On the Polynomial Parity Argument Complexity of the Combinatorial Nullstellensatz*

Aleksandrs Belovs[1], Gábor Ivanyos[2], Youming Qiao[3], Miklos Santha[4], and Siyi Yang[5]

1 **Faculty of Computing, University of Latvia, Riga, Lettland**
stiboh@gmail.com

2 **Institute for Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary**
Gabor.Ivanyos@sztaki.mta.hu

3 **Centre for Quantum Software and Information, University of Technology Sydney, Sydney, Australia**
jimmyqiao86@gmail.com

4 **IRIF, Université Paris Diderot, CNRS, Paris, France; and Centre for Quantum Technologies, National University of Singapore and MajuLab, CNRS, Singapore**
santha@irif.fr

5 **Centre for Quantum Technologies, National University of Singapore, Singapore**
ysyshtc@gmail.com

Abstract

The complexity class PPA consists of NP-search problems which are reducible to the parity principle in undirected graphs. It contains a wide variety of interesting problems from graph theory, combinatorics, algebra and number theory, but only a few of these are known to be complete in the class. Before this work, the known complete problems were all discretizations or combinatorial analogues of topological fixed point theorems.

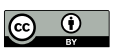
Here we prove the PPA-completeness of two problems of radically different style. They are PPA-Circuit CNSS and PPA-Circuit Chevalley, related respectively to the Combinatorial Nullstellensatz and to the Chevalley-Warning Theorem over the two elements field $\mathbb{F}_2$. The input of these problems contain PPA-*circuits* which are arithmetic circuits with special symmetric properties that assure that the polynomials computed by them have always an even number of zeros. In the proof of the result we relate the multilinear degree of the polynomials to the parity of the *maximal parse subcircuits* that compute monomials with maximal multilinear degree, and we show that the maximal parse subcircuits of a PPA-circuit can be paired in polynomial time.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Chevalley-Warning theorem, Combinatorial Nullstellensatz, PPA

Digital Object Identifier 10.4230/LIPIcs.CCC.2017.30

* This research was partially funded by the Singapore Ministry of Education and the National Research Foundation, also through the Tier 3 Grant "Random numbers from quantum processes," MOE2012-T3-1-009. The research was also supported by the ERC Advanced Grant MQC, the French ANR Blanc program under contract ANR-12-BS02-005 (RDAM project), the Hungarian National Research, Development and Innovation Office – NKFIH Grant K115288 and the Australian Research Council DECRA DE150100720.



32nd Computational Complexity Conference (CCC 2017).
Editor: Ryan O'Donnell; Article No. 30; pp. 30:1–30:24

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

1.1 The class PPA

The complexity class TFNP [21] consists of NP-search problems corresponding to total relations. In the last 25 years various subclasses of TFNP have been thoroughly investigated. The polynomial parity argument classes PPA and PPAD were defined in the seminal work of Papadimitriou [22]. PPA consists of the search problems which are reducible to the parity principle stating that in an undirected graph the number of odd vertices is even. The more restricted class PPAD is based on the analogous principle for directed graphs.

The class PPAD contains a relatively large number of complete problems from various areas of mathematics. In his paper Papadimitrou [22] has already shown that among others the 3-dimensional Sperner and Brouwer problems, as well as the Exchange Equilibrium problem from mathematical economics were PPAD-complete. A few years later Chen and Deng [9] proved that 2-dimensional Sperner was also PPAD-complete, and after a sequence of beautiful papers Chen and Deng [10] has established the PPAD-completeness of computing 2-player Nash equilibrium, see also [11]. Kintali [18] has compiled a list of 25 PPAD-complete problems; the list is far from complete.

In comparison with PPAD, relatively few complete problems are known in the class PPA, all of which are discretizations or combinatorial analogues of topological fixed point theorems. While the original paper of Papadimitriou [22] exhibited a large collection of problems in PPA, none of them was proven to be PPA-complete. Historically the first PPA-completeness result was given by Grigni [14] who, realizing that analogues of PPAD-complete problems in non-orientable spaces could become PPA-complete, has shown the PPA-completeness of the Sperner problem for a non-orientable 3-dimensional space. This result was strengthened by Friedl et al. [17] to a non-orientable and locally 2-dimensional space. Up to our knowledge, until 2015 just these two problems were known to be PPA-complete. Last year Deng et al. [13] established the PPA-completeness of several 2-dimensional problems on the Möbius band, including Sperner and Tucker, and they have obtained similar results for the Klein bottle and the projective plane. Recently Aisenberg, Bonet and Buss [1] have shown that 2-dimensional Tucker in the Euclidean space was PPA-complete.

Compared to the fundamental similarity of these complete problems in PPA, the list of problems in the class for which no completeness result is known is very rich. Already in Papadimitriou's paper [22] we find problems from graph theory, such as Smith and Hamiltonian decomposition, from combinatorics, such as Necklace splitting and Discrete Ham sandwich (the proof in [23] that these problems are in PPAD was incorrect [1]), and from algebra, a variant of Chevalley's theorem over the 2 elements field $\mathbb{F}_2$, which we call Explicit Chevalley. Cameron and Edmonds [8] gave new proofs based on the parity principle for a long series of theorems from graph theory [25, 29, 6, 5, 7], the corresponding search problems are therefore in PPA. Recently Jeřábek [15] has put several number theoretic problems, such as square root computation and finding quadratic nonresidues modulo n into PPA, and he has also shown that Factoring is in PPA under randomized reduction.

1.2 Our contribution

The main result of this paper is that two appropriately defined problems related to Chevalley-Warning Theorem [12, 28] and to Alon's Combinatorial Nullstellensatz [2] over $\mathbb{F}_2$ are complete in PPA. These are the first PPA-completeness results involving problems which are not inspired by topological fixed point theorems.

The Chevalley-Warning Theorem is a classical result about zeros of polynomials. It says that if $P_1, \ldots, P_k$ are n-variate polynomials over a field of characteristic p such that the sum of their degrees is less than n, then the number of common zeros is divisible by p. The Combinatorial Nullstellensatz (CNSS) of Alon states that if P is an n-variate polynomial over $\mathbb{F}$ whose degree is $d_1 + \cdots + d_n$, and this is certified by the monomial $cx_1^{d_1} \cdots x_n^{d_n}$, for some $c \neq 0$, then in $S_1 \times \cdots \times S_n \subseteq \mathbb{F}^n$ there exists a point where P is not zero, whenever $|S_i| > d_i$, for $i = 1, \ldots, n$. The CNSS has found a wide range of applications among others in graph theory, combinatorics and additive number theory [2, 3].

Over the field $\mathbb{F}_2$ the two theorems greatly simplify via the notion of *multilinear degree*. For any polynomial P over $\mathbb{F}_2$, there exists a unique multilinear polynomial M such that P and M compute the same function on $\mathbb{F}_2^n$. We call the degree of M the multilinear degree of P, denoted as $\mathrm{mdeg}(P)$. We use $\deg(P)$ to denote the usual degree of P. Then the Chevalley-Warning Theorem and the CNSS over $\mathbb{F}_2$ are equivalent to the following statement: *An n-variate $\mathbb{F}_2$-polynomial has an odd number of zeros if and only if its multilinear degree is n.* The natural search problem corresponding to the CNSS therefore is: given an n-variate polynomial P whose multilinear degree is n, find a point a where $P(a) = 1$. Similarly, the search problem corresponding to the Chevalley-Warning Theorem is: given an n-variate polynomial P whose multilinear degree is less than n and a zero of P, find another zero.

Obviously, these problems are not yet well defined algorithmically, since it is not specified, how the polynomial P is given. The starting point of our investigations is the result of Papadimitriou about some instantiation of the Chevalley-Warning Theorem. Specifically, in [22] Papadimitriou considered the following problem. Let the polynomials $P_1, \ldots, P_k$ be given explicitly as sums of monomials, and define $P(x) = 1 + \prod_{i=1}^{k}(P_i(x) + 1)$. We have then $\deg(P) = \sum_{i=1}^{k} \deg(P_i)$, and clearly $P(x) = 0$ if and only if $P_i(x) = 0$, for $i \in [n]$. Suppose that $\deg(P) < n$, and that we are given $a \in \mathbb{F}_2^n$ such that $P(a) = 0$. Then the task is to find $a' \neq a$ such that $P(a') = 0$. We call this problem Explicit Chevalley, and Papadimitriou has shown [22] that it is in PPA.

Could it be that Explicit Chevalley is PPA-complete? We find this highly unlikely. There are two restrictions on the input of Explicit Chevalley. Firstly, the polynomial P is given by an arithmetic circuit (in fact by an arithmetic formula) of specific form. Secondly, and more importantly, the number of variables not only upper bounds the multilinear degree of P, but also the degree of P. The first restriction can be easily relaxed. We can define and compute recursively very easily the circuit degree (also known as the formal degree; see Section 2.3) of the arithmetic circuit which is an upper bound on the degree of the polynomial computed by the circuit. Could it be that the problem, specified by an arithmetic circuit whose circuit degree is less than n, becomes PPA-complete? While this problem might be indeed harder than Explicit Chevalley, we still don't think that it is PPA-complete.

We believe that the more important restriction in Papadimitriou's problem is the one on the degree of the polynomial P computed by the input circuit. As we have seen, to have an even number of zeros, mathematically it is only required that the multilinear degree of P is less than n, so putting the restriction on the degree of P is too stringent. Let's try then to consider instances specified by arithmetic circuits computing polynomials of multilinear degree less than n. However, here we face a serious difficulty. We can't just promise that the polynomial has multilinear degree less than n since PPA is a syntactic class. We must be able to verify syntactically that it is indeed the case.

The multilinear degree of the polynomial is decided by the parity of the monomials computed by the circuit which contain every variable. Let us call such monomials *maximal*. Indeed, the multilinear degree of P is less than n if and only if an even number of maximal

monomials are computed by the circuit. A very general way to prove efficiently that a set is of even cardinality is to give a polynomial Turing machine which computes a perfect matching on the elements of the set. However, the parsing of monomials in arbitrary arithmetic circuits is a rather complex task [19]. For a start, the number of maximal monomials computed by a polynomial size arithmetic circuit can be doubly exponential, making even the description of such a monomial impossible in polynomial time. Fortunately, the situation over the field $\mathbb{F}_2$ simplifies a lot, thanks to cancellations due to certain symmetries. In fact, we are able to show that over $\mathbb{F}_2$ it is sufficient to consider only those monomials which are computed by consistent left/right labellings of the sum gates participating in the computation of the monomial, because the rest of the monomials cancel out. We call such labellings *parse subcircuits*, and we call those parse subcircuits which compute maximal monomials *maximal*. The introduction of parse subcircuits was inspired by the concept of parse trees in [16, 20]. Technically, this results shows that that computing the multilinear degree is in $\oplus$P, the complexity class Parity P.

Is there a chance that for a general circuit computing the multilinear degree is in P? As it turns out not, unless $\oplus\mathrm{P} = \mathrm{P}$, because we can show that computing the multilinear degree is also $\oplus$P-hard. Therefore we have to identify a restricted class of circuits computing polynomials of even multilinear degree which satisfy two properties: the class is on the one hand restricted enough that we are able to construct a polynomial time perfect matching for the maximal parse subcircuits, but it is also large enough that finding another zero for the circuit is PPA-hard. The main contribution of this paper is that we identify such a class of arithmetic circuit which we call PPA-*circuits*.

