Relationless completeness and separations

Pavel Hrubeš*, Avi Wigderson† and Amir Yehudayoff†
* *Department of Computer Science, Princeton University, Princeton, NJ.*
Email: pahrubes@gmail.com.
† *School of Mathematics, Institute for Advanced Study, Princeton, NJ.*
Emails: avi@ias.edu and amir.yehudayoff@gmail.com.

***Abstract*—This paper extends Valiant's work on VP and VNP to the settings in which variables are not multiplicatively commutative and/or associative. Our main result is a theory of completeness for these algebraic worlds. We define analogs of Valiant's classes VP and VNP, as well as of the polynomials permanent and determinant, in these worlds. We then prove that even in a completely relationless world which assumes no commutativity nor associativity, permanent remains VNP-complete, and determinant can polynomially simulate any arithmetic formula, just as in the standard commutative, associative world of Valiant.**

In the absence of associativity, the completeness proof gives rise to the following combinatorial problem: what is the smallest binary tree which contains as minors *all* binary trees with n leaves. We give an explicit construction of such a universal tree of polynomial size, a result of possibly independent interest.

Given that such non-trivial reductions are possible even without commutativity and associativity, we turn to lower bounds. In the non-associative, commutative world we prove exponential circuit lower bounds on explicit polynomials, separating the non-associative commutative analogs of VP and VNP. Obtaining such lower bounds and a separation in the complementary associative, non-commutative world has been open for about 30 years.

***Keywords*-Algebraic complexity. Completeness. Separations.**

I. INTRODUCTION

In his seminar paper [19], Valiant extended complexity theory to the algebraic setting of computation of multivariate polynomials. As analogs of the fundamental classes of Boolean functions P and NP, he defined two classes of polynomials, now called VP and VNP: the first are polynomials (of small degree) computable by small arithmetic circuits, and the second are polynomials definable as a Boolean sum $\sum_{\bar{e}\in\{0,1\}^p} f(\bar{x},\bar{e})$, with $f(\bar{x},\bar{e})$ in VP. The exponential summation is an arithmetic analogy of disjunction in the non-deterministic Boolean case. The strength of VNP is witnessed by the fact VNP contains every *explicit* polynomial[1]. Most polynomials one meets in mathematics are explicit in this sense, and this is also the case of the important permanent polynomial. Valiant goes on to show that under the natural *projection* reductions the permanent polynomial is complete for VNP, and that the determinant polynomially simulates any arithmetic formula (and thus by Hyafil's result [12] the determinant quasi-polynomially simulates VP). He goes on to draw important conclusions from these results, most prominently exposing the universality of linear algebra for efficient arithmetic computation, and bringing the natural *mathematical* question of permanent vs. determinant as capturing (a version of) the fundamental *computational* P vs. NP problem.

In the subsequent paper [20], Valiant proceeds to make the case that, given the lack of progress on proving lower bounds for general *Boolean* circuits, one may naturally turn to the strictly simpler problem of proving them for general *arithmetic* circuits. Assuming $GF(2)$ as the underlying field, the difference between the two problems is that Boolean circuits (which compute functions) may use cancellations arising from the relation $x^2 = x$, while arithmetic circuits (computing polynomials) may not. In a sense, Valiant proposes removing relations from the mathematical structure underlying the computation, as an alternative to the more common approach of handicapping circuits' structure, for example, by limiting their depth.

In this paper, we take Valiant's proposal of removing relations to its logical conclusion. In the usual arithmetic setting one assumes that polynomials satisfy standard multiplicative relations: that multiplication is commutative and associative. Proving superpolynomial lower bounds for commutative associative circuits (implying VP $\neq$ VNP) seems far from reach despite decades of attempts, and so we might try to prove lower bounds for circuits which cannot use either (or both) of these multiplicative relations[2]; a task which is strictly easier.

