and some two vectors of bits $\overline{x},\overline{y}$ that were already computed by the circuit (since this is a circuit, once the vectors $\overline{x},\overline{y}$ were computed we can use their result as many times as we like, without the need to compute them again). Hence, each addition gate in F contributes $O(t)$ nodes to $\overline{\text{BIT}}(F)$ and each product gate in F contributes $O(t^2)$ nodes to $\overline{\text{BIT}}(F)$. This accounts for the size $O(t^2\cdot|F|)$ for $\overline{\text{BIT}}(F)$ (as well as for $\text{BIT}_i(F)$).

For technical reasons we need the following definition:

Definition 6.8 (IPS sub-proof; multi-output IPS proofs)**.** *Let $C(\overline{x},\overline{y})$ be an IPS proof from a set of polynomial equations as assumptions $\overline{\mathcal{F}}$ of $p(\overline{x})$ (that is, $C(\overline{x},\overline{\mathcal{F}}) = p(\overline{x})$ and $C(\overline{x},\overline{0}) = 0$), and*

suppose that $C'(\overline{x},\overline{y})$ is a sub-circuit of $C(\overline{x},\overline{y})$ such that $C'(\overline{x},\overline{y})$ is an IPS proof of $g(\overline{x})$ (that is, $C'(\overline{x},\overline{\mathcal{F}}) = g(\overline{x})$ and $C'(\overline{x},\overline{0}) = 0$).[11] *Then, we say that $C'(\overline{x},\overline{y})$* is a sub-proof of $C(\overline{x},\overline{y})$*, and also (by slight abuse of terminology) that $g(\overline{x})$* is a sub-proof of the IPS proof C of $p(\overline{x})$ from $\overline{\mathcal{F}}$*.*

Furthermore, a multi-output *circuit $C(\overline{x},\overline{y})$ is said to be an IPS proof from assumptions $\overline{\mathcal{F}}$, if each of its output gates computes an IPS proof.*

For example, let the assumptions be $\overline{\mathcal{F}} = \{x_2, (1+x_1x_2)\}$. The two-output circuit $C(\overline{x},\overline{x})$ defined as $(x_1 \cdot y_1, x_1 \cdot y_2)$, where x_1 is joined by the two sub-circuits $x_1 \cdot y_1$ and $x_1 \cdot y_2$, is an IPS proof having two sub-proofs: the first is a sub-proof of $x_1 \cdot x_2$ from $\overline{\mathcal{F}}$, and the second is a sub-proof of $x_1 \cdot (1+x_1x_2)$ from $\overline{\mathcal{F}}$.

Lemma 6.9. *(main binary value lemma) Let $F(\overline{x})$ be an algebraic circuit over $\mathbb{Z}$ in the variables $\overline{x} = \{x_1, \ldots, x_n\}$, and suppose that the syntactic length of F is at most t. Then, there is an IPS proof of*

$$F = \mathrm{VAL}(\overline{\mathrm{BIT}}(F))$$

of size $\mathrm{poly}(|F|, t)$ *(there are no axioms for this IPS proof, except for the Boolean axioms). Furthermore, if $F(\overline{x})$ is constant-free, the* $\mathrm{poly}(|F|, t)$*-size IPS proof is also constant-free.*

Proof: We will proceed, essentially, by induction on the structure of F. For technical reasons (since we work with circuits of which sub-circuits can be re-used) we are going to state our induction hypothesis on an IPS proof that consists, as sub-proofs, of other IPS proofs.

More precisely, let $F_1, \ldots, F_k$ be a set of sub-circuits of F. We denote by $\lambda(F_1, \ldots, F_k)$ the size of the IPS proof we are to construct; this proof will consist (as sub-proofs) of IPS proofs of $F_i = \mathrm{VAL}(\overline{\mathrm{BIT}}(F_i))$, for all $i \in [k]$. We let $\lambda(\emptyset) := 0$. We shall assume that at every step of the construction F_1 is of maximal size, namely there is no F_i that has size bigger than F_1 (possibly there are other F_i's with the same size). In this way, we make sure that F_1 is not a sub-circuit of any other F_i. The IPS proof is to be constructed by induction on $|F_1|$ so that in each step of the induction we deal with a single sub-circuit F_1 of F, such that $|F_1| > 1$. In the base case of the induction we thus have $\lambda(F_1, \ldots, F_k)$ such that all F_i's have size 1, namely, they are all the variables and constant gates that appear in F.

Note that since we treat the input to λ as a *set* we discard duplicate F_i's from its input. For example, $\lambda(G, H) = \lambda(G)$ in case $G = H$. (This is convenient in what follows, because F is a circuit and the IPS proof we construct is also a circuit, and hence can re-use multiple times the same IPS sub-proof; see below.)

We proceed by induction on $|F_1|$, the maximal size of a circuit in $F_1, \ldots, F_k$, to show the following: in case all $F_1, \ldots, F_k$ are variables or constant nodes we show that

$$\lambda(F_1, \ldots, F_k) \leq c_0 kt,$$

for some constant c_0.

In case $F_1 = G \circ H$, for $\circ \in \{+, \cdot\}$, we show that

$$\lambda(F_1, \ldots, F_k) \leq \lambda(G, H, F_2, \ldots, F_k) + (t \cdot |F_1|)^b,$$

for some constants b independent of $|F_1|$ and t. This recurrence relation immediately implies that $\lambda(F) \leq |F| \cdot (t \cdot |F|)^b$, which is polynomial in $|F|$ and t (informally, every node in F contributes the additive term $c_0 t$ or $(t \cdot |F|)^b$ to the recurrence).

[11] Notice that not all sub-circuits of C are IPS proofs; e.g., if they are polynomials that are not in the ideal generated by $\overline{y}$, they are not sub-proofs.

Base case: All $F_1, \ldots, F_k$ are variables or constant nodes. We construct a multi-output IPS proof $C(\overline{x}, \overline{y})$, that consists of k disjoint proofs of $\mathrm{VAL}(\overline{\mathrm{BIT}}(F_j)) = F_j$, for $j \in [k]$.

Case 1: $F_j = x_i$, for $i \in [n]$. Thus, the syntactic length of F_j is 2 and by definition $\mathrm{VAL}(\overline{\mathrm{BIT}}(x_i)) := \mathrm{VAL}(0x_i) := x_i - 2^1 \cdot 0$ (the left equality is by definition of $\overline{\mathrm{BIT}}$, and the right equality is by definition of VAL). Hence, $\mathrm{VAL}(\overline{\mathrm{BIT}}(x_i)) = x_i$ is a true polynomial identity and so by Fact A.1 we have an IPS proof of size precisely the size of the circuit for $x_i - 2^1 \cdot 0 - x_i$ which is at most, say, 20.

Case 2: $F_j = \alpha$, for $\alpha \in \mathbb{Z}$. Then, by Definition 6.7, $\mathrm{VAL}(\overline{\mathrm{BIT}}(\alpha)) := \sum_{i=0}^{t-2} 2^i \alpha_i - 2^{t-1} \cdot \alpha_{t-1}$, where $\alpha_{t-1} \ldots \alpha_0$ is the correct bit vector of α in the two's complement notation (where t is the syntactic length of F_j). Hence, $\mathrm{VAL}(\overline{\mathrm{BIT}}(\alpha)) = \alpha$ is a true polynomial identity of size at most $c_0 t$, for some constant c_0. By Fact A.1 we have an IPS proof of $\mathrm{VAL}(\overline{\mathrm{BIT}}(\alpha)) = \alpha$ of size at most $c_0 t$.

Hence, the total size of all the proofs of $\mathrm{VAL}(\overline{\mathrm{BIT}}(F_j)) = F_j$, for $j \in [k]$, amounts to $\lambda(F_1, \ldots, F_k) \leq c_0 k t$, as required.

Induction step: We assume that the syntactic length of F_1 is t. We show that, in case $F_1 = G + H$, $\lambda(F_1, \ldots, F_k) \leq \lambda(G, H, F_2, \ldots, F_k) + c_1 + (t \cdot |F_1|)^{b'}$, for some constants b' and c_1, and in case $F_1 = G \cdot H$ we show that $\lambda(F_1, \ldots, F_k) \leq \lambda(G, H, F_2, \ldots, F_k) + (t \cdot |F_1|)^a + (t \cdot |F_1|)^{b'}$, for some constants b' and a independent of t and $|F_1|$. Thus, choosing a big enough constant b, e.g., $b > 10 \cdot \max(a, b')$, will conclude that $\lambda(F) \leq |F| \cdot (t \cdot |F|)^b$, and hence will conclude the proof.

Case 1: $F_1 = G + H$, with F_1 of syntactic length t. Since the syntactic length of F_1 is t, the syntactic length of $\overline{\mathrm{BIT}}(G), \overline{\mathrm{BIT}}(H)$ is $t - 1$ (after padding $\overline{\mathrm{BIT}}(G), \overline{\mathrm{BIT}}(H)$ to have the same size). We need to construct an IPS proof consisting of sub-proofs of $\mathrm{VAL}(\overline{\mathrm{BIT}}(F_1)) = F_1, \ldots, \mathrm{VAL}(\overline{\mathrm{BIT}}(F_k)) = F_k$. By induction hypothesis we have an IPS proof consisting of sub-proofs of $G + H = \mathrm{VAL}(\overline{\mathrm{BIT}}(G)) + \mathrm{VAL}(\overline{\mathrm{BIT}}(H))$ and $F_i = \mathrm{VAL}(\overline{\mathrm{BIT}}(F_i))$, for $i = 2, \ldots, k$, of total size $\lambda(G, H, F_2, \ldots, F_k) + c_1$, for some constant c_1 (the constant c_1 here is needed for the addition of the two proofs; see Fact A.3 in which $c_1 = 1$). It thus suffices to prove

$$\mathrm{VAL}(\overline{\mathrm{BIT}}(G)) + \mathrm{VAL}(\overline{\mathrm{BIT}}(H)) = \mathrm{VAL}(\overline{\mathrm{BIT}}(F_1))$$

with an IPS proof of size at most $(t \cdot |F_1|)^{b'}$, for some constant b' independent of t.