The definition of these circuits is inspired by a rather straightforward translation of Papadimitriou's basic PPA-problem into a problem for arithmetic circuits. In a nutshell, the basic PPA-problem is the following. Given a degree-one vertex of a graph, in which every vertex has degree at most two, find another degree-one vertex. Here, the graph, whose vertices are the 0-1 strings of given length, is given via a polynomial time Turing machine M determining the neighbourhood of any specified node. We construct an arithmetic circuit over $\mathbb{F}_2$ which, given a vertex v in this graph, computes the opposite parity of the number of v's neighbours. Therefore, finding another degree-one vertex is then just the same as finding another zero of the polynomial computed by the circuit. Most importantly, the circuit is constructed to be in a special form, which allows for a polynomial-time-computable perfect matching over its maximal parse subcircuits. Roughly speaking, from the Turing machine M that describes the neighbours of vertices, we extract two arithmetic circuits D and F that also describe the neighbours in a certain way. We then define the so-called PPA-*composition* of these two circuits, which produces a circuit C that accesses D and F in a black box fashion. Symmetries of the PPA-composition, reflecting the special structure of degree computation, enable us to construct a polynomial-time-computable perfect matching over its maximal parse subcircuits (cf. Lemma 8). Finally we define a PPA-*circuit* as the sum of a PPA-composition and another circuit whose circuit degree is less than n. This is just a minor extension of the family of PPA-compositions since circuits with degree less than n don't have maximal parse subcircuits. The reason for considering this extended family is that this way our result immediately generalizes Papdimitrou's result [22] about Explicit Chevalley, and it makes also easier to express the equivalence between the algorithmic versions of the Chevalley-Warning theorem and the CNSS.

The definition of our problems, PPA-Circuit-CNSS and PPA-Circuit-Chevalley, is therefore the following. In both cases we are given an n-variable, PPA-circuit C over $\mathbb{F}_2$ and an element $a \in \mathbb{F}_2^n$. In the case of PPA-Circuit Chevalley, a is a zero of C, and for

PPA-Circuit CNSS, we consider the sum of the circuits C and L_a, where L_a is a simple *Lagrange-circuit* having a as its only zero and having a single maximal parse subcircuit. The computational task is to compute another zero of C in case of PPA-Circuit Chevalley, and a satisfying assignment for $C + L_a$ in case of PPA-Circuit CNSS. Our result is then stated in the following theorem.

▶ **Theorem 1.** *The problems* PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are* PPA*-complete.*

Since the two problems are easily interreducible, for the proof of Theorem 1 we will show that PPA-Circuit CNSS is PPA-easy and PPA-Circuit Chevalley is PPA-hard. For the easiness part we define a graph, inspired by Papadimitriou's construction, whose vertices are the assignments for the variables and the parse subcircuits. There is an edge between a parse subcircuit and an assignment if the monomial defined by the subcircuit takes the value 1 on the assignment. In addition, we also put an edge between two maximal parse subcircuits of the PPA-composition part of the circuit if they are paired by the perfect matching. As it turns out, the odd degree vertices in this graph are exactly the assignments where the polynomial defined by the circuit is 1, and the unique maximal parse subcircuit of the Lagrange-circuit. Technically, the main part of the proof is to give, for every assignment, a polynomial time computable pairing between its exponentially many neighboring parse subcircuits. For the hardness part (which is much simpler to prove) we express the basic PPA-complete problem as a PPA-composition, as we explained above.

1.3 Previous work

Papadimitriou has proven that Explicit Chevalley is in PPA. Varga [27] has shown the same for the special case of CNSS where the input polynomial P is specified as the sum of a polynomial number of polynomials P_i, where each P_i is the product of explicitly given polynomials whose sum of degrees is at most n. In addition, the input also contains a polynomial time computable matching for all but one of the monomials $x_1 \cdots x_n$ of P. However, the paper doesn't address the question why this doesn't make the problem a promise problem. Concerning the hardness of CNSS, Alon proved in [3] the following result. Let P be specified by an arithmetic circuit in a way that it can be checked efficiently that its multilinear degree is n. If a polynomial time algorithm can find a point a where $P(a) = 1$, then there are no one-way permutations.

1.4 Structure of the paper

In Section 2 we recall the definition of the class PPA, the Combinatorial Nullstellensatz and the Chevalley-Warning Theorem, and arithmetic circuits. In Section 3 we define the parse subcircuits of an arithmetic circuit over $\mathbb{F}_2$, and in Proposition 6 we prove that the polynomial computed by the circuit is the sum of the monomials computed by the parse subcircuits. In Section 4 we define PPA-circuits, and in Lemma 8 we prove that in such circuits a perfect matching for the maximal parse subcircuits can be computed in polynomial time. In Section 5 we state the problems PPA-Circuit CNSS and PPA-Circuit Chevalley over $\mathbb{F}_2$ and observe that they are polynomially interreducible. In Section 6 in Theorem 11 we prove that PPA-Circuit CNSS is in PPA, and in Section 7 in Theorem 13 we prove that PPA-Circuit Chevalley is PPA-hard.

2 Preliminaries

2.1 Total functional NP and the class PPA

We denote the set $\{1, \ldots, n\}$ by $[n]$. A polynomially computable binary relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is called balanced if for some polynomial $p(n)$, for every x and y such that $R(x,y)$ holds, we have $|y| \leq p(|x|)$. Such a relation defines an NP-*search problem* Π_R whose input is x, and the task is to find for inputs x, where $R(x,y)$ holds for some y, such a *solution* y, and report "failure" otherwise. The class FNP of *functional* NP consists of NP-search problems. For two problems Π_R and Π_S in FNP, we say that Π_R is *reducible to* Π_S if there exist two functions f and g computable in polynomial time such that for every positive x, $S(f(x), y)$ implies $R(x, g(x,y))$.

An NP-search problem is *total* if for every x, there exists a solution y. The class of these problems is called TFNP (for Total Functional NP) by Megiddo and Papadimitriou [21]. Problems in TFNP exhibit very interesting complexity properties. An FNP-complete search problem can not be total unless NP = coNP. It is also unlikely that every problem in TFNP could be solved in polynomial time since this would imply P = NP ∩ coNP. TFNP is a semantic complexity class, in the sense that it involves a promise about the totality of the relation R. It is widely believed that such a promise can not be enforced syntactically on a Turing machine, in fact there is no known recursive enumeration of Turing machines that compute total search problems. As usual with semantic complexity classes, TFNP doesn't seem to have complete problems. On the other hand, several syntactically defined subclasses of TFNP with a rich structure of complete problems have been identified along the lines of the mathematical proofs establishing the totality of the defining relation.

The parity argument subclasses of TFNP were defined by Papadimitriou [22, 23]. They can be specified via concrete problems, by closure under reduction. The Leaf problem is defined as follows. The input is a triple (z, M, ω) where z is a binary string and M is the description of a polynomial time Turing machine[1] that defines a graph $G_z = (V_z, E_z)$ as follows. The set of vertices is $V_z = \{0,1\}^{p(|z|)}$ for some polynomial p. For any vertex $v \in V_z$, the machine M outputs on (z, v) a set of at most two vertices. Then, we define G_z as a graph without self-loops, where $\{v, v'\} \in E_z$ for $v \neq v'$, if $v' \in M(z, v)$ and $v \in M(z, v')$. Obviously G_z is an undirected graph where the degree of each vertex is at most 2, and therefore the number of leaves, that is of degree one vertices, is even. Finally $\omega \in V_z$ is a degree one vertex that we call the *standard leaf*. The output of the problem Leaf is a leaf of G_z different from the standard leaf. The Polynomial Parity Argument class PPA is the set of total search problems reducible to Leaf. The directed class PPAD is defined by D-Leaf, the directed analog of Leaf. In the problem D-Leaf the Turing machine defines a directed graph, where the indegree and outdegree of every vertex is at most one. The standard leaf ω is a source, and the output is a sink or source different from the ω.

As shown in [23], the definition of PPA can capture also those problems for which the underlying graph has unbounded degrees and we are seeking for another odd-degree vertex. Specifically, suppose there exists a polynomial time *edge recognition* algorithm $\epsilon(v, v')$, which decides whether $\{v, v'\} \in E_z$. Assume also, that in addition we have a polynomial time *pairing function* $\phi(v, w)$, where by definition, for every vertex v, the function $\phi(v, \cdot)$ satisfies the following properties. For every even degree vertex v, it is a pairing between the vertices adjacent to v, that is for every such vertex w, we have $\phi(v, w) = w'$, where $w' \neq w$, w' is also

[1] The requirement for M to run in polynomial can be imposed by adding a clock.

adjacent to v, and $\phi(v, w') = w$. For odd degree vertices v, we have exactly one adjacent vertex w such that w is mapped to itself, and on the remaining adjacent vertices it is pairing as in the case of an even degree vertex v. The input also contains an odd degree vertex v with a proof for that, in the form of an adjacent vertex w, such that $\phi(v, w) = w$. In [23, Corollary to Theorem 1], Papadimitriou showed that any problem defined in terms of an edge recognition algorithm and a pairing function is in PPA.

2.2 Combinatorial Nullstellensatz and Chevalley-Warning Theorem

Let $\mathbb{F}$ be a field. An *polynomial over* $\mathbb{F}$ (or shortly a polynomial) in n variables is a formal expression $P(x) = P(x_1, \ldots, x_n)$ of the form

$$P(x_1, \ldots, x_n) = \sum_{d_1, \ldots, d_n \geq 0} c_{d_1, \ldots, d_n} x_1^{d_1} \cdots x_n^{d_n},$$

where the coefficients $c_{d_1,\ldots,d_n}$ are from $\mathbb{F}$, and only a finite number of them are different from zero. The *degree* $\deg(P)$ of P is the largest value of $d_1 + \cdots + d_n$ for which the coefficient $c_{d_1,\ldots,d_n}$ is non-zero, where by convention the degree of the zero polynomial is $-\infty$. The ring of polynomials over $\mathbb{F}$ in n variables is denoted by $\mathbb{F}[x_1, \ldots, x_n]$.

Every polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ defines naturally a function from $\mathbb{F}^n$ to $\mathbb{F}$. While over infinite fields this application is one-to-one, this is not true over finite fields where different polynomials might define the same function. For example, over the field $\mathbb{F}_q$ of size q, the polynomial $x^q - x$ is not the zero polynomial (it has degree q), but it computes the zero function.

Numerous results are known about the properties of zero sets of polynomials. The Combinatorial Nullstellensatz of Alon [2] is a higher dimensional extension of the well known fact that a non-zero polynomial of degree d has at most d zeros. It was widely used to prove a variety of results, among others, in combinatorics, graph theory and additive number theory.

▶ **Theorem 2** (Combinatorial Nullstellensatz). *Let $\mathbb{F}$ be a field, let $d_1, \ldots, d_n$ be non-negative integers, and let $P \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial. Suppose that $\deg(P) = \sum_{i=1}^n d_i$, and that the coefficient of $x_1^{d_1} \cdots x_n^{d_n}$ is non-zero. Then for all subsets $S_1, \ldots, S_n$ of $\mathbb{F}$ with $|S_i| > d_i$, for $i = 1, \ldots, n$, there exists $(s_1, \ldots s_n) \in S_1 \times \cdots \times S_n$ such that $P(s_1, \ldots, s_n) \neq 0$.*

The classical result of Chevalley [12] and Warning [28] asserts that if the sum of degrees of some polynomials is less than the number of variables, than the number of their common zeros is divisible by the characteristic of the field.