This approach gives rise to four natural classes of polynomial algebras. For a field $\mathbb{F}$ and a set of variables X, let us consider free algebras $\mathbb{F}_{\mathsf{A},\mathsf{C}}[X]$, $\mathbb{F}_{\mathsf{A},\bar{\mathsf{C}}}[X]$, $\mathbb{F}_{\bar{\mathsf{A}},\mathsf{C}}[X]$, $\mathbb{F}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}[X]$, where the indices indicate whether we assume that the variables X are commutativite resp. associative. The first is the standard polynomial algebra. In the non-comutative cases the order of variables in a monomial matters, and in the non-associative cases each monomial comes with a binary

---

Research supported by NSF Grant CCF-0832797.

[1] A polynomial of low degree is explicit, if we can compute the coefficient of any monomial in it in polynomial time.

[2] However, we assume that addition, as well as multiplication with field elements, is associative and commutative, and that multiplication distributes over addition.

tree which describes its mltiplicative structure.

Like Valiant, we can now look at complexity classes, which may be similarly denoted $\mathsf{VP}_{\mathsf{A},\mathsf{C}}$, $\mathsf{VP}_{\mathsf{A},\bar{\mathsf{C}}}$, $\mathsf{VP}_{\bar{\mathsf{A}},\mathsf{C}}$, $\mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ and $\mathsf{VNP}_{\mathsf{A},\mathsf{C}}$, $\mathsf{VNP}_{\mathsf{A},\bar{\mathsf{C}}}$, $\mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$, $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$. The first in each set is Valiant's VP and VNP. The classes $\mathsf{VP}_{\bar{\mathsf{A}},\mathsf{C}}$, $\mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$, for example, are the non-associative, commutative versions of Valiant's classes. It now makes sense to study the power and weakness of these classes, wonder about completeness, and in particular ask the VP vs. VNP question in each of its four variants.

A. *Non-commutative, associative computation*

Non-commutative algebras abound in mathematics, with the most common examples being tha algebra of matrices, group algebras of non-commutative groups, and the quaternion algebra. Note that, unlike the algebra $\mathbb{F}_{\mathsf{A},\bar{\mathsf{C}}}[X]$, these examples are not "free": for example, $d \times d$ matrices $x_1, \ldots x_{2d}$ satisfy the identity $\sum_{\sigma \in S_{2d}} \operatorname{sgn}(\sigma) \prod_{i=1}^{2d} x_{\sigma(i)} = 0$.

Algorithms over (non-commuting) matrices can be carried out by (commutative) operations on their entries, so it may not be clear where non-commutative algorithms arise naturally. But considering matrices as "atomic", and thus their multiplication as non-commutative, can be extremely useful. One classical example is Strassen's matrix multiplication algorithm, and its improvements [8,7,...], where the recursive step treats submatrices as atomic. Another example is the sequence of works, culminating in [5], on approximating permanent via evaluating determinants on random non-commuting elements (from Clifford algebras).

The weakness of non-commutative computation as compared with commutative one is usually illustrated by the following simple example. The polynomial $x^2 - y^2$ can be commutatively computed as $(x+y)(x-y)$, using one multiplication. Without commutativity, the term $xy - yx$ does not cancel, and we need two multiplications.

This example is just the tip of an iceberg. Computation over the free non-commutative algebra $\mathbb{F}_{\mathsf{A},\bar{\mathsf{C}}}[X]$ was studied in several papers, with Nisan's seminal paper [14] containing the most important techniques and results. Defining non-commutative versions of determinant and permanent $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}}$ and $\mathrm{PERM}_{\mathsf{A},\bar{\mathsf{C}}}$, Nisan proves an exponential lower bound for the size of any non-commutative formula for any of these polynomials. Moreover, he proves an exponential separation between the formula and circuit size in this model (a result which stands in sharp contrast to the commutative case, where formulae can superpolynomially simulate circuits [12], [22]). Nisan's lower bound techniques were later strengthened by Chien and Sinclair [4] to prove the same exponential lower bounds for $\mathrm{PERM}_{\mathsf{A},\bar{\mathsf{C}}}$ and $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}}$ in concrete non-commutative algebras, including the ones mentioned above.