For simplicity of notation, let us denote the circuits for bits $\overline{\mathrm{BIT}}(G)$ and $\overline{\mathrm{BIT}}(H)$, by $\overline{y}$ and $\overline{z}$, respectively, and the syntactic length of $\overline{y}, \overline{z}$ by $r = t - 1$. We proceed slightly informally within IPS as follows (recall that polynomial identities of size s have trivial IPS proofs of size s by Fact A.1).

$$\mathrm{VAL}(\overline{y}) + \mathrm{VAL}(\overline{z}) = \sum_{i=0}^{r-2} 2^i (y_i + z_i) - 2^{r-1}(y_{r-1} + z_{r-1}).$$

On the other hand we have (recall the padded bits $y_r = y_{r-1}$, $z_r = z_{r-1}$ in the definition of $\overline{\mathrm{ADD}}$)

$$\begin{aligned}
\mathrm{VAL}(\overline{\mathrm{BIT}}(F_1)) &= \mathrm{VAL}\left(\mathrm{ADD}_0\left(\overline{y}, \overline{z}\right) \ldots \mathrm{ADD}_r\left(\overline{y}, \overline{z}\right)\right) \qquad \text{(by definition of } \overline{\mathrm{BIT}}\text{)} \\
&= \sum_{i=0}^{r-1} 2^i \left(z_i \oplus y_i \oplus \mathrm{CARRY}_i(\overline{y}, \overline{z})\right) - 2^r (z_{r-1} \oplus y_{r-1} \oplus \mathrm{CARRY}_r(\overline{y}, \overline{z})) \\
&\qquad\qquad \text{(by definition of } \mathrm{ADD}_i \text{ and VAL).}
\end{aligned}$$

Thus, to complete the case of addition, it remains to prove the following:

Claim 6.10. *There is an IPS proof with size at most $(r \cdot |F_1|)^{b''}$, for a constant b'' (independent of r, and such that b' will be chosen so that $b' > b''$) of the equation*

$$\sum_{i=0}^{r-2} 2^i(y_i + z_i) - 2^{r-1}(y_{r-1} + z_{r-1})$$
$$= \sum_{i=0}^{r-1} 2^i \left(z_i \oplus y_i \oplus \mathrm{CARRY}_i(\overline{y}, \overline{z})\right) - 2^r(z_{r-1} \oplus y_{r-1} \oplus \mathrm{CARRY}_r(\overline{y}, \overline{z}))\,.$$

Proof of claim: This is proved by induction on r as follows.

Base case: $r = 2$. It is easy to see (or can be verified by hand) that in this case the two sides of the claim are equal for every $y_0, z_0, y_1, z_1 \in \{0, 1\}$. Given that the number of bits in this case is constant, this is enough to conclude that there is an IPS proof of the above equation (using reasoning by Boolean cases as in Prop. A.5, over a constant number of possible truth assignments for y_0, z_0, y_1, z_1) of size $(2 \cdot |F|)^{b''}$, for some constant b''.

Induction step: To prove this step, notice that using the induction hypothesis we see that the equality we need to prove is

$$(z_{r-2} + y_{r-2}) - (z_{r-1} + y_{r-1}) = z_{r-2} \oplus y_{r-2} \oplus \mathrm{CARRY}_{r-1}(\overline{y}, \overline{z}) - z_{r-1} \oplus y_{r-1} \oplus \mathrm{CARRY}_r(\overline{y}, \overline{z})\,.$$

Substituting the definition for CARRY_{r-1} and CARRY_r, we get a polynomial equation in five variables: z_{r-2}, y_{r-2}, z_{r-1}, y_{r-1}, and C, where $C = \mathrm{CARRY}_{r-2}(\overline{y}, \overline{z})$. Once it is verified by hand on $\{0, 1\}$, we conclude that the circuit size of the proof is polynomial in the size of the circuits provided that these five "variables" are indeed Boolean. Four of them are Boolean by the hypothesis of the lemma, and the equation $C^2 - C = 0$ for the carry bit C is also easy to derive. Similarly to the above, we get an IPS proof of size at most $(r \cdot |F|)^{b''}$, for a constant b'. ■Claim

This concludes Case 1 (i.e., addition) of the induction step of the proof of Lemma 6.9.

Case 2: $F_1 = G \cdot H$, with F_1 of syntactic length t. We need to construct an IPS proof consisting of sub-proofs of $\mathrm{VAL}(\overline{\mathrm{BIT}}(F_1)) = F_1, \ldots, \mathrm{VAL}(\overline{\mathrm{BIT}}(F_k)) = F_k$, of size at most $\lambda(G, H, F_2, \ldots, F_k) + (t \cdot |F_1|)^a + (t \cdot |F_1|)^{b'}$, for constants a, b' independent of $|F_1|$ and t. By induction hypothesis we have an IPS proof consisting of sub-proofs of $G \cdot H = \mathrm{VAL}(\overline{\mathrm{BIT}}(G)) \cdot \mathrm{VAL}(\overline{\mathrm{BIT}}(H))$ and $F_i = \mathrm{VAL}(\overline{\mathrm{BIT}}(F_i))$, for $i = 2, \ldots, k$, of total size $\lambda(G, H, F_2, \ldots, F_k) + |F_1| + c_2$, for some constant c_2 (the term $|F_1| + c_2$ here is needed for the product of the two proofs $G = \mathrm{VAL}(\overline{\mathrm{BIT}}(G))$ and $H = \mathrm{VAL}(\overline{\mathrm{BIT}}(H))$; see Fact A.4). It thus suffices to prove

$$\mathrm{VAL}(\overline{\mathrm{BIT}}(F_1)) = \mathrm{VAL}(\overline{\mathrm{BIT}}(G)) \cdot \mathrm{VAL}(\overline{\mathrm{BIT}}(H))$$

with an IPS proof of size at most $(t \cdot |F_1|)^{b'}$, for a constant b'. Let r denote the syntactic length of G, H. Since the syntactic length of F_1 is t we have $t = 2r + 3$.

In what follows, we use the notation from Definition 6.6, namely, $\overline{y} = \overline{\mathrm{BIT}}(G)$ and $\overline{z} = \overline{\mathrm{BIT}}(H)$. We first prove two simple statements about $\overline{\mathrm{ABS}}$.

Claim 6.11. *Let $\overline{x}$ be a bit vector of length r representing an integer in two's complement and let s be the sign bit of $\overline{x}$. Then $\mathrm{VAL}(\overline{x}) = (1 - 2s) \cdot \mathrm{VAL}\left(\overline{\mathrm{ABS}}(\overline{x})\right)$ has an IPS proof from the Boolean axioms, of size at most r^c, for some constant c independent of r.*

Proof of claim: Recall that the size of $\overline{\text{ABS}}(\overline{x})$ is $O(r)$. We will apply (slightly informally) Prop. A.5 for reasoning by Boolean cases in IPS as follows. Consider the two cases for the sign bit s. In case $s = 0$ the claim is not hard to check; we will show only the case $s = 1$.

Recall that inverting a negative number via $\overline{\text{ABS}}$ is done by subtracting 1 (which is the same as adding the all-one vector) and then inverting all the bits in the resulting vector. Let $\overline{y}$ be a bit vector and $\mathbf{1}$ be the all-one vector of the same length of $\overline{y}$, then

$$\text{VAL}(\overline{y} \oplus \mathbf{1}) = \sum_{i=0}^{r-2}(1 - y_i)2^i - (1 - y_{r-1})2^{r-1} = -1 - \text{VAL}(\overline{y}). \tag{11}$$

Using this, we have

$$\begin{aligned}(1 - 2s) \cdot \text{VAL}(\overline{\text{ABS}}(\overline{x})) &= -1 \cdot \text{VAL}(\overline{\text{ADD}}(\overline{x}, \mathbf{1}) \oplus \mathbf{1}) \quad \text{(by definition of } \overline{\text{ABS}}) \\ &= -1 \cdot (-1 - \text{VAL}(\overline{\text{ADD}}(\overline{x}, \mathbf{1}))) \quad \text{(by eq. 11 above)} \\ &= 1 + \text{VAL}(\overline{\text{ADD}}(\overline{x}, \mathbf{1})).\end{aligned}$$

By the addition case (Case 1 above) we can construct an IPS proof of $\text{VAL}\left(\overline{\text{ADD}}(\overline{x}, \mathbf{1})\right) = \text{VAL}(\overline{x}) + \text{VAL}(\mathbf{1}) = \text{VAL}(\overline{x}) - 1$ of size at most $r^{b'}$, for some constant b'. This concludes the proof since we finally get $1 + \text{VAL}(\overline{\text{ADD}}(\overline{x}, \mathbf{1})) = \text{VAL}(\overline{x})$, where the whole proof is of size at most r^c, for some constant c. $\blacksquare_{\text{Claim}}$

Claim 6.12 (non-negativeness of $\overline{\text{ABS}}$)**.** *Let $\overline{x}$ be a bit vector of length r representing an integer in two's complement and let s be the circuit computing the sign bit of $\overline{\text{ABS}}(\overline{x})$ according to Definition 6.4. Then $s = 0$ has a polynomial-size* IPS proof (using only the Boolean axioms).

Proof of claim: We proceed as before by considering the two cases of the sign of $\overline{x}$. The case of positive sign is easy to verify. In the case of a negative sign we have $\text{ABS}(\overline{x}) = \overline{\text{ADD}}(\overline{x}, \mathbf{1}) \oplus \mathbf{1}$, where by the definition of $\overline{\text{ADD}}$, $\overline{x}$ is padded with an additional one bit $x_r = x_{r-1} = 1$, and hence the sign bit of $\text{ABS}(\overline{x})$ is computed as $\text{CARRY}_r(\overline{x}, \mathbf{1}) \oplus 1$ (note that $\overline{\text{ADD}}$ has one more bit than $\overline{x}$). By (eq. 10), $\text{CARRY}_r(\overline{x}, \mathbf{1})$ is equal to (the arithmetization of) $\bigvee_{i<r} x_i$. Since $x_{r-1} = 1$, we can prove in IPS by a simple substitution that the arithmetization of $\bigvee_{i<r} x_i$ is the constant 1, leading to $\text{CARRY}_r(\overline{x}, \mathbf{1}) \oplus 1 = 0$. $\blacksquare_{\text{Claim}}$

We consider then the case of the multiplication of nonnegative numbers.

Claim 6.13. *Let $\overline{y}, \overline{z}$ be two bit vectors of length r in the two's complement notation. Then,*

$$\text{VAL}\left(\overline{\text{PROD}}_+\left(\overline{\text{ABS}}(\overline{y}), \overline{\text{ABS}}(\overline{z})\right)\right) = \text{VAL}\left(\overline{\text{ABS}}(\overline{y})\right) \cdot \text{VAL}\left(\overline{\text{ABS}}(\overline{z})\right)$$

has an IPS derivation (from the Boolean axioms) of size r^c, for a constant c independent of r.

Proof of claim: Let $\overline{y}^+$ denote $\overline{\text{ABS}}(\overline{y})$ and $\overline{z}^+$ denote $\overline{\text{ABS}}(\overline{z})$, both of length $r + 1$ (we know from Claim 6.12 that the sign bits $\overline{y}^+_r, \overline{z}^+_r$ of $\overline{y}^+$ and $\overline{z}^+$, respectively, are zero). Recall Definition 6.6 of $\overline{\text{PROD}}$, in which we defined the vector $\overline{s}_i$ to be the result of multiplying the ith bit of $\overline{z}^+$, denoted z^+_i, with $\overline{y}^+$, and then padding it with i zeros to the right. First, we show that IPS can prove that this multiplication step is correct, in the sense that IPS has an $O(r)$-size proof of:

$$\text{VAL}(\overline{s}_i) = \text{VAL}(\overline{y}^+) \cdot z^+_i \cdot 2^i\,, \tag{12}$$

for every $i = 0, \ldots, r$. Indeed, for every $i = 0, \ldots, r$, by definition of $\overline{s}_i$ we have the following polynomial identities:

$$\begin{aligned}\mathrm{VAL}(\overline{s}_i) &= \sum_{j=i}^{r+i-1} (y^+_{j-i} z^+_i) 2^j - (y^+_r \cdot z^+_i) 2^{r+i} = \left(\sum_{j=0}^{r-1} y^+_j 2^j \right) \cdot z^+_i \cdot 2^i \\ &= \mathrm{VAL}(\overline{y}^+) \cdot z^+_i \cdot 2^i\end{aligned}$$

(we have used $y^+_r = z^+_r = 0$ here).