▶ **Theorem 3** (Chevalley-Warning Theorem). *Let $\mathbb{F}$ be a field of characteristic p, and let $P_1, \ldots, P_k \in \mathbb{F}[x_1, \ldots, x_n]$ be non-zero polynomials. If $\sum_{i=1}^k \deg(P_i) < n$, then the number of common zeros of $P_1, \ldots, P_k$ is divisible by p. In particular, if the polynomials have a common zero, they also have another one.*

Both of these results clearly suggest a computational problem in TFNP: Given a (set of) polynomial(s) satisfying the respective condition of these theorems, find an element in $\mathbb{F}^n$ satisfying the respective conclusion. We study here these problems over the two-element field $\mathbb{F}_2$ where both theorems have a particularly simple form, in fact they become almost the same statement. To see that, let us recall that a *multilinear polynomial* is a polynomial of the form $M(x_1, \ldots, x_n) = \sum_{T \subseteq \{1,\ldots,n\}} c_T x_T$, where x_T stands for the monomial $\prod_{i \in T} x_i$, and the coefficients c_T are elements of $\mathbb{F}_2$. We say that a monomial x_T is *in* M if $c_T = 1$. The degree of a multilinear polynomial M is the cardinality of the largest set T such that x_T is

in M. It is well known that for every polynomial P over $\mathbb{F}_2$, there exists a unique multilinear polynomial $M_P(x_1, \ldots, x_n)$ such that P and M_P compute the same function. We define the *multilinear degree* of a polynomial P over $\mathbb{F}_2$ by $\mathrm{mdeg}(P) = \deg(M_P)$. We call a monomial *maximal* if its multilinear degree is n. Clearly $\mathrm{mdeg}(P) \leq \deg(P)$, and $\mathrm{mdeg}(P) = n$ if and only if the number of maximal monomials of P is odd. Using the notion of multilinear degree, we can now state the rather simple equivalent formulations of the above theorems over $\mathbb{F}_2$.

▶ **Theorem 4** (Combinatorial Nullstellensatz over $\mathbb{F}_2$)**.** *Let $P \in \mathbb{F}_2[x_1, \ldots, x_n]$ be a polynomial such that $\mathrm{mdeg}(P) = n$. Then there exists $a \in \mathbb{F}_2^n$ such that $P(a) = 1$.*

▶ **Theorem 5** (Chevalley-Warning Theorem over $\mathbb{F}_2$)**.** *Let $P \in \mathbb{F}_2[x_1, \ldots, x_n]$ be a polynomial such that $\mathrm{mdeg}(P) < n$, and let $a \in \mathbb{F}_2^n$ such that $P(a) = 0$. Then there exists $b \neq a$ such that $P(b) = 0$.*

2.3 Arithmetic circuits

An n-variable, m-output *arithmetic circuit* C over a field $\mathbb{F}$ is a vertex-labeled, acyclic directed graph whose vertices are called *gates*. It has n *variable gates* of in-degree 0, labeled by the variables $x_1, \ldots, x_n$. There is at most one *constant gate* of in-degree 0, labeled by the constant, for each non-zero field element. The variable and constant gates are called *input gates*. The other gates are of in-degree 2, and are called *computational gates*. They are labeled by $+$ or $\times$, the former are the *sum gates*, and the latter the *product gates*. The number of computational gates of out-degree 0 is m, and they are called the *output gates*.

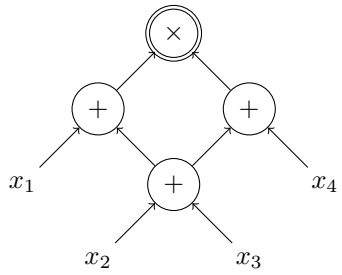


Figure 1 A 4-variable, single-output arithmetic circuit.

For a computational gate g, we distinguish its two children, by specifying the *left* and the *right* child. The left child is denoted by g_ℓ and the right child by g_r. We denote the set of sum gates by G^+, and the set of product gates by $G^\times$. The *size* of C is the number of its gates, and the *depth* of C is the length of the longest path from an input gate to an output gate.

The definition of an arithmetic circuit can be extended naturally to include computational gates of in-degree different from 2. Unary computational gates by definition act as the identity operator. The children of computational gates of in-degree $k > 2$ are distinguished by some some distinct labeling over some set of size k. It is easy to see that such an extended circuit can be simulated by a circuit with binary computational gates, which computes the same polynomial, and has only a polynomial blow-up in size. Our default circuits will be with binary computational gates, and we will mention explicitly when this is not the case.

A *subcircuit* of a circuit C is a subgraph of C which is also a circuit. The *subcircuit rooted at* gate g is the subgraph induced by all vertices contained on some path from the input gates to g, it will be denoted by C_g. The *left subcircuit of* C, denoted by C_ℓ, is the subcircuit

rooted at the left child of the root of C, and the *right subcircuit* C_r is defined similarly. The *composition* of arithmetic circuits is defined in a natural way. If C_1 is an n-variable, m-output circuit and C_2 is a k-variable, n-output circuit then $C_1 \circ C_2$ is the k-variable, m-output circuit composed of C_1 and C_2 where the output gates of C_1 are identified with the variable gates of C_2, and the identical constant gates of the two circuits are also identified. Let C_1 and C_2 be n-variable, single-output arithmetic circuit. The *disjoint sum* $C_1 \oplus C_2$ of C_1 and C_2 is the n-variable, single-output arithmetic circuit whose output gate is a sum gate, its left and right subcircuits are disjoint copies of C_1 and C_2 except for the input gates that C_1 and C_2 share. The disjoint sum naturally generalizes to more than two circuits.

Every gate g in an arithmetic circuit computes an n-variable polynomial $P_g(x)$ in the natural way, which can be defined by recursion on the depth of the gate. An input gate g labeled by $\alpha \in \{x_1, \ldots, x_n\} \cup \mathbb{F}$ computes $P_g = \alpha$. If $g \in G^+$ then $P_g = P_{g_\ell} + P_{g_r}$, if $g \in G^\times$ then $P_g = P_{g_\ell} P_{g_r}$. The polynomial computed by a single-output arithmetic circuit C is the polynomial computed by its output gate, which we will denote by $C(x)$. We define similarly by recursion the *circuit degree* $\mathrm{cdeg}(C)$ of C. If an input gate g is labeled by $\alpha \in \mathbb{F}$ then $\mathrm{cdeg}(C_g) = 0$, and if it is labeled by $\alpha \in \{x_1, \ldots, x_n\}$ then $\mathrm{cdeg}(C_g) = 1$. For computational gates, if $g \in G^+$ then $\mathrm{cdeg}(C_g) = \max\{\mathrm{cdeg}(C_{g_\ell}), \mathrm{cdeg}(C_{g_r})\}$, and if $g \in G^\times$ then $\mathrm{cdeg}(C_g) = \mathrm{cdeg}(C_{g_\ell}) + \mathrm{cdeg}(C_{g_r})$. The circuit degree can be computed in polynomial time, and we clearly have $\deg(C(x)) \leq \mathrm{cdeg}(C)$.

Over the base field $\mathbb{F}_2$, we call an element $a \in \mathbb{F}_2^n$, such that $C(a) = 1$, a *satisfying assignment* for C, and an element a, such that $C(a) = 0$, a *zero* of C. For every $a \in \mathbb{F}_2^n$, we define the *Lagrange-circuit* L_a as $C_1 \times \cdots \times C_n$, where $C_i = x_i$ if $a_i = 1$, and $C_i = x_i + 1$ if $a_i = 0$. Clearly $\mathrm{mdeg}(L_a(x)) = n$, and the only satisfying assignment for L_a is a.

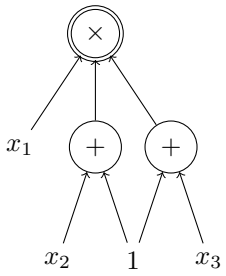


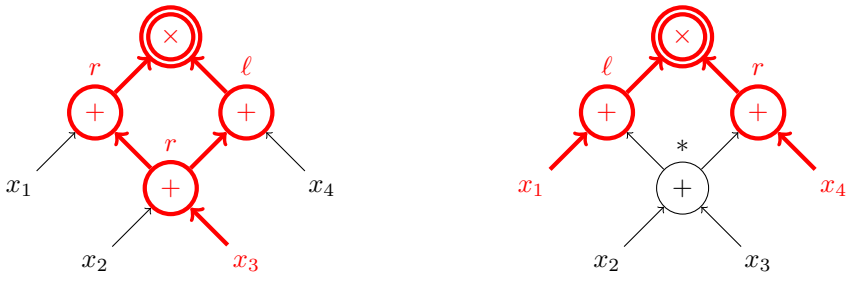
Figure 2 Lagrange-circuit L_{100}.

3 Parse subcircuits

We would like to understand how monomials are computed by a single-output arithmetic circuit C. If g is a sum gate, then the set of monomials computed by C_g is a subset of the union of the set of monomials computed by C_{g_ℓ} and by C_{g_r}. If g is a multiplication gate, then every monomial computed by C_g is the product of a monomial computed by C_{g_ℓ} and a monomial computed by C_{g_r}. A marking of the gates in G^+ from the set $\{\ell, r\}$ therefore computes naturally a monomial of $C(x)$. At first sight it seems that by considering markings restricted to the sum gates effectively participating in the computing of the monomial, we could compute all of them. This is in fact the case when the fanout of every sum gate is one, but this is not true in general circuits since the sum gates can be used several times in the computation of a monomial with possibly inconsistent markings. However, as we show it below, this is essentially true over fields of characteristic 2, where it is sufficient to consider only consistent markings. By doing that, we have to be careful about two things: when

computing a monomial by some marking, we shouldn't mark those sum gates which don't participate in its computation. Indeed, by considering the two possible markings also for irrelevant gates, we would assure that the monomial is necessarily computed an even number of times, making the whole process false. On the other hand, we should mark all the sum gates necessary for the computation of the monomial. We make all this precise by the notion of closed marking and parse subcircuit.

Let C be a single-output arithmetic circuit. A *marking* of C is a partial function $S\colon G^+ \to \{\ell, r\}$, from the sum gates of C to the marks $\{\ell, r\}$. We can equivalently specify a marking by a total function $S^*\colon G^+ \to \{\ell, r, *\}$ where $S^*(g) = *$ if and only if $S(g)$ is undefined. We denote by $\mathrm{Dom}(S)$ the domain of S. For the output gate of C, let S_ℓ be the restriction of S to the sum gates in C_ℓ and let S_r be the restriction of S to the sum gates in C_r. We define $G_S = (V_S, E_S)$, the *accessibility graph* of S by induction on the depth of C. If C is a single vertex then V_S consists of this vertex, and $E_S = \emptyset$. Otherwise, if the output gate is a product gate, then V_S consists of the output gate of C added to $V_{S_\ell} \cup V_{S_r}$, and E_S consists of the two edges from the two children of the output gate to the output gate, added to $E_{S_\ell} \cup E_{S_r}$. If the output gate of C is a sum gate with mark ℓ then V_S consist of the output gate of C added to V_{S_ℓ}, and E_S consists of the edge from the left child of the output gate to the output gate, added to E_{S_ℓ}. The definition in the case when the mark of the output gate is r is analogous. If the output gate of C doesn't have a mark then the accessibility graph is just this single node.