The question of lower bounds for non-commutative circuits, however, remains wide open. We mention two very recent papers on the subject. One is by the present authors [11], where it is shown how exponential lower bounds for general non-commutative circuits may be obtained from certain *super-linear* commutative lower bounds. The other, by Arvind and Srinivasan [1], shows that $\mathrm{PERM}_{\mathsf{A},\bar{\mathsf{C}}}$ and $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}}$ have the same non-commutative circuit complexity, up to polynomial factors. This implies that small non-commutative circuits for the determinant gives small *commutative* circuits for the standard permanent. In other words, they prove that $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}} \in \mathsf{VP}_{\mathsf{A},\bar{\mathsf{C}}}$ implies $\mathsf{VP}_{\mathsf{A},\bar{\mathsf{C}}} = \mathsf{VNP}_{\mathsf{A},\bar{\mathsf{C}}}$ (and hence $\mathsf{VP} = \mathsf{VNP}$).

Our main contribution in this setting is simply establishing the completeness of $\mathrm{PERM}_{\mathsf{A},\bar{\mathsf{C}}}$ and $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}}$. Like in the commutative case, $\mathrm{PERM}_{\mathsf{A},\bar{\mathsf{C}}}$ is complete for $\mathsf{VNP}_{\mathsf{A},\bar{\mathsf{C}}}$ and $\mathrm{DET}_{\mathsf{A},\bar{\mathsf{C}}}$ can polynomially simulate non-commutative formulae. This was apparantly never observed before. We obtain this as a consequence of the more general statement in the non-commutative and non-associative case. A direct proof in non-commutative associative setting would, however, be easier and give better parameters.

B. *Non-associative computation*

We discuss here both the commutative and non-commutative versions of non-associative computation.

Non-associative algebras are quite common in mathematics as well. The most notable ones are Lie algebras, which may be viewed as matrix algebras with the "bracket" product[3]. Lie algebras are not commutative either, but do satisfy other nontrivial relations like $xx = 0$ and the Jacobi identity $(xy)z + (yz)x + (zx)y = 0$. There are some known non-associative algebras which are commutative, e.g. the Jordan algebras, but they too are not free.

It is evident that in the non-associative setting, when describing a polynomial, every monomial must come with a specification of the ordering of multiplications of the variables in it (this can be given by a parenthesis structure, or equivalently a binary tree whose leaves are the variables).

The composition of operations in computer programs is typically non-associative, which makes this issue present in wide areas of program schemas, verification and symbolic computation. One beautiful concrete example of how a non-associative algorithm is useful in solving a basic problem is Valiant's CFL (Context Free Language) recognizer, see [18]. For a fixed context free grammar[4] G, the recognizer on input x must determine if x can be generated by the grammar G. A classical cubic time algorithms existed for the problem, and Valiant gives a sub-cubic algorithm by reducing it to matrix multiplication. However, the matrix entries are elements of a non-associative monoid defined by Valiant from the given grammar G and input x. The algorithm solves a general problem: computing the transitive closure of an upper-triangular matrix with non-associative entries, giving a polynomial in which every monomial is computed in all possible multiplication orderings (corresponding to all possible derivations of the string x by the grammar G).

---

[3] For square matrices A, B, this product is defined by $[A, B] = AB - BA$.

[4] Without loss of generality, in Chomsky normal form.

The non-associative model, even when commutative, turns out to be simple enough to allow lower bound proofs. We use standard rank arguments to obtain exponential lower bounds for an explicit polynomial. We also prove, as in the non-commutative case, completeness theorems for the determinant $\mathrm{DET}_{\bar{\mathsf{A}},\mathsf{C}}$ and the permanent $\mathrm{PERM}_{\bar{\mathsf{A}},\mathsf{C}}$. Thus the exponential lower bound above holds also for the permanent, and we get a separation $\mathsf{VP}_{\bar{\mathsf{A}},\mathsf{C}} \neq \mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$. We note that this proof is certainly "natural" in the sense of Razborov and Rudich [16], which may be interpreted as meaning that efficient non-associative arithmetic circuits cannot compute "pseudo-random" polynomials.