Second, based on the proof of the case of addition (Case 1 above), we can derive

$$\begin{aligned}&\mathrm{VAL}(\overline{\mathrm{ADD}}(\overline{s}_r, \overline{\mathrm{ADD}}(\overline{s}_{r-1}, \ldots, \overline{\mathrm{ADD}}(\overline{s}_0, \overline{s}_1) \ldots))) \qquad (13)\\ &\qquad = \mathrm{VAL}(\overline{s}_r) + \mathrm{VAL}(\overline{\mathrm{ADD}}(\overline{s}_{r-1}, \ldots, \overline{\mathrm{ADD}}(\overline{s}_0, \overline{s}_1) \ldots))) \\ &\qquad \quad \ldots \\ &\qquad = \mathrm{VAL}(\overline{s}_r) + \cdots + \mathrm{VAL}(\overline{s}_2) + \mathrm{VAL}(\overline{\mathrm{ADD}}(\overline{s}_0, \overline{s}_1)) \\ &\qquad = \sum_{i=0}^{r} \mathrm{VAL}(\overline{s}_i)\,. \qquad (14)\end{aligned}$$

Consider line eq. 13: each $\overline{\mathrm{ADD}}$ there contributes $O(r)$ gates. Thus, in total eq. 13 has a circuit of size $O(r^2)$. Since line eq. 13 is of size $O(r^2)$, every step in which we use the addition case of the induction statement (Case 1), takes $r^{c'}$, for some constant $c' > 2$ independent of r. Hence, overall we obtain an IPS proof of the equality between eq. 13 and eq. 14, of size $r^{b''}$, for some constant b'' independent of r.

Using (eq. 12) and $z^+_r = 0$ we conclude with an IPS proof that eq. 13 above (which by Definition 6.6 is $\mathrm{VAL}\left(\overline{\mathrm{PROD}}_+(\overline{y}^+, \overline{z}^+)\right)$) equals

$$\mathrm{VAL}(\overline{y}^+) \cdot \left(\sum_{i=0}^{r-1} z^+_i 2^i \right),$$

which in turn is equal to $\mathrm{VAL}(\overline{y}^+) \cdot \mathrm{VAL}(\overline{z}^+)$, by definition of VAL and the fact that $z^+_r = 0$. This amounts to an IPS proof of total size r^c, for a constant c independent of r. ■Claim

Finally, we arrive at the main case of multiplying two possibly negative integers written in the two's complement scheme, each with bit vector of length r. Let $s = y_{r-1} \oplus z_{r-1}$ and let $\overline{m} = \mathbf{e}(s)$ be a vector of length r in which every bit is s. Recall that

$$\overline{\mathrm{PROD}}(\overline{y}, \overline{z}) = \overline{\mathrm{ADD}}\left(\overline{\mathrm{PROD}}_+\left(\overline{\mathrm{ABS}}(\overline{y}), \overline{\mathrm{ABS}}(\overline{z}) \right) \oplus \overline{m}, s \right).$$

Claim 6.14. $\mathrm{VAL}\left(\overline{\mathrm{PROD}}(\overline{y}, \overline{z})\right) = (1-2s) \cdot \mathrm{VAL}\left(\overline{\mathrm{PROD}}_+\left(\overline{\mathrm{ABS}}(\overline{y}), \overline{\mathrm{ABS}}(\overline{z})\right)\right)$ *has an IPS derivation from the Boolean axioms of size r^c, for some constant c independent of r.*

Proof of claim: Consider the following two cases.
Case 1: $s = 1$. Note that inverting all bits affects the value of a bit vector as follows: if $\overline{x}$ is a length k bit vector, then

$$\mathrm{VAL}(\overline{x} \oplus \mathbf{e}(s)) = \sum_{i=0}^{k-2} (1 - x_i) 2^i - (1 - x_{k-1}) 2^{k-1} = -1 - \mathrm{VAL}(\overline{x}). \qquad (15)$$

Hence, since $s = 1$,

$$\begin{aligned}
\mathrm{VAL}\Big(\overline{\mathrm{PROD}}(\overline{y},\overline{z})\Big) &= \mathrm{VAL}\left(\overline{\mathrm{ADD}}\left(\overline{\mathrm{PROD}}_+(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z}))\oplus\overline{m},s\right)\right) \text{ by definition of } \overline{\mathrm{PROD}}\\
&= \mathrm{VAL}\left(\overline{\mathrm{PROD}}_+(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z}))\oplus\overline{m}\right)+1\\
&\qquad\qquad \text{by Case 1 (addition) of induction statement}\\
&= -1-\mathrm{VAL}\left(\overline{\mathrm{PROD}}_+(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z}))\right)+1 \qquad \text{by eq. 15}\\
&= (1-2s)\cdot\mathrm{VAL}\left(\overline{\mathrm{PROD}}_+(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z}))\right) \qquad \text{since } s=1.
\end{aligned}$$

Case 2: $s = 0$. This is an easier case, in which we show $\mathrm{VAL}\Big(\overline{\mathrm{PROD}}(\overline{y},\overline{z})\Big) = \mathrm{VAL}(\overline{\mathrm{PROD}}_+\left(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z})\right))$, and so we are done by $s = 0$. We omit the details.

Using reasoning by Boolean cases in IPS according to Prop. A.5 we conclude the claim. ■Claim

Taking together Claim 6.14, Claim 6.13 and Claim 6.11 (for $\overline{y}$ and for $\overline{z}$ of length t) we get the desired equality for the product case, where $s = y_{t-1} \oplus z_{t-1}$:

$$\begin{aligned}
&\mathrm{VAL}\left(\overline{\mathrm{PROD}}(\overline{y},\overline{z})\right)\\
&= (1-2s)\cdot\mathrm{VAL}\left(\overline{\mathrm{PROD}}_+\left(\overline{\mathrm{ABS}}(\overline{y}),\overline{\mathrm{ABS}}(\overline{z})\right)\right)\\
&= (1-2s)\cdot\mathrm{VAL}\left(\overline{\mathrm{ABS}}(\overline{y})\right)\cdot\mathrm{VAL}\left(\overline{\mathrm{ABS}}(\overline{z})\right)\\
&= (1-2y_{r-1})\cdot\mathrm{VAL}\left(\overline{\mathrm{ABS}}(\overline{y})\right)\cdot(1-2z_{r-1})\cdot\mathrm{VAL}\left(\overline{\mathrm{ABS}}(\overline{z})\right)\\
&= \mathrm{VAL}(\overline{y})\cdot\mathrm{VAL}(\overline{z})\,,
\end{aligned}$$

where the penultimate equation stems from the polynomial identity $(1-2y_{t-1})\cdot(1-2z_{t-1}) = 1-2(y_{t-1}\oplus z_{t-1})$.

This concludes the proof of the first part of Lemma 6.9. For the second part, assuming that $F(\overline{x})$ is constant-free, the proof is identical, noting simply that in the IPS proof we constructed above all coefficients are at most exponential in n, and thus by the upper bound $\tau(m) \le O(\log m)$ for every $m \in \mathbb{N}$, we get a constant-free IPS proof of size $\mathrm{poly}(n)$. □

7 Algebraic versus Semi-Algebraic Proof Systems

Here we show that IPS simulates CPS over $\mathbb{Q}$ assuming the existence of small IPS refutations for the generalized binary value principle (and the binary value principle for the case of $\mathbb{Z}$). We work with the *Boolean versions* of both CPS and IPS, meaning that the Boolean axioms are present.

We demonstrate two kinds of conditional simulations: a (standard) polynomial simulation for the language of unsatisfiable sets $\overline{\mathcal{F}}$ of polynomial *equations*, and in Sect. 7.2 an *effective simulation* (in the sense of Pitassi-Santhanam [46]) for the language of unsatisfiable sets containing both equations $\overline{\mathcal{F}}$ and inequalities $\overline{\mathcal{H}}$ over $\mathbb{Z}$; similar reasoning works over $\mathbb{Q}$). Note that we cannot hope to show a (standard) simulation of CPS by IPS for the language containing both polynomial equalities and polynomial inequalities, because inequalities are not expressible directly as polynomial equations in IPS; hence, for the sake of the second kind of simulation we first translate $\overline{\mathcal{H}}$ to bit representation and only then simulate the CPS proof, yielding an effective simulation.

We now prove the simulation for constant-free proofs over $\mathbb{Q}$, and in Sect. 7.2 we will prove the effective simulation (over $\mathbb{Z}$, which implies the same result over $\mathbb{Q}$).

Recall that $\mathrm{IPS}_{\mathbb{Q}}$ and $\mathrm{CPS}_{\mathbb{Q}}$ stand for IPS and CPS proofs over $\mathbb{Q}$, respectively, and that by Prop. 4.3, given a constant-free circuit C over $\mathbb{Q}$ we can turn it into a constant-free circuit C' over $\mathbb{Z}$ computing $M \cdot \widehat{C}$, for some nonzero integer M, with $|C'| \leq 4|C|$ and $\tau(M) \leq 4|C|$.

Definition 7.1 (syntactic length of a circuit over $\mathbb{Q}$)**.** *The syntactic length of a circuit C over $\mathbb{Q}$ is the syntactic length of the corresponding circuit C' over $\mathbb{Z}$ constructed from C in Prop. 4.3.*

The main technical theorem of this section is the following:

Theorem 7.2 (conditional simulation of constant-free Boolean $\mathrm{CPS}_{\mathbb{Q}}$ by constant-free Boolean $\mathrm{IPS}_{\mathbb{Q}}$)**.** *Let $\overline{\mathcal{F}}$ denote a system of polynomial equations over $\mathbb{Q}$ written as constant-free circuits $\{F_i(\overline{x}) = 0\}_{i \in I}$ and let $C(\overline{x}, \overline{\mathcal{F}}) = -1$ be a constant-free $\mathrm{CPS}_{\mathbb{Q}}$ refutation of $\overline{\mathcal{F}}$ where $C(\overline{x}, \overline{\mathcal{F}})$ is of size s and syntactic length t (as in Definition 7.1).*[12] *Assume that the binary value principle $\mathrm{BVP}_{t,M}$ has a size $\leq r$ constant-free $\mathrm{IPS}_{\mathbb{Q}}$ refutation, for every given positive integer M with $\tau(M) = O(s)$. Then, there is a constant-free $\mathrm{IPS}_{\mathbb{Q}}$ refutation of $\overline{\mathcal{F}}$ with size $\mathrm{poly}(s, t, r)$.*

Remark 7.3.