(a) parse subcircuit computing x_3^2 **(b)** parse subcircuit computing x_1x_4

Figure 3 Two parse subcircuits for Figure 1, note that the second one doesn't access all sum gates.

We say that a marking S is *closed* if $\mathrm{Dom}(S) = V_S \cap G^+$, that is if the accessible sum gates of C are exactly those where S is defined. If S is closed then the accessibility graph G_S, with the vertex labels inherited from C, is in fact a subcircuit of C. The inclusion $\mathrm{Dom}(S) \subseteq V_S \cap G^+$ ensures that the only node of out-degree 0 in G_S is the output gate of C, and the inclusion $V_S \cap G^+ \subseteq \mathrm{Dom}(S)$ ensures that the leaves of G_S are leaves in C. We call this subcircuit the *parse subcircuit* induced by S, and denote it by C_S. The set of parse subcircuits of C will be denoted by $\mathcal{S}(C)$. Observe that a parse subcircuit has binary product gates but unary sum gates which act as the identity operator. The polynomial $C_S(x)$ *computed by* the parse subcircuit C_S is therefore a monomial, which we denote by $m_S(x)$. We say that a parse subcircuit C_S is *maximal* if the multilinear degree of $m_S(x)$ is n, that is $m_S(x) = x_1 \cdots x_n$. We say that two parse subcircuits C_S and $C_{S'}$ are *consistent* if for every $g \in \mathrm{Dom}(S) \cap \mathrm{Dom}(S')$, we have $S(g) = S'(g)$.

Clearly, the mapping from closed markings to induced parse subcircuits is a bijection. Therefore, to ease notation, we will often call the closed marking S itself the parse subcircuit, and we will speak about the gates, subcircuits and other circuit related notions of S, instead

of C_S. The notation used for the monomial computed by a parse subcircuit is already consistent with this convention.

▶ **Proposition 6.** *Let C be a single-output arithmetic circuit over a field $\mathbb{F}$ of characteristic 2. Then*

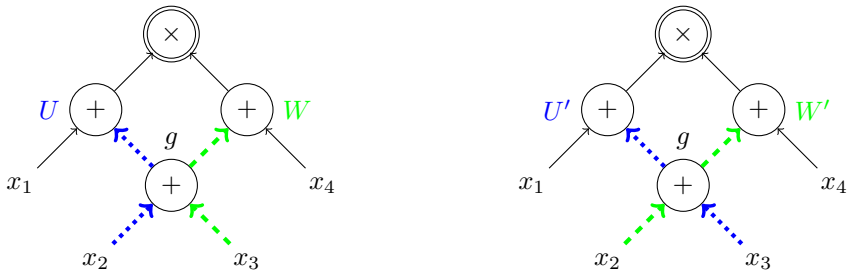
$$C(x) = \sum_{S \in \mathcal{S}(C)} m_S(x).$$

Proof. We prove by induction on the depth of the circuit. If C consists of a single gate, the statement is obvious.

Otherwise, the parse subcircuits of $\mathcal{S}(C_\ell)$ (respectively $\mathcal{S}(C_r)$) are exactly the parse subcircuits of $\mathcal{S}(C)$ restricted to the sum gates of C_ℓ (respectively C_r). When the output gate of C is a sum gate then conversely, $\mathcal{S}(C)$ can be obtained from $\mathcal{S}(C_\ell) \cup \mathcal{S}(C_r)$ by extending the markings in the latter set with the appropriate mark for the root of C. Therefore, using the definitions of $C(x)$ and $m_S(x)$, we get

$$\begin{aligned} C(x) &= C_\ell(x) + C_r(x) \\ &= \sum_{S \in \mathcal{S}(C_\ell)} m_S(x) + \sum_{S \in \mathcal{S}(C_r)} m_S(x) \\ &= \sum_{S \in \mathcal{S}(C),\ S(\text{root})=\ell} m_{S_\ell}(x) + \sum_{S \in \mathcal{S}(C),\ S(\text{root})=r} m_{S_r}(x) \\ &= \sum_{S \in \mathcal{S}(C)} m_S(x), \end{aligned}$$

where the second equality comes from the inductive hypothesis.



(a) inconsistent U, W **(b)** inconsistent U', W'

Figure 4 The involutive pair $(U, W) \leftrightarrow (U', W')$ in the proof of Proposition 6 with $m_U m_W = x_2 x_3 = m_{U'} m_{W'}$ contributes zero to $C(x)$.

When the output gate of C is a product gate, the situation is more complicated. The parse subcircuits S_ℓ and S_r are always consistent for $S \in \mathcal{S}(C)$, but an arbitrary parse subcircuit $U \in \mathcal{S}(C_\ell)$ is not necessarily consistent with an arbitrary parse subcircuit $W \in \mathcal{S}(C_r)$. Therefore the crux of the induction step is to show that the contribution of $m_U(x)m_W(x)$ to $C(x)$ is zero when we sum over all inconsistent U and W. Indeed, we claim that

$$\sum_{(U,W) \in \mathcal{S}(C_\ell) \times \mathcal{S}(C_r),\ U,W \text{inconsistent}} m_U(x) m_W(x) = 0.$$

To prove this, we define an involution $(U, W) \leftrightarrow (U', W')$ over inconsistent pairs in $\mathcal{S}(C_\ell) \times \mathcal{S}(C_r)$ such that $m_U(x)m_W(x) + m_{U'}(x)m_{W'}(x) = 0$. For this let us fix some

topological ordering of the gates in C with respect to the edges of the circuit, and let g be the first sum gate in this ordering where U and W have different marks, say $U(g) = \ell$ and $W(g) = r$. Let the restriction of U to the sum gates of C_g be T_0 and let the restriction of W to the sum gates of C_g be T_1. Both T_0 and T_1 are parse subcircuits in C_g, which are inconsistent only at g. Also, for some monomials $m_0(x)$ and $m_1(x)$, we have $m_U(x) = m_0(x)m_{T_0}(x)$ and $m_W(x) = m_1(x)m_{T_1}(x)$. The parse subcircuit U' is obtained from U by exchanging inside C_g the parse subcircuit T_0 for the parse subcircuit T_1, that is $U' = (U \setminus T_0) \cup T_1$. The parse subcircuit W' is similarly defined from W with the roles of T_0 and T_1 reversed. It follows from the choice of g that U' and W' are parse subcircuits respectively in $\mathcal{S}(C_\ell)$ and $\mathcal{S}(C_r)$ such that the first inconsistency between them in the topological order is at g. Therefore starting the same process with (U', W') we obtain (U, W), and thus the mapping is indeed an involution. Since $m_{U'}(x) = m_0(x)m_{T_1}(x)$ and $m_{W'}(x) = m_1(x)m_{T_0}(x)$, we can conclude that $m_U(x)m_W(x) + m_{U'}(x)m_{W'}(x) = 0$.

We can now complete the induction step for product gates by observing the equalities

$$\begin{aligned}
C(x) &= C_\ell(x) \times C_r(x) \\
&= \left(\sum_{U \in \mathcal{S}(C_\ell)} m_U(x) \right) \times \left(\sum_{W \in \mathcal{S}(C_r)} m_W(x) \right) \\
&= \sum_{(U,W) \in \mathcal{S}(C_\ell) \times \mathcal{S}(C_r),\ U,W \text{consistent}} m_U(x) m_W(x) \\
&= \sum_{S \in \mathcal{S}(C)} m_{S_\ell}(x) m_{S_r}(x) \\
&= \sum_{S \in \mathcal{S}(C)} m_S(x).
\end{aligned}$$

◀

Though it is not directly related to the main result of the paper, we prove here, essentially as a corollary of the previous proposition, that deciding if the polynomial computed by a circuit over the two elements field has maximal multilinear degree is $\oplus$P-complete. Note that by the Chevalley-Warning theorem, the multilinear degree of a circuit is maximal if and only if it has odd number of satisfying assignments, and via this correspondence Proposition 7 can also be proved by using the number of 1's to build a balanced relation. The point of our proof of Proposition 7 is to show this without referring to the Chevalley-Warning theorem, and therefore illustrate the use of maximal parse subcircuits.

▶ **Proposition 7.** *Let C be an n-variable, single-output arithmetic circuit over the field $\mathbb{F}_2$. The problem of deciding if* $\mathrm{mdeg}(C(x)) = n$ *is* $\oplus$P*-complete.*

Proof. For the easiness part, we can define a balanced relation $R(C, S)$ where $S \in \mathcal{S}(C)$, which equals 1 if and only if S is a maximal parse subcircuit. By Proposition 6, we know that the polynomial computed by the circuit C is the sum of all the monomials computed by the parse subcircuits. Among all the parse subcircuits, only the monomials computed by maximal parse subcircuits have degree n. Thus $\mathrm{mdeg}(C(x)) = n$ if and only if there is an odd number of maximal parse subcircuits.

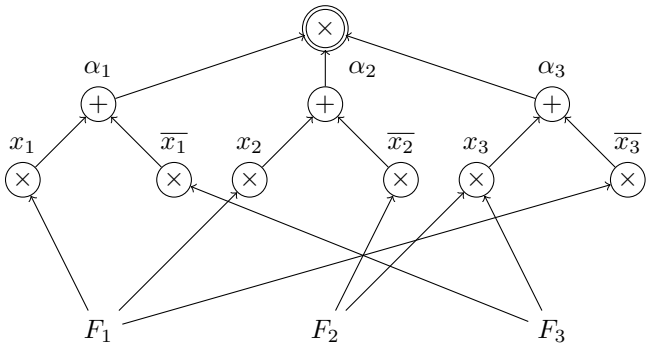


Figure 5 Image of $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3) \wedge (x_3 \vee \overline{x_1})$ by the reduction.

For the hardness part, we will reduce the well known $\oplus$P-complete problem $\oplus$3-SAT [26] to the maximality of $\mathrm{mdeg}(C(x))$. Let $\phi = \{F_1, F_2, \ldots, F_m\}$ be an instance of 3-SAT, where the clause F_i is the conjunction of three literals belonging to $\{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$. The reduction maps ϕ to an m-variable, single-output and depth-3 arithmetic circuit C defined as follows. The output gate at level 0 is a product gate. It has n children $\alpha_1, \ldots, \alpha_n$, all plus gates, which compose the first level of the circuit. At level 2, for all $1 \leq j \leq n$, the gate α_j has two children x_j and $\overline{x_j}$, which are product gates. The gate x_j is the left child of α_j, and $\overline{x_j}$ is its right child. Finally at level 3 are the m variable gates $F_1, \ldots, F_m$, such that F_i is a child of $y \in \{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$ if $y \in F_i$ in ϕ. The following is an illustration of the circuit which is the image of the formula $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_2} \vee x_3) \wedge (x_3 \vee \overline{x_1})$ by the reduction.