C. *Completeness in the relationless model*

The completeness results in the previous sections are proved in a very weak computational model, namely, a model where no multiplicative relations hold. We consider such a weak model merely to view the completeness results in their most general form. Our proof extends Valiant's proof to the relationless setting. Indeed, we closely follow the recent version of this proof by Malod and Portier [13], which we find very amenable to the extensions we need. The loss of commutativity presents only a few subtleties which can be handled reasonably simply. The loss of associativity, however, presents a major hurdle, which is of a purely combinatorial nature.

In the absence of either commutativity or associativity, there are many ways to define permanent and determinant. In order for the completeness theorems to hold, we must define permanent and determinant in a careful, perhaps not the most natural, way. We find it noteworthy that such a definition is possible, and that the nontrivial completeness proofs can be carried out in such a weak computational model. When defining the relationless determinant $\mathrm{DET}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ and permanent $\mathrm{PERM}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$, we need to "commit" to an ordering of multiplication of monomials, which is capable of emulating arbitrary orderings. As orderings are specified by trees, this gives rise to a problem about finding a certain "universal" tree; we explain this problem and our results in the next subsection. We conclude this one with the consequences of this reduction: $\mathrm{PERM}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ is complete for the class $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$, and every polynomial computed by a relationless formula of size s is a projection of $\mathrm{DET}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ of size $s+1$. This consequently holds also in the free algebras $\mathbb{F}_{\mathsf{A},\mathsf{C}}[X], \mathbb{F}_{\mathsf{A},\bar{\mathsf{C}}}[X]$, and $\mathbb{F}_{\bar{\mathsf{A}},\mathsf{C}}[X]$.

D. *Universal trees*

In this paper, *binary trees* are rooted, uniform and ordered (see Section II-A).The *size* of a binary tree is the number of leaves in it. We now discuss two standard notions of universality of trees. We say that a binary tree $\mathcal{T}$ is

- n*-subgraph universal*, if $\mathcal{T}$ contains every binary tree of size at most n as a subtree,
- n*-minor universal*, if $\mathcal{T}$ contains every binary tree of size at most n as a minor.

In both cases, the tree $\mathcal{T}$ must be able to accommodate both very balanced and very unbalanced trees. As it happens, either type of universal tree would suffice for a completeness reduction, if it has polynomial size.

The question of subgraph universal trees was studied in [6]. They showed that the smallest n-subgraph universal tree has size $n^{\Theta(\log n)}$. This is superpolynomial in n and thus unsuitable for our purposes. Interesting, though not related to our goal, is the fact that there exist small subgraph universal *graphs*: in [9], it was shown that when a graph G is a good enough expander of linear size, it contains all (unordered) binary trees of size at most n as subgraphs. To the best of our knowledge, the question of minor universal trees was not studied before. In this paper we construct a n-minor universal tree of size $O(n^4)$, the construction being in polynomial time.

II. FORMAL DEFINITIONS AND RESULTS

A. *Definitions*

We give formal definitions of the notions described above.

Relationless polynomials: Let $\mathbb{F}$ be a field, and X a set of variables. Unless otherwise stated, the variables we consider have no multiplicative relations: they are non-commutative and non-associative. However, addition remains commutative and associative, and multiplication is distributive over addition. We define *algebra of relationless polynomials*, $\mathbb{F}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}[X]$, as the free algebra generated by $\mathbb{F}, X$ and operations $+, \cdot$, where the operations satisfy, for every $a, b \in \mathbb{F}$ and $f, g, h \in \mathbb{F}\langle X\rangle$,

- $a + b = a +_{\mathbb{F}} b$, $a \cdot b = a \cdot_{\mathbb{F}} b$, where $+_{\mathbb{F}}, \cdot_{\mathbb{F}}$ are the field operations,
- $a \cdot f = f \cdot a$,
- $0 \cdot f = 0$, $1 \cdot f = f$, where $0, 1 \in \mathbb{F}$ is the zero, unit element of $\mathbb{F}$,
- $f + g = g + f$, $f + (g + h) = (f + g) + h$, and
- $f \cdot (g + h) = f \cdot g + f \cdot h$, $(g + h) \cdot f = g \cdot f + h \cdot f$.