1. *By inspection of the proof of Thm. 7.2 one can see that the degree of the simulating IPS refutation can be exponential in the size of the resulting circuit (clearly, the degree cannot be larger than that).*
2. *Assuming that indeed propositional IPS simulates propositional CPS, by Prop. 5.19 propositional IPS also simulates any propositional CPS (or Positivstellensatz/*SoS*) refutation of CNF formulas given as* inequalities*. This is because if propositional CPS has a short refutation for a CNF given as inequalities (Definition 2.11) then from Prop. 5.19, propositional CPS also has a short refutation of the CNF given as equations (Definition 2.5).*

Since the simulation of CPS by IPS in Thm. 7.2 depends on the syntactic length t of the simulated CPS proof, if we aim to achieve a (polynomial) simulation we need to bound the syntactic length of the CPS proofs to be at most polynomial in the proof size. We denote this restricted proof system by $\mathrm{CPS}^{\star}_{\mathbb{Z}}$ and $\mathrm{CPS}^{\star}_{\mathbb{Q}}$. In other words, a family $\{\pi_i\}_{i=1}^{\infty}$ of $\mathrm{CPS}_{\mathbb{Z}}$ (resp. $\mathrm{CPS}_{\mathbb{Q}}$) proofs is said to be *a family of $\mathrm{CPS}^{\star}_{\mathbb{Z}}$ (resp. $\mathrm{CPS}^{\star}_{\mathbb{Q}}$) proofs* if there is a constant c such that for every $i \in \mathbb{N}$, the syntactic length of π_i is at most $|\pi_i|^c$. In other words, the maximal value (over $\{0,1\}$-assignments to the variables) of every gate in $\mathrm{CPS}^{\star}_{\mathbb{Z}}$ proof-sequence $\{\pi_i\}_{i=1}^{\infty}$ is bounded from above by $2^{|\pi_i|^{O(1)}}$.

It is important to note that most known examples of short semi-algebraic proofs of propositional formulas *have polynomial syntactic length*, as the multiplication of arbitrary inequalities is not used, and multiplying by x or by $1 - x$ for a variable x increases the syntactic length additively. The use of division by scalars (for example, in the LS proof of PHP) can increase the syntactic length in Prop. 4.3; however, as those scalars have at most exponential (actually, polynomial) values, the transformation from rational numbers to integers can bring at most $(exp(\mathrm{poly}(n)))^{\text{proof-size}}$ factor, thus a polynomial number of bits.

Recall the terminology in Sect. 4.1: a refutation in $\mathrm{IPS}_{\mathbb{Z}}$ means a proof of M for some nonzero integer M. Further, we say that $\mathrm{IPS}_{\mathbb{Z}}$ simulates $\mathrm{CPS}_{\mathbb{Q}}$ if a size-s $\mathrm{CPS}_{\mathbb{Q}}$ proof of p from assumptions $\overline{\mathcal{F}}$ over $\mathbb{Z}$ implies that there is a $\mathrm{poly}(s)$-size $\mathrm{IPS}_{\mathbb{Z}}$ proof of $M \cdot p$ from $\overline{\mathcal{F}}$, for some nonzero integer M.

The binary value principle thus characterizes exactly the apparent advantage CPS has over IPS, in the following sense:

[12] We need to consider also the size of the CPS refutation *after* the substitution of $\overline{\mathcal{F}}$ for the placeholder variables, that is, $C(\overline{x}, \overline{\mathcal{F}})$, because of the slightly peculiar nature of IPS (similar to CPS) in which the size of a refutation does not include directly the size of the assumptions it refutes.

Corollary 7.4 (BVP characterizes the strength of Boolean CPS)**.** *In what follows, IPS and CPS stand for Boolean IPS and Boolean CPS, respectively, where both are proof systems for refuting unsatisfiable sets of polynomial equalities (not necessarily CNF formulas).*

1. *Constant-free* $\mathrm{IPS}_{\mathbb{Z}}$ *simulates constant-free* $\mathrm{CPS}^{\star}_{\mathbb{Z}}$ *iff constant-free* $\mathrm{IPS}_{\mathbb{Z}}$ *admits* $\mathrm{poly}(t)$*-size refutations of* BVP_t*.*

2. *Constant-free* $\mathrm{IPS}_{\mathbb{Q}}$ *simulates constant-free* $\mathrm{CPS}^{\star}_{\mathbb{Q}}$ *iff for every positive integer* M*, constant-free* $\mathrm{IPS}_{\mathbb{Q}}$ *admits* $\mathrm{poly}(t, \tau(M))$*-size refutations of* $\mathrm{BVP}_{t,M}$*.*

Proof: We show the proof of item 2 (which includes all the ideas for the other case).
($\Leftarrow$) Assume that for every positive integer M constant-free $\mathrm{IPS}_{\mathbb{Q}}$ admits $\mathrm{poly}(t, \tau(M))$-size refutations of $\mathrm{BVP}_{t,M}$. Then specifically for $\tau(M) = O(s)$ there is a $\mathrm{poly}(t, s)$ upper bound on the size of constant-free $\mathrm{IPS}_{\mathbb{Q}}$ refutations of $\mathrm{BVP}_{t,M}$. By Thm. 7.2 if there exists a syntactic-length t constant-free $\mathrm{CPS}^{\star}_{\mathbb{Q}}$ refutation of $\overline{\mathcal{F}}$ then there exists a constant-free IPS refutation of $\overline{\mathcal{F}}$ with size $\mathrm{poly}(s, t, r) = \mathrm{poly}(s)$, because $t = \mathrm{poly}(s)$ by assumption and $r = \mathrm{poly}(s, t)$.
($\Rightarrow$) This follows from the $\mathrm{CPS}_{\mathbb{Z}}$ upper bound on BVP_n demonstrated in Prop. 5.6. More precisely, it suffices to show that given a positive integer M there are constant-free $\mathrm{CPS}^{\star}_{\mathbb{Q}}$ refutations of $\mathrm{BVP}_{t,M}$ having $\mathrm{poly}(t, \tau(M))$-size. Using the notation as in the proof of Prop. 5.6, we claim that the conic circuit $\frac{1}{M} \cdot \left(\sum_{i=1}^{t} 2^{i-1} \cdot y_i\right) + \frac{1}{M} \cdot y_{t+1}$ serves as such a refutation. Indeed, this conic circuit is easily written as an $O(t \cdot \log t + \tau(M))$-size *constant-free* circuit. This is because $\tau(2^{i-1}) = \log(i-1)$, for every $i = 1, \ldots, t$, and $1/M$ is clearly of size $2 + \tau(M)$. That this conic circuit is a refutation of $\mathrm{BVP}_{t,M}$ follows immediately from the definition (see the proof of Prop. 5.6).

The proof of item 1 is similar and we omit the details. □

Remark 7.5. *The results above in Thm. 7.2 and Cor. 7.4 hold trivially also in the* unit-cost model*, where we consider the size of coefficient in the ring or field to be* 1*. More precisely, if we replace the term "constant-free proof" with the term "proof" the results still hold. This is because we limit the syntactic length of the original CPS circuit, and the size of circuit families of polynomial syntactic length in the unit-cost model is smaller or equal than their size in the constant-free model. And if a family of constant-free circuits (proofs)* C_n *simulates a family of constant-free circuits* with a polynomial syntactic length D_n*, then the corresponding circuit family* C'_n *in the unit-cost model also simulates the corresponding circuit family* D'_n *in the unit-cost model (because* $|D_n| \leq \mathrm{poly}(|D'_n|)$*).*

7.1 Proof of Thm. 7.2

We need to show that there is an $\mathrm{IPS}_{\mathbb{Z}}$ refutation of $\overline{\mathcal{F}}$. We first translate the setting to the integers, since this will allow us to use the main binary value Lemma 6.9 which is stated for $\mathbb{Z}$, as follows: we take the $\mathrm{CPS}_{\mathbb{Q}}$ refutation, turn it into a $\mathrm{CPS}_{\mathbb{Z}}$ refutation without increasing the size too much (the syntactic length stays the same by definition), and then simulate this refutation in $\mathrm{IPS}_{\mathbb{Z}}$, that is, construct an $\mathrm{IPS}_{\mathbb{Z}}$ proof from $\overline{\mathcal{F}}$ of a nonzero integer M. Dividing this $\mathrm{IPS}_{\mathbb{Z}}$ refutation by M we get the desired $\mathrm{IPS}_{\mathbb{Q}}$ refutation of $\overline{\mathcal{F}}$. We formalize this conversion in the following proposition:

Proposition 7.6 (going from constant-free $\mathrm{CPS}_{\mathbb{Q}}$ to constant-free $\mathrm{CPS}_{\mathbb{Z}}$)**.** *Let* $\overline{\mathcal{F}}$ *denote a system of polynomial equations over* $\mathbb{Q}$ *written as constant-free circuits* $\{F_i(\overline{x}) = 0\}_{i \in I}$ *and let* $C(\overline{x}, \overline{\mathcal{F}}) = -1$ *be a constant* $\mathrm{CPS}_{\mathbb{Q}}$ *refutation of* $\overline{\mathcal{F}}$*, where* $C(\overline{x}, \overline{\mathcal{F}})$ *has size* s *and syntactic length* t*. Then, there exists a set of polynomial equations over* $\mathbb{Z}$ *denoted* $\overline{\mathcal{F}}^{\star} = \{F_i^{\star}(\overline{x}) = 0\}_{i \in I}$*, where* $F_i^{\star}(\overline{x}) = M_i \cdot F_i(\overline{x})$ *for some nonnegative* $M_i \in \mathbb{Z}$*, for all* $i \in I$*, and a constant-free* $\mathrm{CPS}_{\mathbb{Z}}$ *proof* $C^{\star}(\overline{x}, \overline{y})$ *from* $\overline{\mathcal{F}}^{\star}$ *of* $M \cdot (-1)$*, for some nonzero* $M \in \mathbb{Z}$*, such that* $C^{\star}(\overline{x}, \overline{\mathcal{F}}^{\star})$ *has both size and syntactic length* $\mathrm{poly}(s, t)$*.*

Proof: The proof is identical to the proof of Prop. 4.3 (cf. Cor. 4.4). Specifically, given a constant-free circuit D over $\mathbb{Q}$ the Induction Statement in the proof of Prop. 4.3 shows that there exists a size at most $4|D|$ constant-free circuit $D^\star$ over $\mathbb{Z}$ that computes $M \cdot \widehat{D}$ for some nonzero integer M. Accordingly, we turn $\overline{\mathcal{F}}$ into $\overline{\mathcal{F}^\star}$ and $C(\overline{z},\overline{y})$ into $C^\star(\overline{z},\overline{y})$ in this way. By definition of syntactic length for circuits over $\mathbb{Q}$ the syntactic length of $C^\star(\overline{z},\overline{y})$ is t. □

By Prop. 7.6, to prove Thm. 7.2 we can assume without loss of generality that $\overline{\mathcal{F}}$ is a system of constant-free-circuit equations *over* $\mathbb{Z}$ and that $C(\overline{x},\overline{\mathcal{F}}) = -M$ is a constant-free $\mathrm{CPS}_{\mathbb{Z}}$ refutation, where $C(\overline{x},\overline{\mathcal{F}})$ is of size s and syntactic length t. Thus, *from now on we assume that all constant-free circuits and proofs are over* $\mathbb{Z}$.