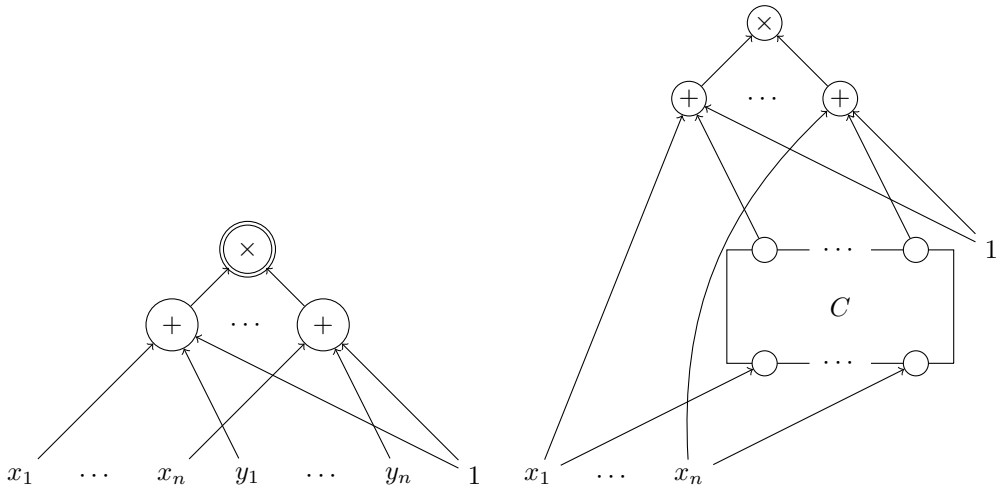
We give a one-to-one mapping S from the assignments of ϕ to the parse subcircuits of $\mathcal{S}(C)$. Since all plus gates of C are reachable from the output gate, a parse subcircuit of C is an $\{\ell, r\}$-marking of the gates $\alpha_1, \ldots, \alpha_n$. The parse subcircuits are therefore naturally identified with the elements of $\{\ell, r\}^n$. For an assignment $x \in \{0,1\}^n$, the map S is defined as

$$S(x)_i = \begin{cases} \ell & \text{if } x_i = 1 \\ r & \text{if } x_i = 0. \end{cases}$$

To finish the proof we show that x is a satisfying assignment if and only if $S(x)$ is a maximal parse subcircuit. To see that, observe that x is a satisfying assignment if and only if each F_i in ϕ contains a true literal. By the definition of S, the clause F_i contains a true literal exactly when the variable F_i of C is in the parse subcircuit $C_{S(x)}$. Since $C_{S(x)}$ is maximal if and only if F_i is in the parse subcircuit $C_{S(x)}$ for all i, this concludes the proof. ◀

4 PPA-circuits

Given an arbitrary circuit C and a satisfying assignment, asking for another satisfying assignment would be an NP-hard problem. We want to restrict the form of the circuit C in a way which takes into consideration the structure of problems in PPA.



(a) The arithmetic circuit I. **(b)** The compound arithmetic circuit $I \diamond C$.

Figure 6 The arithmetic circuits I and $I \diamond C$.

For this, we use repeatedly a $2n$-variable, single-output arithmetic circuit I. The circuit I is of depth 2, its output gate is a product gate with n children, all sum gates. Every sum gate has 3 children, the left child of the ith gate is the variable gate x_i, its center child is the variable gate y_i, and its right child is the constant gate 1. For an n-variable, n-output circuit C, we define $I \diamond C$, the *diamond composition* of I with C, as the n-variable, single-output circuit composed from a circuit I at the top and C below. More precisely, the variable gates of $I \diamond C$ labeled by $x_1, \ldots, x_n$ are also the first n variables of I, and the variable gates $y_1, \ldots, y_n$ of I are identified with the output gates of C. If C has also a constant gate 1, it is identified with the constant gate 1 of I.

The polynomial computed by the circuit I is $I(x_1, \ldots, x_n, y_1, \ldots, y_n) = \prod_{i=1}^{n}(x_i + y_i + 1)$. It is easy to check that $I(x, y)$ is 1 if and only if the two n-bit strings $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ are equal. Therefore $I \diamond C(x) = 1$ if and only if $C(x) = x$.

Given two n-variable, n-output arithmetic circuits D and F, we consider the set of six n-variable, single-output circuits

$$\mathcal{C}_{D,F} = \{I_1 \diamond D_1 \circ F_1, I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4, I_4 \diamond D_5, I_5 \diamond F_3 \circ F_4, I_6 \diamond F_5\},$$

where $I_1, \ldots, I_6$ are copies of I; $D_1, \ldots, D_5$ are copies of D; $F_1, \ldots, F_5$ are copies of F, and the six circuits share the same input gates. The PPA-*composition* of D and F is the n-variable, single-output circuit $C_{D,F}$ is the disjoint sum of the six circuits in $\mathcal{C}_{D,F}$. We call the circuits in $\mathcal{C}_{D,F}$ the *components* of $C_{D,F}$. The polynomial computed by $C_{D,F}$ is

$$\begin{aligned} C_{D,F}(x) = {} & I(x, D(F(x))) + I(x, F(D(x))) + I(x, D(D(x))) \\ & + I(x, D(x))) + I(x, F(F(x))) + I(x, F(x))). \end{aligned}$$

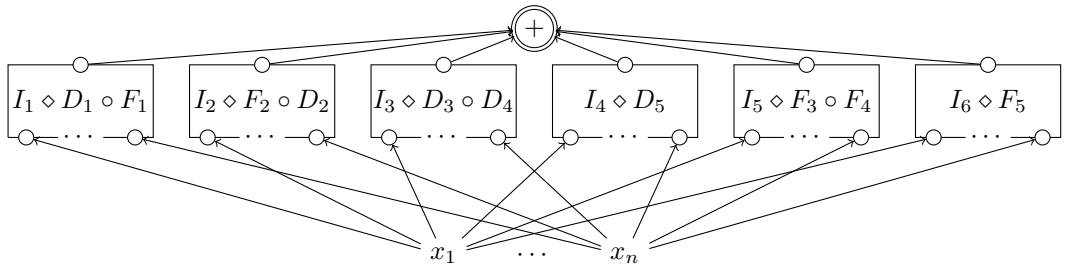


Figure 7 The circuit $C_{D,F}$, the PPA-composition of the circuits D and F.

The main structural property of a PPA-composition C is that it computes a polynomial whose multilinear degree is less than n. Moreover, a witness for that can be computed in polynomial time. By Proposition 6, the multilinear degree of $C(x)$ is determined by the parity of its maximal parse subcircuits, $\mathrm{mdeg}(C(x)) = n$ if and only if the parity of the maximal parse subcircuits is odd. Thus, the multilinear degree of $C(x)$ can be certified by a special type of syntactically defined matching over its maximal parse subcircuits. Formally, a *matching for maximal parse subcircuits in* C is a polynomial time Turing machine μ which defines a matching over the maximal parse subcircuits of C as follows: S and S' are *matched* if $\mu(C, S) = S'$ and $\mu(C, S') = S$. If μ defines a perfect matching between the maximal parse subcircuits, then $\mathrm{mdeg}(C(x)) < n$. If μ defines a perfect matching *outside* some maximal parse subcircuit T, meaning that T is the only maximal parse subcircuit without a matching pair in μ, then $\mathrm{mdeg}(C(x)) = n$.

All the above statements hold also for circuits which are the direct sum of a PPA-composition and another circuit which certifiably has no maximal parse subcircuit. This is obviously the case of circuits which compute polynomials of degree less than n. Our final set of authorized circuits are of this form. We say that a circuit C is a PPA-*circuit* if for some D and F, we have $C = C_{D,F} \oplus C'$, where $\mathrm{mdeg}(C') < n$.

▶ **Lemma 8.** *If C is a* PPA*-circuit then* $\mathrm{mdeg}(C(x)) < n$*, and a perfect matching μ between the maximal parse subcircuits of C can be computed in polynomial time.*

Proof. Let $C = C_{D,F} \oplus C'$ where $\mathrm{mdeg}(C') < n$. We can suppose without less of generality that C' is the empty circuit, that is $C = C_{D,F}$. Since the six components of C are pairwise disjoint (except for the input gates), every maximal parse subcircuit in C consists of the mark of the root of C from the set $\{1, \ldots, 6\}$, and a maximal parse subcircuit in the corresponding component. For the definition of μ we decompose C into the disjoint sum of three circuits C_1, C_2 and C_3 where each of them is the disjoint sum of two PPA-components, and will define the matching inside each of these circuits. The three circuits are as follows: $C_1 = I_1 \diamond D_1 \circ F_1 \oplus I_2 \diamond F_2 \circ D_2$, $C_2 = I_3 \diamond D_3 \circ D_4 \oplus I_4 \diamond D_5$, and $C_3 = I_5 \diamond F_3 \circ F_4 \oplus I_6 \diamond F_5$. Clearly C_2 and C_3 are similar, therefore it is sufficient to define μ for C_1 and C_2.

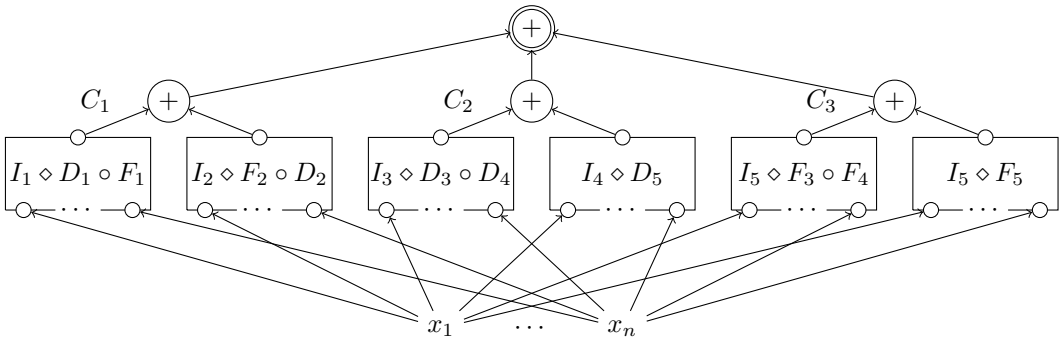


Figure 8 The decomposition $C = C_1 \oplus C_2 \oplus C_3$.

To ease the notation, we rename the subcircuits of C_1 as $I \diamond D \circ F$ and $I' \diamond F' \circ D'$, and we suppose that $I \diamond D \circ F$ is the left subcircuit of C_1 and $I' \diamond F' \circ D'$ is its right subcircuit. Let us denote the output (sum) gate of C_1 by h, the sum gates of I by $h_1, \ldots, h_n$, the output gates of D by $d_1, \ldots d_n$, and the output gates of F by $f_1, \ldots, f_n$. For every gate g in I, D and F, we denote the corresponding gate in I', D' and F' by g', and we also set $h' = h$. Let us recall the h_i has three children, the left child is the input gate x_i, the center child is d_i, the ith output gate of D, and its right child is the constant gate 1. A parse subcircuit can map h_i into one of the three marks ℓ, c and r, corresponding respectively to its left, center, and right child.