A *relationless polynomial* is an element of the algebra $\mathbb{F}\langle X\rangle$. *A monomial* in $\mathbb{F}\langle X\rangle$ is a product of variables in X. Every polynomial g in $\mathbb{F}\langle X\rangle$ can be uniquely written as $\sum_i b_i\alpha_i$, where $b_i \in \mathbb{F}$ and α_i are distinct monomials. The field element b_i is called the *coefficient* of α_i in g. The *degree* of a monomial α is defined in the obvious way: $\deg(x_i) = 1$ for a variable x_i and $\deg(\alpha_1\alpha_2) = \deg(\alpha_1) + \deg(\alpha_2)$. The *degree* of a polynomial f is the maximal degree of a monomial in f with a non-zero coefficient.

We can similarly define three other classes of polynomials, $\mathbb{F}_{\mathsf{A},\mathsf{C}}[X]$, $\mathbb{F}_{\bar{\mathsf{A}},\mathsf{C}}[X]$, $\mathbb{F}_{\mathsf{A},\bar{\mathsf{C}}}[X]$, depending on whether we include the relations $f \cdot g = g \cdot f$ and $f \cdot (g \cdot h) = (f \cdot g) \cdot h$.

Relationless circuits: We consider computations in the algebra of relationless polynomials. A *relationless arithmetic circuit* Φ is a directed acyclic graph as follows. Nodes (or gates) of in-degree zero are labelled by either a variable in X or a field element in $\mathbb{F}$. All the other nodes have in-degree two and they are labelled by either $+$ or $\times$. The two edges going into a gate v labelled by $\times$ are labelled by *left* and *right*. This is important for functionality, as the variables do not commute.

The two standard complexity measures for circuits are size and depth: the *size* of a circuit, $|\Phi|$, is the number of edges in it, and the *depth* of a circuit is the length of the longest directed path in it. For a node v in a circuit Φ, denote by Φ_v the sub-circuit of Φ rooted at v. Denote by X_v the set of variables that occur in Φ_v. The gate v computes a polynomial $\widehat{\Phi}_v \in \mathbb{F}\langle X_v \rangle$ in the obvious way. The *degree* of v, $\deg(v)$, is the degree of $\widehat{\Phi}_v$. A *formula* is a circuit in which every gate has out-degree one (and so it is a directed binary tree).

In this paper, *polynomials* and *circuits* stand for relationless polynomials and circuits, unless stated otherwise.

Complexity classes.: As mentioned before, Valiant defined the algebraic analogs of P and NP, which are now called VP and VNP. In this paper, we denote these classes by $\mathsf{VP}_{\mathsf{A,C}}, \mathsf{VNP}_{\mathsf{A,C}}$, and we define three other classes of polynomials $\mathsf{VP}_{\mathsf{A},\bar{\mathsf{C}}}, \mathsf{VP}_{\bar{\mathsf{A}},\mathsf{C}}, \mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ and $\mathsf{VNP}_{\mathsf{A},\bar{\mathsf{C}}}, \mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}, \mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ as well. Technically, there is one such class for every field $\mathbb{F}$, but we omit this dependency for the sake of simplicity. We state the formal definitions of $\mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}, \mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$, the other three definitions are similar.

A family of relationless polynomials $\{f_n\}$ is called *p-bounded*, if there exists a polynomial $q(n)$ so that for every n, the polynomial f_n is in $q(n)$ variables, it has degree at most $q(n)$, and it can be computed by a circuit of size at most $q(n)$. The class of relationless p-bounded families of polynomials is denoted $\mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$.

A family of relationless polynomials $\{f_n\}$ is called *p-definable*, if there exist polynomials $p(n)$ and $q(n)$, and a p-bounded family $\{g_n\}$ so that each f_n is in $q(n)$ variables, each g_n is in $q(n) + p(n)$ variables, and

$$f_n(x_1, \ldots, x_{q(n)}) = \sum_{e_1,\ldots,e_{p(n)} \in \{0,1\}} g_n(e_1, \ldots, e_{p(n)}, x_1, \ldots, x_{q(n)}).$$

The class of relationless p-definable families of polynomials is denoted $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$.