Given a multi-output circuit of size s, with m output gates, each computing the circuit H_i (for $i \in [m]$), we assume that an algebraic circuit for $\sum_{j=1}^{m} H_j^2$ is defined to be a sum of m summands, written as a binary tree of logarithmic in m depth, in which each summand H_j^2 is defined as the circuit whose output is a product gate with its two children connected to the output gate of H_j, and where different H_j's can have common nodes (so that the size of the circuit computing $\sum_{j=1}^{m} H_j^2$ is linear in s).

Lemma 7.7 (sign bit of sum of squares is zero)**.** *Consider the circuit* $H = \sum_{j\in J} H_j^2$*, and let* $\mathrm{BIT}_t(H)$ *be the sign bit of* $\overline{\mathrm{BIT}}(H)$*. Then* $\mathrm{BIT}_t(H) = 0$ *has a polynomial-size IPS proof (using only the Boolean axioms).*

Proof: Informally, the idea is to prove the desired equation using only the structure of sign bits of additions and squares appearing in top layers only (the layers close to the output gate) of H, without looking at the individual structure of the circuits H_j's.

First, we show that the sum of two nonnegative numbers is nonnegative, that is, if a pair of circuits have sign bits that are zero then the sign bit of their addition is also zero, and in symbols:

$$\mathrm{BIT}_t(F) = 0,\ \mathrm{BIT}_t(G) = 0 \vdash_{\mathrm{IPS}}^{\mathrm{poly}(|F|,|G|)} \mathrm{BIT}_{t+1}(F+G) = 0\,,$$

where the sign bit of F, G is bit t and the sign bit of $F+G$ is bit $t+1$.

Let $y := \mathrm{BIT}_t(F)$ and $z := \mathrm{BIT}_t(G)$, then by Definition 6.3 the sign bit of $F+G$ is computed as $y \oplus z \oplus \mathrm{CARRY}_{t+1}(\overline{\mathrm{BIT}}(F), \overline{\mathrm{BIT}}(G))$, because we have padded F and G by their sign bits y, z, respectively, before the addition. Given that $y = 0$ and $z = 0$ by assumption, we need to prove that $\mathrm{CARRY}_{t+1}(\overline{\mathrm{BIT}}(F), \overline{\mathrm{BIT}}(G)) = 0$. By Definition 6.3 $\mathrm{CARRY}_{t+1}(\overline{\mathrm{BIT}}(F), \overline{\mathrm{BIT}}(G)) = (y \wedge z) \vee ((y \vee z) \wedge \cdots))$. Since the arithmetic expressions (according to Definition 6.2) for $y \wedge z$ and $y \vee z$ can be easily proved to be zero (from $y = 0$, $z = 0$), and the same holds for $0 \wedge \cdots$, we conclude that the sign bit of $F + G$ is zero.

To prove that each of the squares H_j^2 are nonnegative, one needs to consider the two cases of the sign bit x of H_j and infer that the sign bit of the square is zero in both cases using Prop. A.5.

Recall that

$$\overline{\mathrm{PROD}}(\overline{y},\overline{z}) := \overline{\mathrm{ADD}}\left(\overline{\mathrm{PROD}}_+\left(\overline{\mathrm{ABS}}(\overline{y}), \overline{\mathrm{ABS}}(\overline{z})\right) \oplus \overline{m}, s\right),$$

where $s = y_{t'} \oplus z_{t'}$ and $\overline{m} = \mathbf{e}(s)$, with $y_{t'}, z_{t'}$ the sign bits of $\overline{y}, \overline{z}$ as bit vectors in the two's complement notation, respectively.

In both cases of the sign of H_j, we have $s = 0$ and $\overline{m} = \overline{0}$ as y and z are equal in our case. Everything else is identical in both cases: the sign bit of $\overline{\mathrm{PROD}}_+$ is always zero, because $\overline{\mathrm{PROD}}_+$ is a consecutive sum of nonnegative numbers (the sign of each of those numbers s_i from the definition of $\overline{\mathrm{PROD}}_+$ is obtained by $\wedge$-ing a single bit with the sign of $\overline{\mathrm{ABS}}$, the latter being zero by Claim 6.12), and we have already proved that the sum of nonnegative numbers is nonnegative. Applying the latter fact once again, we conclude that the sign of H_j^2 is zero in both cases. □

We will need the following simple lemma:

Lemma 7.8. *Let G be an algebraic circuit which is an arithmetization of a Boolean circuit g (Definition 6.2). Then, IPS has a polynomial-size in $|G|$ derivation of $G^2 - G$ from the Boolean axioms.*

Proof: This is proved by induction on $|G|$; see for example [25, Lemma 4], where this is proved for Polynomial Calculus over algebraic formulas denoted $\mathcal{F}$-PC. □

Since for any circuit F, $\mathrm{BIT}_i(F)$ is the result of an arithmetization of a Boolean circuit we have:

Corollary 7.9. *Let F be a circuit, then IPS has a polynomial-size derivation of $\mathrm{BIT}_i(F)^2 - \mathrm{BIT}_i(F)$ from the Boolean axioms.*

Lemma 7.10 (sign bit of literals is zero)**.** *Let x_i be a variable and let $\mathrm{BIT}_1(x_i)$ and $\mathrm{BIT}_1(1-x_i)$ be the sign bits of of $\overline{\mathrm{BIT}}(x_i)$ and $\overline{\mathrm{BIT}}(1-x_i)$, respectively. Then $\overline{\mathrm{BIT}}(x_i) = 0$ and $\overline{\mathrm{BIT}}(1-x_i) = 0$ have constant-size IPS proofs (using only the Boolean axioms).*

Proof: Observe that indeed the syntactic length of x_i and $1 - x_i$ is 2. Now, $\mathrm{BIT}_1(x_i) = 0$ holds by definition, since we define $\overline{\mathrm{BIT}}(x_i) = 0x_i$ (Definition 6.7). For $\mathrm{BIT}_1(1 - x_i) = 0$, this follows by considering the two options $x_i = 0$ and $x_i = 1$ (where the size of the proofs is constant, since the statement itself is of constant size, namely, it involves only a single variable and a two-bit vector). □

Lemma 7.11 (sign bits of axioms are zero)**.** *Under the assumption that BVP_n has $\mathrm{poly}(n)$-size IPS refutations, there are polynomial-size IPS proofs of $\mathrm{BIT}_t(f(\overline{x})) = 0$ from $f(\overline{x}) = 0$ and the Boolean axioms, where $t+1$ is the syntactic length of $f(\overline{x})$.*

Proof: By Lemma 6.9 we know that $\mathrm{VAL}(\overline{\mathrm{BIT}}(f)) = f$, and hence by assumption $\mathrm{VAL}(\overline{\mathrm{BIT}}(f)) = 0$. We need to show that under $\mathrm{VAL}(\overline{\mathrm{BIT}}(f)) = 0$ we can infer $\mathrm{BIT}_t(f) = 0$ with a short IPS proof. Note that this inference is a substitution instance of the following inference:

$$\sum_{i=1}^{t} 2^{i-1}x_i - 2^t x_{t+1} = 0 \ \vdash_{\mathrm{IPS}} \ x_{t+1} = 0, \tag{16}$$

where we substitute $\mathrm{BIT}_{i-1}(f)$ for x_i $(i = 1, \ldots, t+1)$. By Fact A.8, IPS proofs are closed under substitution instances (together with the fact that the corresponding substitution instances of the Boolean axioms $\overline{x}^2 - \overline{x}$ are also provable in IPS by Cor. 7.9) and so it remains to show that under the assumption that BVP has polynomial-size IPS refutations, eq. 16 holds.

To prove eq. 16 it suffices to show that the assumptions $x_{t+1} = 1$ and $\sum_{i=1}^{t} 2^{i-1}x_i - 2^t x_{t+1} = 0$ can be refuted with a polynomial-size IPS refutation.

Assuming $x_{t+1} = 1$, $\sum_{i=1}^{t} 2^{i-1}x_i - 2^t x_{t+1} = 0$ becomes $\sum_{i=1}^{t} 2^{i-1}x_i - 2^t = 0$, and so it remains to show the following:

Claim. *Under the assumption that BVP_n has $\mathrm{poly}(n)$-size IPS refutations, there are polynomial-size IPS refutations of $\sum_{i=1}^{t} 2^{i-1}x_i - 2^t = 0$.*

Proof of claim: Our assumption that there are polynomial-size IPS refutations of BVP_{t+1} $\sum_{i=1}^{t+1} 2^{i-1}x_i + 1 = 0$, implies that there are short refutation also of its substitution instance $\sum_{i=1}^{t+1} 2^{i-1}(1 - y_i) + 1 = 0$ (again, by Fact A.8 and the fact that the substitution instance of the Boolean axioms $\overline{x}^2 - \overline{x}$, are easily provable when substituting $1 - y_i$ for x_i's; cf. Lemma Lemma 7.8). But $\sum_{i=1}^{t+1} 2^{i-1}(1 - y_i) + 1 = -(\sum_{i=1}^{t+1} 2^{i-1}y_i - 2^t) = 0$. ■Claim □

Up to now, we have shown that for each algebraic circuit in the "base" of the conic circuit $C(\overline{x},\overline{y})$ comprising a CPS refutation (namely, the sub-circuits that substitute the placeholder variables $\overline{y}$, as well as the $\overline{x}$ variables themselves), the sign bit can be proved to be zero in IPS. The following lemma shows that under these assumptions IPS can prove that the conic circuit $C(\overline{x},\overline{y})$ itself has a zero sign bit (for simplicity we use only $\overline{x}$ variables in the circuit $C(\overline{x})$ below).

Lemma 7.12 (conic circuits preserve zero sign bits)**.** *Let $C(\overline{x})$ be a conic circuit over $\mathbb{Z}$ in the variables $\overline{x}=\{x_1,\ldots,x_n\}$, let $\overline{H}:=\{H_i(\overline{x})\}_{i=1}^{n}$ be n circuits and suppose that t is the syntactic length of $C(\overline{H})$. Then, there is a polynomial-size in $|C(\overline{H})|$ IPS proof that the sign bit of $C(\overline{H})$ is 0, that is, of $\mathrm{BIT}_t(C(\overline{H}))=0$, from the assumptions $\mathrm{BIT}_{t_i-1}(H_i(\overline{x}))=0$, for all $i\in[n]$, where t_i is the syntactic length of $H_i(\overline{x})$.*

Proof: The proof is by induction on the size of C. Note that any conic circuit C is one of the following: (1) a variable x_i, (2) a non-negative constant α, (3) a square of some (not-necessarily conic) circuit, that is, $C=G^2$, or (4) an addition $C=G+H$ or product $C=G\cdot H$ of two conic circuits G,H. Therefore, the base cases of our induction will be the first three cases (1)-(3), and the induction steps will be the latter case (4).