We define $\mu(S)$ for the maximal parse subcircuits of $I \diamond D \circ F$, that is when $S(h) = \ell$. The definition for the case $S(h) = r$ is symmetric. Let us first define three sets of indices $S_{\text{out}}, S_{\text{middle}}, S_{\text{in}} \subseteq [n]$. Let $S_{\text{out}} = \{i \in [n] : S(h_i) = c\}$, that is S_{out} contains those indices i for which the edge from the d_i to h_i belongs to S. By definition $i \in S_{\text{middle}}$ if there exists an edge in S from f_i to a gate in D. Finally, $i \in S_{\text{in}}$ if there exists an edge in S from x_i to a gate in F. We claim that $S_{\text{out}} \subseteq S_{\text{in}}$. This is indeed true, since if there exists $i \in S_{\text{out}} \setminus S_{\text{in}}$ then the monomial $m_S(x)$ wouldn't contain the variable x_i, contradicting its maximality. We are now ready to define $S' = \mu(S)$ by distinguishing two cases, depending on if S_{out} is a proper subset of S_{in} or not.

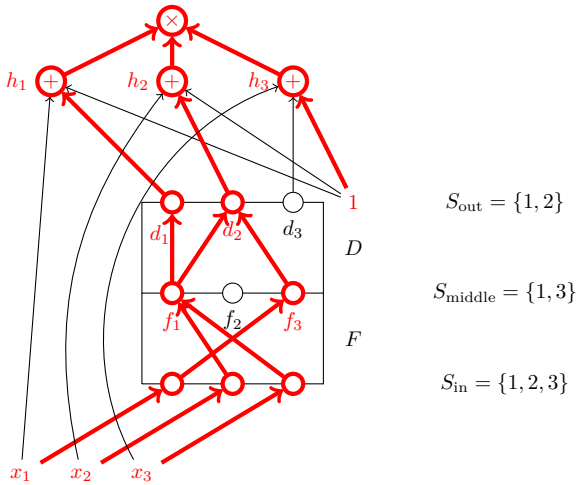
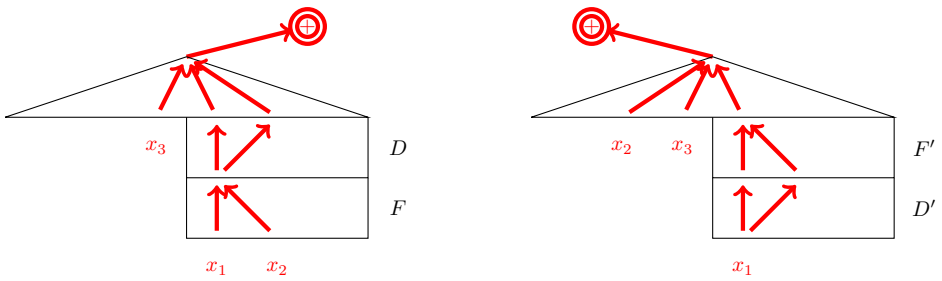


Figure 9 The left subcircuit $I \diamond D \circ F$ of C_1 and the index sets $S_{\text{in}}, S_{\text{middle}}$ and S_{out}.

Case 1: $S_{\text{out}} \subset S_{\text{in}}$. Let i be the smallest index in $S_{\text{in}} \setminus S_{\text{out}}$. By definition, we let S' be the same as S, except on h_i, where S' takes the mark r when $S(h_i) = \ell$, and it takes the mark ℓ when $S(h_i) = r$. This means that the only difference between S and S' is that at the ith sum gate of I, one subcircuit contains the edge from x_i to h_i, whereas the other contains the edge from 1 to h_i. S' is therefore a parse subcircuit. To show that S' is also maximal, the interesting case is when $S(h_i) = \ell$ and $S'(h_i) = r$, that is $m_{S'}(x)$ doesn't directly pick up x_i at h_i. But since $i \in S_{\text{in}}$, the variable x_i is still in S', which is therefore maximal. Finally clearly $\mu(S') = S$.

(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 10 Case 1 of the matching μ for C_1 where i is the smallest index in $S_{\text{in}} \setminus S_{\text{out}}$.



(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 11 Case 2 of the matching μ for C_1: $S_{\text{out}} = S_{\text{in}}$.

Case 2: $S_{\text{out}} = S_{\text{in}}$. In that case first observe that for every index $i \notin S_{\text{out}}$, we have $S(h_i) = \ell$, that is S contains the edge (x_i, h_i), since otherwise $m_S(x)$ wouldn't contain x_i. By definition, let $\text{Dom}(S') = \{g' \in G^+ : g \in \text{Dom}(S)\}$. For the output gate $h' = h$ of C_1 we set $S'(h') = r$, that is S' will be a parse subcircuit of $I' \diamond D' \circ F'$. For the sum gates $h'_1, \ldots, h'_n$ of I, we set $S'(h'_i) = c$ if $i \in S_{\text{middle}}$, and we set $S'(h'_i) = \ell$ otherwise. Finally, for every sum gate $g \in \text{Dom}(S)$ in D or in F, we set $S'(g') = S(g)$.

Let us recall that V_S is the set of vertices of the accessibility graph G_S of S. The proof that S' is a maximal parse subcircuit immediately follows from the following proposition.

▶ **Proposition 9.** *For every computational gate g in $I \diamond D \circ F$, we have*

$$g \in V_S \quad \textit{if and only if} \quad g' \in V_{S'}.$$

Proof. We show the implication from left to right. This is certainly true for the computational gates of I since they are all accessible in G_S, as well as the computational gates of I' in $G_{S'}$.

If $g \in V_S$ is a computational gate of D then there is a path p in G_S from g to h which can be decomposed into $p = p_1 p_2$, where p_1 goes from g to d_i for some $i \in S_{\text{out}}$, and p_2 is the path from d_i to h. In $G_{S'}$ we have therefore a path p'_1 from g' to d'_i. Since $S_{\text{out}} = S_{\text{in}}$, in G_S

we have a path p_3 from x_i to f_j for some $j \in S_{\text{middle}}$. Therefore in $G_{S'}$ there exists a path p'_2 from d'_i to f'_j. Finally, in $G_{S'}$ there is also a path p'_3 from f'_j to h' because $j \in S_{\text{middle}}$. Then $p' = p'_1 p'_2 p'_3$ is a path from g' to h'.

If $g \in V_S$ is a computational gate of F then there is a path p in G_S from g to h which can be decomposed into $p = p_1 p_2 p_3$, where p_1 goes from g to d_i for some $i \in S_{\text{middle}}$, p_2 goes from d_i to f_j for some $j \in S_{\text{out}}$, and p_3 is the path from f_j to h. Then in $G_{S'}$ there exists a path p'_1 from g' to d'_i, and a path p'_2 which goes from d'_i to h' since $i \in S_{\text{middle}}$. Then the path $p' = p'_1 p'_2$ goes from g' to h'.

The implication from right to left follows from the symmetry between S and S'. For this, it is useful to observe that $S'_{\text{out}} = S'_{\text{in}} = S_{\text{middle}}$, and $S'_{\text{middle}} = S_{\text{out}} = S_{\text{in}}$. ◀

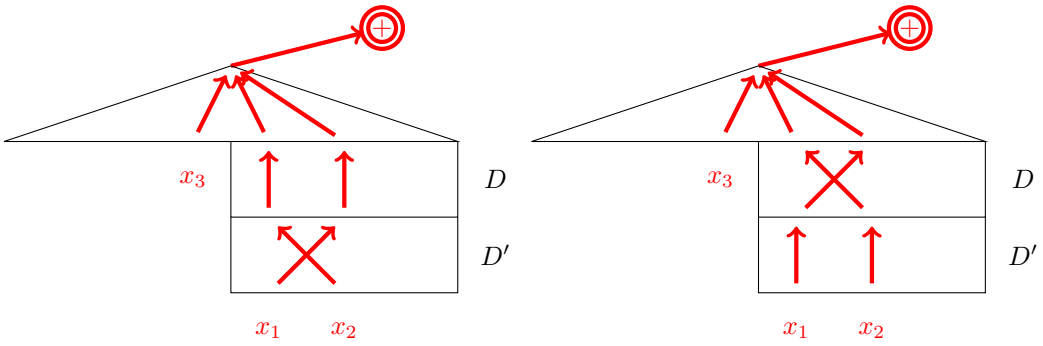
We have $\text{Dom}(S) = V_S \cap G^+$ since S is a parse subcircuit. Proposition 9 and the definition $\text{Dom}(S') = \{g' \in G^+ : g \in \text{Dom}(S)\}$ imply that $\text{Dom}(S') = V_{S'} \cap G^+$, and therefore S' is a parse subcircuit. To prove the maximality of S' let us show that every input gate is in $V_{S'}$. If $i \in S_{\text{middle}}$ then the path p defined above for the computational gates in D yields a path p' from x_i to h'. If $i \notin S_{\text{middle}}$ then the direct path p' from x_i to h' via h'_i exists in $G_{S'}$. Finally μ is clearly involutive in that case too.

We now turn to the description of μ for C_2, where we rename its two subcircuits as $I \diamond D \circ D'$ and $I^* \diamond D^*$. The matching for C_2 has strong analogies with the matching for C_1, to better see this we also use the names I', F and F' respectively for the circuits I, D' and D. This means that $I \diamond D \circ F$ and $I' \diamond F' \circ D'$ are just different names for the circuit $I \diamond D \circ D'$. We suppose that $I \diamond D \circ D'$ is the left subcircuit of C_2 and $I^* \diamond D^*$ is its right subcircuit. Similarly to the circuit C_1, we denote the output gate of C_2 by h, the sum gates of I by $h_1, \ldots, h_n$, the ouput gates of D by $d_1, \ldots d_n$, and the output gates of D' by $d'_1, \ldots, d'_n$. For every gate g in I, D and D', we denote the corresponding gate respectively in I', D' and D by g'. For every gate g in I and D, we denote the corresponding gate in I^* and D^* by g^*. We also set $h^* = h' = h$. Again, h_i has three children, the left child is the input gate x_i, the center child is d_i, the right child is the constant gate 1, and the respective marks are ℓ, c and r.

We first describe $S' = \mu(S)$ when S is a maximal parse subcircuit of $I \diamond D \circ D'$. We define $S_{\text{out}}, S_{\text{middle}}, S_{\text{in}}$ the same way as for the circuit $I \diamond D \circ F$, keeping in mind that $F = D'$. As before, we have $S_{\text{out}} \subseteq S_{\text{in}}$. For the definition of μ we now distinguish three cases.

Case 1: $S_{\text{out}} \subset S_{\text{in}}$. The definition of S' is identical to the first case of the definition of the matching for C_1.

Case 2: $S_{\text{out}} = S_{\text{in}}$ and there exists a sum gate g in D such that $S(g) \neq S(g')$. The definition of S' is identical to the second case of the definition of the matching for C_1, with one exception. The difference is that S' remains in the left subcircuit of C_2, that is for the output gate $h' = h$ we set $S'(h') = \ell$.