A polynomial f is called a *projection* of a polynomial g, if $f(x_1, \ldots, x_n) = g(y_1, \ldots, y_m)$, where each y_i is either a variable x_j or a field element. A family of relationless polynomials $\{f_n\}$ is called $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$-complete, if it is in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ and for every family $\{g_n\}$ in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$, there exists a polynomial $p(n)$ so that for every n, g_n is a projection of some f_m with $m \le p(n)$.

Let us state an important property of the four VNP classes. Let $\{f_n\}$ be a family of polynomials in one of the algebras $\mathbb{F}_{p,q}[X], p \in \{A, \bar{A}\}, q \in \{C, \bar{C}\}$, and let $\mathsf{VNP}_{p,q}$ be the corresponding VNP class. We say that $\{f_n\}$ is *explicit*, if f_n has polynomial degree and there exists a polynomial-time algorithm[5] which, given n and a monomial α as inputs, computes the coefficient of α in f_n.

Fact 1. *If $\{f_n\}$ is explicit then $\{f_n\} \in \mathsf{VNP}_{p,q}$.*

In the case $\mathsf{VNP}_{\mathsf{A,C}} = \mathsf{VNP}$, this was shown in [19]. We leave the other cases as an exercise for the reader.

Binary trees and universal trees: In this paper, *binary trees* are rooted, uniform and ordered. More exactly, a binary tree T

- has one special node called the root of T,
- every node in T which is not a leaf has exactly two children, and
- if a node u in T has two children u_1, u_2, one of the edges $\langle u, u_1 \rangle, \langle u, u_2 \rangle$ is labelled with *left* and the other with *right*.

The size of a binary tree, $|T|$, is the number of leaves in it[6]. If T_1, T_2 are two binary trees with roots u_1, u_2, we define (T_1, T_2) as the tree whose root u is connected to the two roots u_1 and u_2, and the edge $\langle u, u_1 \rangle$ is labelled by *left* and $\langle u, u_2 \rangle$ by *right*. If we do not allow redundant parenthesis $((\ldots))$, every parenthesis structure can be associated with a binary tree. For example, $(()())()$ is associated with the tree $((v_1, v_2), v_3)$. Every monomial α of degree r can thus be associated with a binary tree with r leaves, representing the multiplicative structure of α. Given an ordered set $F = (F_1, \ldots, F_r)$ of relationless polynomials, and given a binary tree T with r leaves, denote by $\prod^T F$ the product of the polynomials in F according to the tree T. For example, when $T = ((v_1, v_2), v_3)$,

$$\prod^{T}(x_1 + 1, x_2, x_3) = ((x_1 + 1)x_2)x_3 = (x_1 x_2)x_3 + x_2 x_3.$$

We want to define relationless permanent so that it is $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$-complete. The number of binary trees of size n is exponential in n. Hence there exists an exponential number of monomials of degree n and n variables, differing only in their multiplicative structure (in the associative, commutative world there is only one such monomial). This poses a difficulty in the definition of permanent, for its multiplicative structure must somehow encompass the structure of all possible monomials. In order to overcome this obstacle, we introduce the notion of *universal tree*. Roughly, a universal tree contains every small tree as a minor.

Let T be a binary tree and let V be a non-empty subset of the leaves of T. Define $\kappa(T; V)$ as the minor of T induced by the set V, formally defined as follows. If $T = v \in V$ then $\kappa(T; \{v\}) = v$. When $T = (T_1, T_2)$, let V_1 be the set of elements in V which are leaves of T_1 and V_2 be the elements of V which are leaves of T_2. Define

---

[5] That is, a Turing machine. To avoid discussing how to handle, say, reals on a Turing machine, assume that f_n has $0, 1$-coefficients.

[6] Since we assume that trees are uniform, the total number of nodes in a tree is at most twice the number of its leaves.

$$\kappa(T; V) = \begin{cases} \kappa(T_2; V_2) & \text{if } V_1 = \emptyset, \\ \kappa(T_1; V_1) & \text{if } V_2 = \emptyset, \\ (\kappa(T_1; V_1), \kappa(T_2, V_2)) & \text{otherwise.} \end{cases}$$

For a node v in T, denote by T_v the sub-tree of T rooted at v. The definition can be restated as follows:

1) Remove from T all nodes v such that the subtree of T_v does not contain an element of V. This gives a tree T'.
2) T' is, in general, not a binary tree, as it may contain nodes v with only one child v'. Contract such vertices until a binary tree is obtained.