Base case:
Case 1: $C=x_i$. Then from the assumption that $\mathrm{BIT}_{t_j-1}(H_j(\overline{x}))=0$ for all $j\in[n]$, we have that $C(\overline{H})=H_i(\overline{x})$, and so we are done.
Case 2: $C=\alpha$, for a non-negative constant α. Then by Definition 6.7 $\overline{\mathrm{BIT}}(\alpha)$ is the actual bits of α in two's complement. Since α is non-negative $\mathrm{BIT}_{t-1}(C(\overline{H}))=\mathrm{BIT}_t(\alpha)=0$, for t the syntactic length of α.
Case 3: $C=G^2$ for some not-necessarily conic circuit G. This case follows from Lemma 7.7.

Induction step:
Case 1: $C=G+H$. This follows from the claim that the sign bit of the addition of non negative numbers is 0, as shown in the proof of Lemma 7.7.
Case 2: $C=G\cdot H$. This follows from the claim that the sign bit of the product of two non-negative integers is non-negative. □

We are now ready to conclude the main theorem of this section.

Proof of Thm. 7.2. By assumption, $C(\overline{x},\overline{y})$ is a conic circuit constituting a CPS refutation of $\overline{\mathcal{F}}$. We assume that $\{f_i(\overline{x})\}_{i\in I}$ can be computed by a sequence of circuits $\{F_i(\overline{x})\}_{i\in I}$ such that $\sum_{i\in I}|F_i(\overline{x})|=u$. Hence, by the definition of CPS, we set $\overline{\mathcal{H}}$ to be the set of circuits that consists of $F_i(\overline{x})$ and $-F_i(\overline{x})$, for all $i\in I$, as well as the Boolean axioms translation $x_i^2-x_i$ and $-x_i^2+x_i$, for all $i\in[n]$, and x_i and $1-x_i$, for all $i\in[n]$. We thus have $C(\overline{x},\overline{\mathcal{H}})=-M$ as a polynomial identity.

Since C is a conic circuit, and the sign bits of all variables $\overline{x}$ and all circuits in $\overline{\mathcal{H}}$ can be proved in polynomial size (in u) to be 0, by Lemma 7.10 and Lemma 7.11, respectively, we know from Lemma 7.12 that the sign bit of $C(\overline{x},\overline{\mathcal{H}})$ is 0 as well. Since $C(\overline{x},\overline{\mathcal{H}})=-M$ is a polynomial identity, by Fact A.1 $C(\overline{x},\overline{\mathcal{H}})+M$ has an IPS proof of size equal to the size of the circuit $C(\overline{x},\overline{\mathcal{H}})+M$ itself. We now proceed to use the short refutation of the BVP to get a short IPS refutation from the fact that the sign bit of $C(\overline{x},\overline{\mathcal{H}})$ is 0 and $C(\overline{x},\overline{\mathcal{H}})+M=0$. The following claim suffices for this purpose:

Claim 7.13. *Assume that $\mathrm{BVP}_{n,M}$ has $\mathrm{poly}(n,\tau(M))$-size IPS refutations Let $F(\overline{x})$ be a circuit of syntactic length t and size s, such that IPS has a $\mathrm{poly}(s,t)$-size proof of $\mathrm{BIT}_{t-1}(F(\overline{x}))=0$ (where $\mathrm{BIT}_{t-1}(F(\overline{x}))$ is the sign bit of $F(\overline{x})$). Then there is a $\mathrm{poly}(s,t,\tau(M))$ refutation of $F(\overline{x})+M=0$.*

Proof of claim: The size of the circuit $F(\overline{x}) + M$ is $s + \tau(M) + 1$. By Lemma 6.9, $\mathrm{VAL}(\overline{\mathrm{BIT}}(F(\overline{x}) + M)) = F(\overline{x}) + M = 0$ has a polynomial size in $s + \tau(M) + 1$ IPS proof from the Boolean axioms. By the proof of Lemma 6.9 we also have a polynomial-size in s and $\tau(M)$ IPS proof of

$$\mathrm{VAL}\left(\overline{\mathrm{BIT}}(F(\overline{x}))\right) + M = 0,$$

namely, a proof of

$$M + \sum_{i=0}^{t-2} 2^i \cdot w_i \; - 2^{t-1} \cdot w_{t-1} = 0\,, \qquad \text{where } w_i := \mathrm{BIT}_i(F(\overline{x})).$$

By assumption, $w_{t-1} = 0$ has a polynomial-size IPS proof, where w_{t-1} is the sign bit of $F(\overline{x})$. This leads to

$$M + \sum_{i=0}^{t-1} 2^i \cdot w_i = 0. \tag{17}$$

Notice that eq. 17 is the binary value principle in which variables x_i for $i = 1, \ldots, t$, are replaced by the circuits $\mathrm{BIT}_{i-1}(F(\overline{x}))$, denoted w_i. We assumed that the binary value principle has polynomial-size (in t and $\tau(M)$) refutations (using the Boolean axioms). Since IPS proofs are closed under substitutions of variables by circuits (Fact A.8), there is a $\mathrm{poly}(t, |\overline{\mathrm{BIT}}(F)|, \tau(M))$-size IPS refutation of eq. 17 from the substitution instances of the Boolean axioms $w_i^2 - w_i = 0$, for $i = 0, \ldots, t-1$. Since for every $i = 0, \ldots, t-1$, $w_i^2 - w_i = 0$ has a short IPS proof by Cor. 7.9, and since $|\overline{\mathrm{BIT}}(F)| = \mathrm{poly}(t, |F|)$, we conclude that there exists a $\mathrm{poly}(s, t, \tau(M))$-size IPS refutation as desired. ■Claim

□

7.2 Effective Simulation of CPS Refutations with Inequalities

We now turn to conditional effective simulation of CPS as a refutation system for *both equalities and inequalities* by IPS. Effective simulation means that we are allowed to non-trivially translate the input equalities and inequalities before refuting them in IPS, as long as the translation procedure is polynomial-time and preserves unsatisfiability [46]. Similar to the case of conditional simulation, it is enough to consider only the case of CPS and IPS proofs over $\mathbb{Z}$ to conclude it also for $\mathbb{Q}$. We show here the case of non-constant-free Boolean IPS and Boolean CPS proofs over $\mathbb{Z}$. The case over $\mathbb{Q}$ and the cases of constant-free proofs over $\mathbb{Z}$ and $\mathbb{Q}$ are similar.

Note that since the construction of the circuit $\mathrm{BIT}_i(\cdot)$ (Sect. 6.2) is mechanical and uniform, there is a straightforward deterministic (uniform) polynomial-time algorithm that receives a set of polynomial inequalities $\overline{\mathcal{H}} = \{H_j(\overline{x}) \geq 0\}_j$ over $\mathbb{Z}$ written as algebraic circuits (with coefficients written in binary) and outputs the polynomial equations, written as algebraic circuits, expressing that the sign bit of each $H_j(\overline{x})$ is 0 (hence, expressing the inequalities $\overline{\mathcal{H}}$). This translation of inequalities to polynomial equalities serves as our translation from $\overline{\mathcal{H}}$ to the language of polynomial equations that is refutable in IPS. Given an inequality $H_j(\overline{x}) \geq 0$ we denote by $[\![H_j(\overline{x}) \geq 0]\!]$ this translation; accordingly, we let $[\![\overline{\mathcal{H}}]\!] = \{[\![H_j(\overline{x}) \geq 0]\!] \;:\; H_j(\overline{x}) \in \overline{\mathcal{H}}\}$.

Theorem 7.14 (conditional effective simulation of Boolean CPS by Boolean IPS)**.** *Assume that the generalized binary value principle* $\mathrm{BVP}_{t,M}$ *has* $\mathrm{poly}(t, \tau(M))$*-size Boolean IPS refutations for every positive integer* M*. Let* $\overline{\mathcal{H}}$ *denote a system of polynomial inequalities written as circuits* $\{H_j(\overline{x}) \geq 0\}_{j \in J}$*. Let* $C(\overline{x}, \overline{\mathcal{H}}) = -1$ *be a CPS refutation* $\overline{\mathcal{H}}$ *where* $C(\overline{x}, \overline{\mathcal{H}})$ *has size* s *and syntactic length* t*. Then, there is a Boolean IPS refutation of* $[\![\overline{\mathcal{H}}]\!]$ *with size* $\mathrm{poly}(s, t)$*.*

Proof: This is identical to the proof of Thm. 7.2, only that we do not need to prove separately that the axioms in $\overline{\mathcal{H}}$ have all bit vector representation in which the sign bit is 0, since here this is given to us as an assumption. □

8 Conclusions

This work demonstrates that a simple subset-sum principle, written as a linear equation, captures, in the Boolean case (i.e., when variables range over $\{0,1\}$), the apparent advantage of semi-algebraic proofs over algebraic proofs in the following sense: it is necessary for any Boolean algebraic proof system that simulates full Boolean semi-algebraic proofs to admit short refutations of the principle; and if the algebraic proof system is strong enough to be able to efficiently perform bit arithmetic, this condition is also *sufficient* to achieve such a simulation. To formalize these results we introduce a very strong proof system CPS that derives polynomials in the cone of initial axioms instead of in the ideal.

We showed that CPS is expected to be stronger than even the very strong algebraic Ideal Proof System (IPS) formulated by Grochow and Pitassi in [28], since our subset-sum principle is hard for IPS assuming the hardness of computing factorials [55]. We established a related lower bound on IPS refutation-size based on the τ-conjecture [55]. These lower bounds advance the approach introduced by Forbes *et al.* [19]: whereas [19] showed how to obtain restricted IPS lower bounds for certain subset-sum instances, based on known lower bounds against restricted circuit classes, we show how to obtain general IPS lower bounds based on specific hardness assumptions from algebraic complexity.[13]

The conjectured hard instance we introduce (namely, the binary value principle) may have a further important role in proof complexity and related areas. In proof complexity, in parallel to the current work Part and Tzameret [43] showed the binary value principle to be unconditionally hard for proof systems in the weak regime of resolution extensions, and Alekseev [1] subsequently showed similar lower bounds. The binary value principle and related bit-arithmetic instances were also found to be relevant to contemporary SAT-solving as shown recently by Liew *et al.* [37].

Appendix

A Basic Reasoning in IPS

Here we develop basic efficient reasoning in IPS. This is helpful for Sect. 6.2.