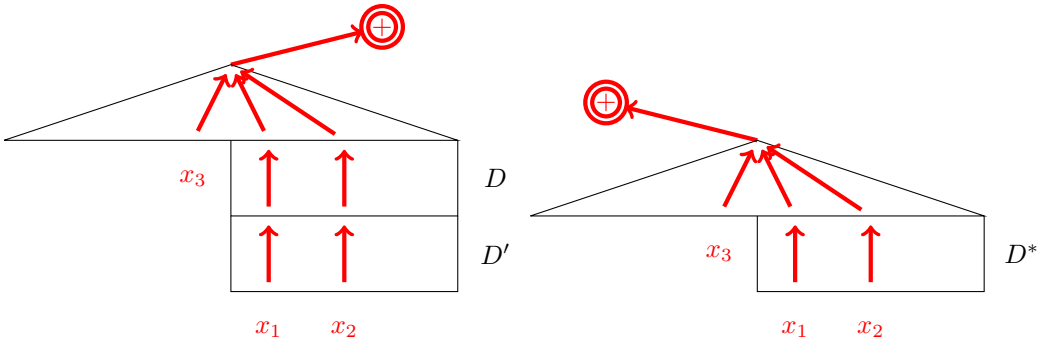


(a) maximal parse subcircuit S **(b)** maximal parse circuit S'

Figure 12 Case 2 of the matching μ for C_2: $S_{\text{out}} = S_{\text{in}}$ and $\exists g, S(g) \neq S(g')$.

Case 3: $S_{\text{out}} = S_{\text{in}}$ and for all sum gate g in D, we have $S(g) = S(g')$. By definition we set $\text{Dom}(S') = \{g^* \in G^+ : g \in \text{Dom}(S)\}$. For the output gate $h^* = h$ of C_2 we set $S'(h^*) = r$, that is S' will be a parse subcircuit of $I^* \diamond D^*$. For every other sum gate $g \in \text{Dom}(S)$, we set $S'(g^*) = S(g)$.

The description $S' = \mu(S)$ when S is a maximal parse subcircuit of $I^* \diamond D^*$ is as follows. By definition we set $\text{Dom}(S') = \{g, g' \in G^+ : g^* \in \text{Dom}(S)\}$. We set $S'(h) = \ell$, that is S' is a parse subcircuit of $I \diamond D \circ D'$. For the sum gates of I, we set $S'(h_i) = S(h_i^*)$. For every sum gate $g^* \in \text{Dom}(S)$ which is in D^*, we set $S'(g) = S'(g') = S(g^*)$.



(a) maximal parse subcircuit S **(b)** maximal parse subcircuit S'

Figure 13 Case 3 of the matching μ for C_2: $S_{\text{out}} = S_{\text{in}}$ and $\forall g, S(g) = S(g')$.

The proof that S' is a maximal parse subcircuit is basically the same as for the case of circuit C_1. It follows immediately from the definition that μ is an involution. The only additional point to see is that in the second case $S' \neq S$ because $S(g) \neq S(g')$, for some gate g in D. ◀

5 The computational problems

We are now ready to define PPA-Circuit CNSS and PPA-Circuit Chevalley, the two computational problems corresponding to the CNSS and to the Chevalley-Warning theorem over $\mathbb{F}_2$. The input will be in both cases an n-variable, single-output PPA-circuit C, and an element $a \in \mathbb{F}_2^n$. In the case of PPA-Circuit Chevalley, it is a zero of C, and

Lemma 8 ensures that C satisfies the hypotheses of the Chevalley-Warning Theorem. For PPA-Circuit CNSS, we consider the circuit $C \oplus L_a$, and Lemma 8 again ensures that this circuit satisfies the hypothesis of the CNSS. The computational task is to compute $b \in \mathbb{F}_2^n$ whose existence is stipulated by these theorems.

The definition of the two problems is the following.

PPA-Circuit Chevalley
Input: (C, a), where C is an n-variable PPA-circuit over $\mathbb{F}_2$, and a is a zero of C.
Output: Another zero $b \neq a$ of C.

PPA-Circuit CNSS
Input: (C', a), where C' is an n-variable PPA-circuit over $\mathbb{F}_2$, and $a \in \mathbb{F}_2^n$.
Output: An element $b \in \mathbb{F}_2^n$ satisfying $C = C' \oplus L_a$.

Let us restate here our main theorem.

▶ **Theorem 1** (restated). *The problems* PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are* PPA*-complete.*

Proof. In Proposition 10 below we show that PPA-Circuit CNSS and PPA-Circuit Chevalley are polynomially interreducible. In Theorem 11 in Section 6 we prove that PPA-Circuit CNSS is in PPA, and in Theorem 13 in Section 7 we prove that PPA-Circuit Chevalley is PPA-hard. ◀

We now turn to the proof of the various parts of Theorem 1.

▶ **Proposition 10.** PPA-Circuit CNSS *and* PPA-Circuit Chevalley *are polynomially equivalent.*

Proof. First we reduce PPA-Circuit CNSS to PPA-Circuit Chevalley. Let (C', a) be an instance of PPA-Circuit CNSS, and set $C = C' \oplus L_a$. We can suppose that $C'(a) = 1$, since otherwise we are done. We define the circuit $C'' = C \oplus 1$. Then clearly C'' is a PPA-circuit, and $C''(a) = 0$. The result of the reduction is then the input (C'', a) to PPA-Circuit Chevalley. If the solution to that input is another zero $b \neq a$ of $C''(x)$, then clearly $C(b) = 1$.

The reduction from PPA-Circuit Chevalley to PPA-Circuit CNSS is very similar. Let (C, a) be an instance of PPA-Circuit Chevalley. We set $C' = C \oplus 1$, and $C'' = C' \oplus L_a$. Clearly C' is a PPA-circuit. The result of the reduction is (C', a). If the solution to that input is a satisfying assignment $C''(b) = 1$ then b is a zero of C. Also, $b \neq a$ since $C''(a) = 0$, therefore b is another zero of C. ◀

6 PPA-easiness

▶ **Theorem 11.** PPA-Circuit CNSS *is in* PPA.

Proof. We will give a reduction from PPA-Circuit CNSS to Leaf. Given an input $N = (C', a)$ to PPA-Circuit CNSS, we set $C = C' \oplus L_a$. We construct a graph $G_N = (V_N, E_N)$ by a polynomial time edge recognition algorithm and a polynomial time pairing function ϕ as explained in Section 2.1. The vertices of G_N are $V_N = \mathbb{F}_2^n \cup \mathcal{S}(C)$.

There are two types of edges in E_N, the first type is between an assignment and a parse subcircuit, and the second type is between two maximal parse subcircuits. By definition, the

edge $\{a, S\}$ exists between $a \in \mathbb{F}_2^n$ and $S \in \mathcal{S}(C)$ if $m_S(a) = 1$. Such an edge can be easily recognized since the monomial $m_S(x)$ can be evaluated in linear time in the size of C.

Since C is the disjoint sum of C' and L_a, the maximal parse subcircuits of C are the maximal parse subcircuits of C' extended with the appropriate mark at the output gate, and the unique maximal parse subcircuit of L_a, again extended with the appropriate mark at the output gate. Let us denote the latter parse subcircuit by T. Let μ be a polynomial time computable perfect matching between the maximal parse subcircuits of C', which exists by Lemma 8. By definition, the edge $\{S, S'\}$ exists between $S, S' \in \mathcal{S}(C')$ if both are extensions of maximal parse subcircuits of C', and their restrictions to C' are matched by μ.

Observe that by Proposition 6, a vertex $a \in \mathbb{F}_2^n$ has odd degree if and only if $C(a) = 1$. If S is a maximal parse subcircuit then among the vertices in $\mathbb{F}_2^n$ it is only connected to 1^n. If $S \neq T$, then it has one more neighbor, its matching pair given by μ, and therefore its degree is two. On the other hand, the degree of T is one and therefore it is odd. We can therefore take T as the standard leaf.

We first give the pairing for the vertices in $\mathcal{S}(C)$. We fix $S \in \mathcal{S}(C)$, and let $a \in \mathbb{F}_2^n$ such that $m_S(a) = 1$. If S is not a maximal parse subcircuit then let $i \in [n]$ be the smallest integer such that x_i is not in $m_S(x)$, and let a' be obtained from a by flipping the ith bit. Then by definition $\phi(S, \cdot)$ pairs a with a'. If $S \neq T$ is a maximal parse subcircuit then it has two neighbors: its matching pair S' by μ and 1^n, and $\phi(S, \cdot)$ pairs these two neighbors. For every S, the mapping $\phi(S, \cdot)$ is clearly involutive.

We now turn to the more complicated pairing for the vertices in $\mathbb{F}_2^n$. Observe that this depends only on the edges of the first type, that is edges between an assignment $a \in \mathbb{F}_2^n$ and a parse subcircuit $S \in \mathcal{S}(C)$. These edges can be defined actually for an arbitrary circuit C. Let us denote by $G(C)$ the graph with vertex set $\mathbb{F}_2^n \cup \mathcal{S}(C)$ and with edges of the first type from G_N. First we prove the following lemma about $G(C)$ on induction of the size of C.

▶ **Lemma 12.** *For every n-variable, single-output circuit C, and for every vertex $a \in \mathbb{F}_2^n$ in $G(C)$,*

(a) *if $\deg(a)$ is even then for all $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$, there exists $g \in \mathrm{Dom}(S)$ with $P_g(a) = 0$,*

(b) *if $\deg(a)$ is odd then there exists a unique $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S)$.*

Proof. If C consists of a single node, the statement is obviously true. Otherwise we first handle a). When $\deg(a)$ is even then $C(a) = 0$. If the root is a sum gate then we are done since it is in the domain of every parse subcircuit. If the root is a product gate then at least one of its children (say the left without loss of generality) also evaluates to 0, that is $C_\ell(a) = 0$. Let $S \in \mathcal{S}(C)$ be such that $m_S(a) = 1$, then we also have $m_{S_\ell}(a) = 1$. By the inductive hypothesis there exists $g \in \mathrm{Dom}(S_\ell)$ with $P_g(a) = 0$, and since g is also in the domain of S, we are again done.

We now deal with the induction step of b). When $\deg(a)$ is odd then $C(a) = 1$. If the root is a sum gate then one of its children evaluates to 0, and the other one to 1, say $C_\ell(a) = 0$ and $C_r(a) = 1$. By the inductive hypothesis there exists a unique $S' \in \mathcal{S}(C_r)$ such that $m_{S'}(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S')$. On the other hand, if $S \in \mathcal{S}(C)$ such that $m_S(a) = 1$ and the mark of S at the root is ℓ, then $S_\ell \in \mathcal{S}(C_\ell)$ and $m_{S_\ell}(a) = 1$, and by a) there exists $g \in \mathrm{Dom}(S)$ with $P_g(a) = 0$. Therefore the unique S satisfying the hypothesis of the statement is S' extended with the mark r at the root.