Let $t \geq 1$ be a real number. We say that a tree $\mathcal{T}$ is t-*universal*, if for every binary tree T of size at most t, there exists a subset V of leaves of $\mathcal{T}$ so that $T = \kappa(\mathcal{T}; V)$. Clearly, we can always assume that t is a natural number; we allow $t \in \mathbb{R}$ for our convenience. In Section III, we show how to efficiently construct a universal tree $\mathcal{T}$ of size at most t^4.

Permanent and determinant.: Valiant [19] showed that permanent is VNP-complete. In the relationless world, there are many options to define the permanent. In order to show that it is $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$-complete, we have to define it in a specific way, using universal trees. This enables us to simulate the various multiplicative structures of relationless polynomials.

We first define $\mathrm{PERM}^{(T)}$ relatively to a given binary tree T. Let T be a binary tree with t leaves and let M be a $t \times t$ matrix. Define

$$\mathrm{PERM}^{(T)}(M) = \sum_{\sigma} \prod^{T} (M_{1,\sigma(1)}, \ldots, M_{t,\sigma(t)}),$$

where σ is a permutation of $[t] = \{1, \ldots, t\}$. Fix an integer n, and let $\mathcal{T} = \mathcal{T}(n)$ be the n-universal tree defined in Section III. Let m be the number of leaves in $\mathcal{T}$ (thus m is polynomial in n), and let $X = (x_{i,j})$ be a $m \times m$ matrix of variables. Define

$$\mathrm{PERM}(X) = \mathrm{PERM}_n(X) = \mathrm{PERM}^{(\mathcal{T})}(X)\,.$$

We also show that determinant is universal; we define the determinant similarly:

$$\mathrm{DET}^{(T)}(M) = \sum_{\sigma} (-1)^{\mathrm{sgn}(\sigma)} \prod^{T} (M_{1,\sigma(1)}, \ldots, M_{t,\sigma(t)}),$$

and

$$\mathrm{DET}(X) = \mathrm{DET}_n(X) = \mathrm{DET}^{(\mathcal{T})}(X).$$

B. *Results*

Let us state the main results of this paper.

Theorem 2. *The permanent is* $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$*-complete, over any field of characteristic different than two.*

Theorem 2 implies the following corollary.

Corollary 3. *The permanent is* $\mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$*-complete and* $\mathsf{VNP}_{\mathsf{A},\bar{\mathsf{C}}}$*-complete, over any field of characteristic different than two.*

An important step in the proof of the theorem is the following universality of the permanent and determinant, which is well-known in the associative, commutative world.

Theorem 4. *For every arithmetic formula* Φ*, there exists a matrix* M *of poly*$(|\Phi|)$*-dimension with entries that are either variables or field elements so that* $\widehat{\Phi} = \mathrm{PERM}(M)$*. A similar statement holds for the determinant, that is, there exists a matrix* M' *of poly*$(|\Phi|)$*-dimension with entries that are either variables or field elements so that* $\widehat{\Phi} = \mathrm{DET}(M')$*.*

A key ingredient in the two theorems above is the construction of a universal tree:

Theorem 5. *For every* $t \geq 1$*, there exists a* t*-universal tree* $\mathcal{T}$ *of size at most* t^4*. Moreover, we can construct* $\mathcal{T}$ *in polynomial time, and, given a binary tree* T *of size at most* t*, we can find* V *so that* $T = \kappa(\mathcal{T}; V)$ *in time polynomial in* t*.*

On the other hand, we can show that in the non-associative world, VP $\neq$ VNP. In Section V we prove an exponential lower bound on circuit size of an explicit non-associative commutative polynomial, which gives:

Theorem 6. *Over* any *field,* $\mathsf{VP}_{\bar{\mathsf{A}},\mathsf{C}} \neq \mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$.