First we show that polynomial identities are proved for free in IPS:

Fact A.1. *If $F(\overline{x})$ is a circuit in the variables $\overline{x}$ over the field $\mathbb{F}$ that computes the zero polynomial, then there is an IPS proof of $F(\overline{x}) = 0$ of size $|F|$.*

Proof of fact. The IPS proof of $F(\overline{x}) = 0$ is simply $C(\overline{x}, \overline{z}) := F(\overline{x})$ (note that we do not need to use the Boolean axioms nor any other axioms in this case). Observe that both conditions 1 and 2 for IPS hold in this case (Definition 2.4). □

Fact A.2. *Let F, G, H be circuits and $\mathcal{F}$ be a collection of polynomial equations such that $C : \mathcal{F} \vdash^{s_0}_{\text{IPS}} F = G$ and $C' : \mathcal{F} \vdash^{s_1}_{\text{IPS}} G = H$. Then, $(C + C') : \mathcal{F} \vdash^{s_0+s_1+1}_{\text{IPS}} F = H$.*

Proof of fact. $C(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) + C'(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) = F - G + G - H$. □

Fact A.3. *Let F, G be circuits and $\overline{\mathcal{F}}$ be a collection of polynomial equations such that $C : \overline{\mathcal{F}} \vdash^{s_0}_{\text{IPS}} F = G$ and $C' : \overline{\mathcal{F}} \vdash^{s_1}_{\text{IPS}} H = K$. Then, $(C + C') : \overline{\mathcal{F}} \vdash^{s_0+s_1+1}_{\text{IPS}} F + H = G + K$.*

Proof of fact. $C(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) + C'(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) = F - G + H - K$. □

[13] Note again that extending the approach in [19] to IPS operating with general circuits must result in conditional lower bounds, as long as explicit super-polynomial algebraic circuit lower bounds are not known.

Fact A.4. *Let F, G be circuits and $\overline{\mathcal{F}}$ be a collection of polynomial equations such that $C : \overline{\mathcal{F}} \vdash^{s_0}_{\text{IPS}} F = G$ and $C' : \overline{\mathcal{F}} \vdash^{s_1}_{\text{IPS}} H = K$. Assume that there is a circuit with two output gates, of size s, with one output gate computing H and the other output gate computing G. Then, $\overline{\mathcal{F}} \vdash^{s_0+s_1+s+5}_{\text{IPS}} F \cdot H = G \cdot K$.*

Proof of fact. Observe that $C(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) \cdot H + C'(\overline{x}, \overline{\mathcal{F}}, \overline{x}^2 - \overline{x}) \cdot G = F \cdot H - G \cdot H + H \cdot G - K \cdot G = F \cdot H - G \cdot K$. Hence, the desired proof is the *circuit* $C(\overline{x}, \overline{y}, \overline{z}) \cdot H(\overline{x}) + C'(\overline{x}, \overline{y}, \overline{z}) \cdot G(\overline{x})$, which by assumption that there is a circuit of size s computing both H, G, is at most $s_0 + s_1 + s + 5$ (here, H, G can have common nodes). □

We now wish to show that basic reasoning by *Boolean* cases is efficiently attainable in IPS. Specifically, we are going to show that if for a given constant many variables (or even Boolean valued polynomials) V, for every choice of a fixed (partial) Boolean assignment to the variables V a polynomial equation is derivable, then it is derivable regardless (namely, derivable from the Boolean axioms alone) in polynomial-size.

Proposition A.5 (proof by Boolean cases in IPS)**.** *Let $\mathbb{F}$ be a field. Let $V = \{H_i(\overline{x})\}_{i \in I}$ be a set of circuits with $|V| = r$, and $\overline{\mathcal{F}}$ be a collection of polynomial equations such that $\{H_i^2(\overline{x}) - H_i(\overline{x}) = 0\}_{i \in I} \subseteq \overline{\mathcal{F}}$. Assume that for every fixed assignment $\overline{\alpha} \in \{0,1\}^r$ we have $\overline{\mathcal{F}}, \{H_i(\overline{x}) = \alpha_i\}_{i \in I} \vdash^{s}_{\text{IPS}} f(\overline{x}) = 0$, then $\overline{\mathcal{F}} \vdash^{c^r \cdot s}_{\text{IPS}} f(\overline{x}) = 0$, for some constant c independent of r.*

Proof: We proceed by induction on r.
Base case: $r = 0$. In this case we assume that $\overline{\mathcal{F}} \vdash^{s}_{\text{IPS}} f(\overline{x}) = 0$ and we wish to show that $\overline{\mathcal{F}} \vdash^{c^r \cdot s}_{\text{IPS}} f(\overline{x}) = 0$, for some constant c, which is immediate since $r = 0$.
Induction step: $r > 0$. We assume that for every fixed assignment $\overline{\alpha} \in \{0,1\}^r$ we have $\overline{\mathcal{F}}, \{H_i = \alpha_i\}_{i \in I} \vdash^{s}_{\text{IPS}} f(\overline{x}) = 0$, and we wish to show that $\overline{\mathcal{F}} \vdash^{c^r \cdot s}_{\text{IPS}} f(\overline{x}) = 0$, for some constant c independent of r.

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

$$\overline{\mathcal{F}}, H_1(\overline{x}) = 0, \{H_i(\overline{x}) = \alpha_i\}_{i \in I \setminus 1} \vdash^{s}_{\text{IPS}} f(\overline{x}) = 0, \quad \text{and} \tag{18}$$

$$\overline{\mathcal{F}}, H_1(\overline{x}) = 1, \{H_i(\overline{x}) = \alpha_i\}_{i \in I \setminus 1} \vdash^{s}_{\text{IPS}} f(\overline{x}) = 0. \tag{19}$$

From eq. 18 and eq. 19, by induction hypothesis we have for some constant c independent of r:

$$H_1(\overline{x}) = 0, \overline{\mathcal{F}} \vdash^{c^{r-1} \cdot s}_{\text{IPS}} f(\overline{x}) = 0, \quad \text{and} \tag{20}$$

$$H_1(\overline{x}) = 1, \overline{\mathcal{F}} \vdash^{c^{r-1} \cdot s}_{\text{IPS}} f(\overline{x}) = 0. \tag{21}$$

It thus remains to prove the following claim:

Claim A.6. *Under the above assumptions eq. 20 and eq. 21, we have $\overline{\mathcal{F}} \vdash^{c^r \cdot s}_{\text{IPS}} f(\overline{x}) = 0$.*

Proof of claim: By eq. 20 and eq. 21 we have two IPS proofs $C(\overline{x}, \overline{y}, \overline{z})$ and $C'(\overline{x}, \overline{y}, \overline{z})$ such that $C(\overline{x}, \overline{\mathcal{F}}, H_1(\overline{x}), \overline{x}^2 - \overline{x}) = f(\overline{x})$ and $C'(\overline{x}, \overline{\mathcal{F}}, 1 - H_1(\overline{x}), \overline{x}^2 - \overline{x}) = f(\overline{x})$ (note indeed that $\overline{\mathcal{F}}, H_1(\overline{x})$ and $\overline{x}^2 - \overline{x}$ are the axioms in the former case, and similarly for the latter case, where now $1 - H_1(\overline{x})$ replaces the axiom $H_1(\overline{x})$) each of size $c^{r-1} \cdot s$.

By the definition of IPS $C(\overline{x}, \overline{y}, \overline{z}), C'(\overline{x}, \overline{y}, \overline{z})$ both compute polynomials that are in the ideal generated by $\overline{y}, \overline{z}$. This means that there are some polynomials Q_i, P_i, G, M, L_i, K_i, such that:

$$\widehat{C}(\overline{x}, \overline{\mathcal{F}}, H_1(\overline{x}), \overline{x}^2 - \overline{x}) = \sum_i Q_i \cdot F_i + \sum_i L_i \cdot (x_i^2 - x_i) + G \cdot H_1(\overline{x}) = f(\overline{x}) \quad \text{and}$$

$$\widehat{C}'(\overline{x}, \overline{\mathcal{F}}, 1 - H_1(\overline{x}), \overline{x}^2 - \overline{x}) = \sum_i P_i \cdot F_i + \sum_i K_i \cdot (x_i^2 - x_i) + M \cdot (1 - H_1(\overline{x})) = f(\overline{x})$$

(here, $\overline{\mathcal{F}}, H_1(\overline{x})$ is substituted for $\overline{y}$ in the first equation, and $\overline{\mathcal{F}}, 1 - H_1(\overline{x})$ is substituted for $\overline{y}$ in the second equation).

Hence, we can multiply these two true polynomial identities by $(1 - H_1(\overline{x}))$ and $H_1(\overline{x})$, respectively, to get the following polynomial identities:

$$(1 - H_1(\overline{x})) \cdot \widehat{C}(\overline{x}, \overline{\mathcal{F}}, H_1(\overline{x}), \overline{x}^2 - \overline{x}) =$$
$$(1 - H_1(\overline{x})) \cdot \sum_i Q_i \cdot F_i + (1 - H_1(\overline{x})) \cdot \sum_i L_i \cdot (x_i^2 - x_i) + G \cdot H_1(\overline{x}) \cdot (1 - H_1(\overline{x})) = (1 - H_1(\overline{x})) \cdot f(\overline{x})$$

and

$$H_1(\overline{x}) \cdot \widehat{C}'(\overline{x}, \overline{\mathcal{F}}, H_1(\overline{x}), \overline{x}^2 - \overline{x}) = H_1(\overline{x}) \cdot \sum_i P_i \cdot F_i + H_1(\overline{x}) \cdot \sum_i K_i \cdot (x_i^2 - x_i) + H \cdot H_1(\overline{x}) \cdot (1 - x_1)$$
$$= H_1(\overline{x}) \cdot f(\overline{x}).$$

Each of these two polynomial identities is an IPS proof from the assumptions $\mathcal{F} = \{F_i\}_i$, the Boolean axioms, and the assumption $H_1(\overline{x}) \cdot (1 - H_1(\overline{x})) \in \overline{\mathcal{F}}$ (more formally, $(1 - H_1(\overline{x})) \cdot C$ and $H_1(\overline{x}) \cdot C'$ are the *circuits* that constitute these pair of IPS proofs). Adding these two IPS proofs (note that the addition of two IPS proofs from a set of assumptions is still an IPS proof from that set of assumptions) we obtain the desired IPS proof of $f(\overline{x})$, with size $2 \cdot c^{r-1} \cdot s + c_0 \leq c^r \cdot s$, for a large enough constant c independent of r. $\blacksquare_{\text{Claim}}$ This concludes the proof of the proposition. □

Prop. A.5 allows us to reason by cases in IPS. For example, assume that we know that either $H_i(\overline{x}) = 0$ or $H_i(\overline{x}) = 1$; namely that we have the assumption $H_i(\overline{x}) \cdot (H_i(\overline{x}) - 1) = 0$. Then, we can reason by cases as follows: if we can prove from $H_i(\overline{x}) = 0$ that A, with a polynomial-size proof, and from $H_i(\overline{x}) = 1$ that B, with a polynomial-size proof, then using Prop. A.5 we have a polynomial-size proof that $A \cdot B = 0$ from $H_i(\overline{x}) \cdot (H_i(\overline{x}) - 1) = 0$.