To finish the induction step for b), let us suppose now that the root of C is a product gate. Then by the inductive hypothesis there exists a unique $S' \in \mathcal{S}(C_\ell)$ such that $m_{S'}(a) = 1$,

and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S')$, and similarly there exists a unique $S'' \in \mathcal{S}(C_r)$ such that $m_{S''}(a) = 1$, and $P_g(a) = 1$ for all $g \in \mathrm{Dom}(S'')$. We claim that S' and S'' are compatible, and therefore their union $S = S' \cup S''$ is the unique parse subcircuit of C satisfying the claim. Suppose that it is not the case, that is there exists $g \in \mathrm{Dom}(S') \cap \mathrm{Dom}(S'')$ such that $S'(g) \neq S''(g)$. Since $P_g(a) = 1$, for one of its children, say for g_ℓ, we have $P_{g_\ell}(a) = 0$, contradicting the inductive hypothesis about the parse subcircuit in $\{S', S''\}$ which takes the value ℓ in g. ◀

We give now the pairing $\phi(a, \cdot)$ for $a \in \mathbb{F}_2^n$. If $\deg(a)$ is even then let $S \in \mathcal{S}(C)$ be such that $m_S(a) = 1$. By Lemma 12 there exists a sum gate in the domain of S where P evaluates to 0. Let g be in some topological ordering of the gates of C the first sum gate such that $P_g(a) = 0$, and suppose without loss of generality that $S(g) = \ell$. Let $Z \in \mathcal{S}(C_g)$ be the restriction of S to C_g, and we obviously have $m_Z(a) = m_{Z_\ell}(a) = 1$. We claim that $P_{g_\ell}(a) = P_{g_r}(a) = 1$. Indeed, if $P_{g_\ell}(a) = P_{g_r}(a) = 0$, then by Lemma 12, applied to C_{g_ℓ}, there exists $g' \in \mathrm{Dom}(Z_\ell)$ with $P_{g'}(a) = 0$, which contradicts the choice of g. Therefore again by Lemma 12 there exists a unique $Z'' \in \mathcal{S}(C_{g_r})$ such that $m_{Z''}(a) = 1$, and $P_h(a) = 1$ for all $h \in \mathrm{Dom}(Z'')$. We let $Z' \in \mathcal{S}(C_g)$ be the extension of Z'' with $Z'(g) = r$. Finally we define $\phi(a, S)$ as the parse subcircuit S' obtained from S by exchanging Z with Z', that is $S' = (S \setminus Z) \cup Z'$. It is clear that $m_{S'}(a) = 1$, and $\phi(a, S') = S$.

If $\deg(a)$ is odd then by Lemma 12 there exists a unique parse subcircuit S such that $m_S(a) = 1$, and $P_g(a) = 1$, for all $g \in \mathrm{Dom}(S)$. We set $\phi(a, S) = S$. For all parse subcircuits S such that $P_g(a) = 0$, for some $g \in \mathrm{Dom}(S)$, the construction of $S' = \phi(a, S)$ is identical to the previous case.

The finish the proof, observe that the vertices of odd degree in V_N other than the standard leaf T are the elements $a \in \mathbb{F}_2^n$ such that $C(a) = 1$. Therefore the output of the reduction is a satisfying assignment a for C. ◀

7 PPA-hardness

▶ **Theorem 13.** PPA-Circuit Chevalley *is* PPA*-hard.*

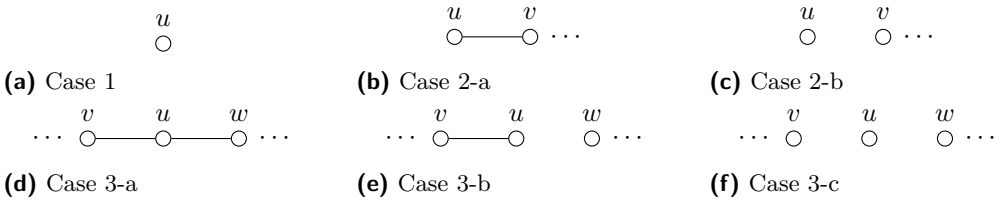
Proof. We will reduce Leaf to PPA-Circuit Chevalley. Let (z, M, ω) be an instance of Leaf, where M defines the graph $G_z = (V_z, E_z)$ with $V_z = \{0,1\}^n$, for some polynomial function n of $|z|$, and ω is the standard leaf in G_z. We know that for every vertex u, $M(z, u)$ is a set of at most two vertices. Composing the standard simulation of polynomial time Turing machines by polynomial size boolean circuits [24] with the obvious simulation of boolean circuits by arithmetic circuits, there exist two n-variables, n-output polynomial size arithmetic circuits D and F with the following properties:

- if $M(z, u) = \emptyset$ or $M(z, u) = \{u\}$ then $D(u) = F(u) = u$,
- if $M(z, u) = \{v\}$ or $M(z, u) = \{v, u\}$ with $v \neq u$ then $D(u) = v$ and $F(u) = u$,
- if $M(z, u) = \{v, w\}$ with $v \neq u \neq w$ then $D(u) = v$ and $F(u) = w$ (or vice versa).

Consider the PPA-composition $C_{D,F}$ of D and F. We claim that for every vertex u, the degree of u in G_z is odd if and only if u is a satisfying assignment for $C_{D,F}$. This is equivalent to saying that the parity of the degree of u is the same as the parity of the satisfied components of $C_{D,F}$. The proof of this claim is straightforward, but somewhat tedious. We distinguish three cases in the proof, depending on the cardinality of $M(z, u) \setminus \{u\}$.

- Case 1: $M(z, u) \setminus \{u\} = \emptyset$. Then u is an isolated vertex, and all six components are satisfied.

- Case 2: $M(z,u) \setminus \{u\} = \{v\}$.
  a) If $u \in M(z,v)$ then the degree of u is one, and $I_5 \diamond F_3 \circ F_4, I_6 \diamond F_5$ and exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ are satisfied.
  b) If $u \notin M(z,v)$ then u is an isolated vertex, and $I_5 \diamond F_3 \circ F_4$ and $I_6 \diamond F_5$ are satisfied.
- Case 3: $M(z,u) \setminus \{u\} = \{v,w\}$.
  a) If $u \in M(z,v) \cap M(z,w)$ then the degree of u is two, and exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ and exactly one of the two components $I_1 \diamond D_1 \circ F_1, I_5 \diamond F_3 \circ F_4$ are satisfied.
  b) If $u \in M(z,v)$ but $u \notin M(z,w)$ and say $D(u) = v$, then exactly one of the two components $I_2 \diamond F_2 \circ D_2, I_3 \diamond D_3 \circ D_4$ is satisfied.
  c) Finally, if $u \notin M(z,v) \cup M(z,w)$ then u is an isolated vertex, and none of the components is satisfied.



(a) Case 1 **(b)** Case 2-a **(c)** Case 2-b

(d) Case 3-a **(e)** Case 3-b **(f)** Case 3-c

Figure 14 The six cases of Theorem 13.

This finishes the proof of the claim. It follows that the number of satisfying assignments for $C_{D,F}$ is equal to the number of leaves in G_z, which is even. The standard leaf ω is a satisfying assignment for $C_{D,F}$, and therefore the output of PPA-Circuit Chevalley is another satisfying assignment, which is another leaf in G_z. ◀

Acknowledgments. Part of this work was performed when A. B., M. S. and S. Y. attended the program "Semidefinite and Matrix Methods for Optimization and Communication" hosted at the Institute for Mathematical Sciences, Singapore. We thank the Institute for the hospitality. G. I. is grateful to the Centre for Quantum Technologies, NUS where part of his research was accomplished.

We are very grateful to several anonymous referees for numerous insightful comments on the paper. We would also like to thank Hervé Fournier, Guillaume Malod and Sylvain Perifel for several helpful conversations.

References

1 J. Aisenberg, M. Bonet and S. Buss. 2–D Tucker is PPA-complete. *ECCC Report* no. 163, 2015.
2 N. Alon. Combinatorial Nullstellensatz. *Combinatorics, Probability and Computing*, 8:1–2, pages 7–29, 1999.
3 N. Alon. Discrete Mathematics: Methods and Challenges. *In Proc. of* 2002 *International Congress of Mathematicians (ICM)*, vol. I, pages 119–135, 2002.
4 P. Beame, S. Cook, J. Edmonds, R. Impagliazzo and T. Pitassi. The relative complexity of NP search problems. *Journal of Computer and System Sciences*, 57(1), pages 3–19, 1998.
5 K. Berman. Parity results on connected f-factors. *Dicrete Math.*, 59, pages 1–8, 1986.
6 J. Bondy and F. Halberstam. Parity theorems for cycles and cycles in graphs. *J. of Graph Theory*, 10, pages 107–115, 1986.

7 K. Cameron and J. Edmonds. Existentially poly-time theorems. *DIMACS Series Discrete Mathematics and Theoretical Computer Science*, 1, pages 83–99, 1990.

8 K. Cameron and J. Edmonds. Some graphic uses of an even number of odd nodes. *Ann. Inst. Fourier* 49, pages 1–13, 1999.

9 X. Chen and X. Deng. On the complexity of 2D discrete fixed point problem. *In Proc. of* 33*rd ICALP*, pages 489–500, 2006.

10 X. Chen and X. Deng. Settling the complexity of two-player Nash equilibrium. *In Proc. of* 47*th FOCS*, pages 261–272, 2006.

11 X. Chen, X. Deng and S.-H. Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3), pages 1–57, 2009.

12 C. Chevalley. Démonstration d'une hypothèse de M. Artin. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 11, pages 73–75, 1936.

13 X. Deng, J. Edmonds, Z. Feng, Z. Liu, Q. Qi and Z. Xu. Understanding PPA-completeness. *in Proc. of* 31*st CCC*, pages 23:1–23:25, 2016.

14 M. Grigni. A Sperner lemma complete for PPA. *Inform. Process. Lett.*, 77(5-6), pages 255–259, 2001.

15 E. Jeřábek. Integer factoring and modular square roots. *Journal of Computer and System Sciences*, 82, no. 2, pages 380–394, 2016.

16 Mark Jerrum and Marc Snir. Some Exact Complexity Results for Straight-Line Computations over Semirings. *J. ACM*, 29, no. 3, pages 874–897, 1982.

17 K. Friedl, G. Ivanyos, M. Santha and Y. Verhoeven. Locally 2-dimensional Sperner problem complete for the Polynomial Parity Argument classes. *In Proc.* 6*th CIAC*, pages 380–391, 2006.

18 S. Kintali. A compendium of PPAD-complete problems.

19 G. Malod. Polynômes et coefficients. *Doctoral Thesis*, Université Claude Bernard, Lyon 1, 2003.

20 G. Malod and N. Portier. Characterizing Valiant's algebraic complexity classes. *Journal of Complexity*, 24, pages 16–38, 2008.

21 N. Megiddo and C. Papadimitriou. On total functions, existence theorems and computational complexity. *Theoret. Comput. Sci.*, 81, pages 317–324, 1991.

22 C. Papadimitriou. On graph-theoretic lemmata and complexity classes. *In Proc. of* 31*st FOCS*, pages 794–801, 1990.

23 C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. System Sci.*, 48(3), pages 498–532, 1994.

24 M. Sipser. Introduction to the Theory of Computation. PWS Publishing Company, 1997.

25 S. Toida. Properties of an Euler graph. *J. Franklin Institute*, 95, pages 343–345, 1973.

26 Leslie Valiant. Completeness for parity problems. In *International Computing and Combinatorics Conference*, pages 1–8. Springer, 2005.

27 L. Varga. Combinatorial Nullstellensatz modulo prime powers and the Parity Argument. *Electr. J. Comb.*, 21(4), P4.44, 2014.

28 E. Warning. Bemerkung zur vorstehenden Arbeit von Herrn Chevalley. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 11, pages 76–83, 1936.

29 D. West. Pairs of adjacent Hamiltonian circuits with small intersection. *Studies of Applied Math.*, 59, pages 245–248, 1978.