As an immediate corollary of Prop. A.5 we get the same proposition with $H_i(\overline{x})$'s substituted for variables:

Corollary A.7. *Let $\mathbb{F}$ be a field. Let $V = \{x_i\}_{i \in I}$ be a set of variables with $|V| = r$, and $\overline{\mathcal{F}}$ be a collection of polynomial equations. Assume that for every fixed assignment $\overline{\alpha} \in \{0,1\}^r$ to the variables in V we have $\overline{\mathcal{F}}, \{x_i = \alpha_i\}_{i \in I} \vdash^s_{\text{IPS}} f(\overline{x}) = 0$, then $\overline{\mathcal{F}} \vdash^{c^r \cdot s}_{\text{IPS}} f(\overline{x}) = 0$, for some constant c independent of r.*

Fact A.8 (IPS proofs are closed under substitutions)**.** *Let $C(\overline{x}, \overline{y}, \overline{z})$ be an IPS proof of $f(\overline{x})$ from the assumptions $\{F_i(\overline{x})\}_{i=1}^m$, and let $\overline{H} = \{H_i(\overline{x})\}_{i=1}^n$ be a set of algebraic circuits. Then, $C(\overline{H}/\overline{x}, \overline{y}, \overline{z})$ is an IPS proof of $f(\overline{H}/\overline{x})$ from $\{F_i(\overline{H}/\overline{x})\}_{i=1}^m$, where $\overline{H}/\overline{x}$ stands for the substitution of x_i by $H_i(\overline{x})$, for all $i \in [n]$.*

The proof of Fact A.8 is immediate.

Acknowledgement

We wish to thank Michael Forbes, Dima Itsykson, Toni Pitassi and Dima Sokolov for useful discussions at various stages of this work. We also wish to thank the anonymous reviewers both of the extended abstract of this work and its journal version for very useful comments, suggestions, and corrections.

References

[1] Yaroslav Alekseev. A lower bound for polynomial calculus with extension rule. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 21:1–21:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 3.4, 8

[2] Albert Atserias and Tuomas Hakoniemi. Size-degree trade-offs for sums-of-squares and Positivstellensatz proofs. In *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA.*, pages 24:1–24:20, 2019. 2.6, 4

[3] Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *STOC*, pages 307–326, 2012. 1, 2.6

[4] 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, 2.5, 2.5, 2.6

[5] Christoph Berkholz. The Relation between Polynomial Calculus, Sherali-Adams, and Sum-of-Squares Proofs. In Rolf Niedermeier and Brigitte Vallée, editors, *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:14, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. 1

[6] Grigoriy Blekherman, Pablo A. Parrilo, and Rekha Thomas, editors. *Semidefinite Optimization and Convex Algebraic Geometry*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), March 2013. 2.6

[7] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and Real Computation*. Springer-Verlag, Berlin, Heidelberg, 1998. 2.3

[8] Lenore Blum, Mike Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: *np*- completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc. (N.S.)*, 21(1):1–46, 07 1989. 2.3

[9] Peter Bürgisser. On defining integers and proving arithmetic circuit lower bounds. *Computational Complexity*, 18(1):81–103, 2009. 2.2, 2.3

[10] Samuel R. Buss. Polynomial size proofs of the propositional pigeonhole principle. *The Journal of Symbolic Logic*, (52):916–927, 1987. 3.2, 3.3, 6.1

[11] 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. 4.1

[12] Qi Cheng. On the ultimate complexity of factorials. *Theor. Comput. Sci.*, 326(1-3):419–429, October 2004. 2.3

[13] 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

[14] Stephen A. Cook and Robert A. Reckhow. Corrections for "On the lengths of proofs in the propositional calculus (preliminary version)". *SIGACT News*, 6(3):15–22, July 1974. 15

[15] Stephen A. Cook and Robert A. Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing (STOC 1974)*, pages 135–148, 1974. For corrections see Cook-Reckhow [14]. 16

[16] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979. This is a journal-version of Cook-Reckhow [15] and Reckhow [52]. 2.4, 2

[17] W. de Melo and B. F. Svaiter. The cost of computing integers. *Proc. Amer. Math. Soc.*, 124(5):1377–1378, 1996. 2.3

[18] Noah Fleming, Pravesh Kothari, and Toniann Pitassi. Semialgebraic proofs and efficient algorithm design. *Found. Trends Theor. Comput. Sci.*, 14(1-2):1–221, 2019. 1

[19] 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, 2.5, 3, 3.1, 3.2, 3.3, 4.2, 4.9, 4.10, 8, 13

[20] 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. 3.2, 3.3, 6.1

[21] Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, pages 38:1–38:19, 2019. 1

[22] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018. 3.2

[23] Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Tzameret. Simple hard instances for low-depth algebraic proofs. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 2022. 3.4

[24] D. Grigoriev. Complexity of Positivstellensatz proofs for the knapsack. *Comput. Complexity*, 10(2):139–154, 2001. 1, 3.2, 3.3

[25] Dima Grigoriev and Edward A. Hirsch. Algebraic proof systems over formulas. *Theoret. Comput. Sci.*, 303(1):83–102, 2003. Logic and complexity in computer science (Créteil, 2001). 7.1

[26] Dima Grigoriev, Edward A. Hirsch, and Dmitrii V. Pasechnik. Complexity of semialgebraic proofs. *Mosc. Math. J.*, 2(4):647–679, 805, 2002. 1, 2.6, 2.6.1, 2.12, 3.2, 3.3, 5.19, 5.1.3

[27] Dima Grigoriev and Nicolai Vorobjov. Complexity of Null- and Positivstellensatz proofs. *Ann. Pure Appl. Logic*, 113(1-3):153–160, 2002. 1, 2.6, 2.8

[28] 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, 2.5, 2.4, 3.1, 3.2, 4.1, 5, 5.11, 8

[29] David Hilbert. *Hilbert's invariant theory papers*. Lie Groups: History, Frontiers and Applications, VIII. Math Sci Press, Brookline, Mass., 1978. Translated from the German by Michael Ackerman, With comments by Robert Hermann. 2.6

[30] Edward Hirsch and Iddo Tzameret. Nullstellensatz is equivalent to sum-of-squares, over algebraic circuits. In Proof Complexity (Dagstuhl Seminar 18051), pages 124–157. Schloss Dagstuhl Leibniz-Zentrum fuer Informatik, 2018., Feb. 2018. https://materials.dagstuhl.de/files/18/18051/18051.IddoTzameret.Slides.pptx. 3.3

[31] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 233–248. ACM, 2012. 3.2

[32] 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. 3.3

[33] Russell Impagliazzo, Pavel Pudlák, and Jiří Sgall. Lower bounds for the polynomial calculus and the Gröbner basis algorithm. *Computational Complexity*, 8(2):127–144, 1999. 1, 3.2, 3.3

[34] Dmitry Itsykson and Arist Kojevnikov. Lower bounds of static Lovasz-Schrijver calculus proofs for Tseitin tautologies. *Zapiski Nauchnyh Seminarov POMI*, 340:10–32, 2006. (in Russian). English translation appeared in Journal of Mathematical Sciences 145(3):4942-4952, 2007. 3.2

[35] J. L. Krivine. Anneaux preordonnes. *Journal d'Analyse Mathématique*, 12(1):307–326, 1964. 2.6, 2.7

[36] Fu Li, Iddo Tzameret, and Zhengyu Wang. Characterizing propositional proofs as noncommutative formulas. In *SIAM Journal on Computing*, volume 47, pages 1424–1462, 2018. Full Version: http://arxiv.org/abs/1412.8746. 1

[37] Vincent Liew, Paul Beame, Jo Devriendt, Jan Elffers, and Jakob Nordström. Verifying properties of bit-vector multiplication using cutting planes reasoning. In *Proceedings of the 20th Conference on Formal Methods in Computer-Aided Design – FMCAD 2020*, 2020. 3.3, 8

[38] L. Lovász. Stable sets and polynomials. *Discrete Mathematics*, 124:137–153, 1994. 1, 2.6.1

[39] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0–1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991. 1, 2.6.1

[40] Mi Lu. *Arithmetic and Logic in Computer Systems*. Wiley, 2004. 6.1

[41] Ryan O'Donnell. SOS is not obviously automatizable, even approximately. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 59:1–59:10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. 1

[42] Ryan O'Donnell and Yuan Zhou. Approximability and proof complexity. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1537–1556, 2013. 1, 2.6

[43] Fedor Part and Iddo Tzameret. Resolution with counting: Dag-like lower bounds and different moduli. *Comput. Complex.*, 30(1):2, 2021. 3.3, 8

[44] Toniann Pitassi Paul Beame and Nathan Segerlind. Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007. 3.2

[45] Toniann Pitassi. Unsolvable systems of equations and proof complexity. In *Proceedings of the International Congress of Mathematicians, Vol. III (Berlin, 1998)*, number Vol. III, pages 451–460, 1998. 2.5, 5.11

[46] Toniann Pitassi and Rahul Santhanam. Effectively polynomial simulations. In *Proceedings of Innovations in Computer Science - ICS*, pages 370–382, 2010. 7, 7.2

[47] Tonnian Pitassi and Iddo Tzameret. Algebraic proof complexity: Progress, frontiers and challenges. *ACM SIGLOG News*, 3(3), 2016. 1, 4.1

[48] Pavel Pudlák. On the complexity of the propositional calculus. In *Sets and proofs (Leeds, 1997)*, volume 258 of *London Math. Soc. Lecture Note Ser.*, pages 197–218. Cambridge Univ. Press, Cambridge, 1999. 1, 2.6.1

[49] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993. 2.6

[50] Prasad Raghavendra and Benjamin Weitz. On the bit complexity of sum-of-squares proofs. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 80:1–80:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. 1

[51] Alexander A. Razborov. Lower bounds for the polynomial calculus. *Comput. Complexity*, 7(4):291–324, 1998. 1, 3

[52] Robert A. Reckhow. *On the lengths of proofs in the propositional calculus*. PhD thesis, University of Toronto, 1976. 16

[53] Rahul Santhanam and Iddo Tzameret. Iterated lower bound formulas: a diagonalization-based approach to proof complexity. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 234–247. ACM, 2021. 1

[54] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010. 2.2

[55] Michael Shub and Steve Smale. On the intractability of Hilbert's Nullstellensatz and an algebraic version of "NP≠P?". *Duke Math. J.*, 81:47–54, 1995. 2.1, 2.3, (document), 1, 2.2, 2.3, 3.1, 8

[56] Steve Smale. Mathematical problems for the next century. *The Mathematical Intelligencer*, 20(2):7–15, 1998. 2.3, 2.3, (document)

[57] Gilbert Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, 1974. 2.6, 2.7

[58] Volker Strassen. Vermeidung von divisionen. *J. Reine Angew. Math.*, 264:182–202, 1973. (in German). 4.1, 5.1.1

[59] 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. 2.2

[60] Leslie G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. 2.2

[61] Leslie G. Valiant. Negation can be exponentially powerful. *Theor. Comput. Sci.*, 12:303–314, 1980. 4.1, 4.2.3

[62] Leslie G. Valiant. Reducibility by algebraic projections. *Logic and Algorithmic: International Symposium in honour of Ernst Specker*, 30:365–380, 1982. 2.2

— Page left blank for ECCC stamp —

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il