This immediately implies:

Corollary 7. *Over* any *field,* $\mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}} \neq \mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$.

III. UNIVERSAL TREES

Universal sequences.: The first ingredient in the construction of the universal tree is a construction of a universal sequence. We say that a sequence of positive real numbers $\bar{b} = \langle b_1, \ldots, b_n \rangle \in \mathbb{R}^n$ *covers* a sequence of positive natural numbers $\bar{a} = \langle a_1, \ldots, a_m \rangle$, if there exist $i_1 < i_2 < \cdots < i_m \in [n]$ so that $a_j \leq b_{i_j}$ for every $j \in [m]$. The indices $i_1, \ldots, i_m$ are called *covering* of $\bar{a}$ by $\bar{b}$. For a real parameter $t \geq 1$, we say that $\bar{b} \in \mathbb{R}^n$ is t-*universal*, if $\bar{b}$ covers every $\bar{a} = \langle a_1, \ldots, a_m \rangle$ such that $a_1 + \cdots + a_m \leq t$.

Lemma 8. *For every* $t \geq 1$*, there exists a* t*-universal sequence* $\bar{b} = \langle b_1, \ldots, b_n \rangle$ *such that*

1) $\bar{b}$ *consists of real numbers of the form* $t/2^j$*,* $j \in \{0, \ldots, \lfloor \log t \rfloor\}$*, and*
2) *for every* $j \in \{0, \ldots, \lfloor \log t \rfloor\}$*, the number* $t/2^j$ *occurs exactly* 2^j *times in* $\bar{b}$*.*

The sequence can be constructed in time polynomial in t *and for a given* $\bar{a}$*, we can find a covering of* $\bar{a}$ *by* $\bar{b}$ *in polynomial time.*

Proof: For $t \geq 1$, let us construct a t-universal sequence $\bar{b}(t)$ by induction on $\lfloor t \rfloor$. If $\lfloor t \rfloor = 1$, set $\bar{b}(t) = \langle t \rangle$. If $\lfloor t \rfloor > 1$, let

$$\bar{b}(t) = \langle \bar{b}(t/2), t, \bar{b}(t/2) \rangle,$$

the concatenation of $\bar{b}(t/2), t, \bar{b}(t/2)$. It is easy to see that 1 and 2 are satisfied, and it thus remains to show that $\bar{b}(t)$ is indeed universal.

Assume $\lfloor t \rfloor > 1$, otherwise the statement is immediate. Let $\bar{a} = \langle a_1, \dots, a_m \rangle$ be a sequence of positive integers such that $s = \sum_{i \in [m]} a_i \leq t$. If $s \leq t/2$, then $\bar{b}(t)$ covers $\bar{a}$, since $\bar{b}(t/2)$ already covers a. Otherwise, let $j \in [m]$ be the smallest natural number so that $\sum_{i \leq j} a_i > t/2$. Hence $\sum_{i<j} a_i \leq t/2$ and $\sum_{i>j} a_i \leq t/2$. Therefore $\bar{b}(t/2)$ covers $a_1, \dots, a_{j-1}$ and $a_{j+1}, \dots, a_m$. Since $a_j \leq t$, $\bar{b}(t)$ covers $\bar{a}$. ■

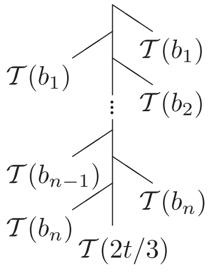
Lemma 8 implies the following corollary (it will not be used anywhere in this paper, but may be interesting in its own right).

Corollary 9. *For every* $t \geq 1$*, there exists a* t*-universal sequence* $\bar{b} = \langle b_1, \dots, b_n \rangle$ *such that* $n = O(t)$ *and* $b_1 + \cdots + b_n = t(\lfloor \log t \rfloor + 1)$.

The following fact is a commonly used property of trees, usually attributed to Spira [17].

Fact 10. *Let* T *be a binary tree with* $|T| \geq 2$*. Then there exists a node* v *in* T *such that* $|T|/3 < |T_v| \leq 2|T|/3$*. Such a node can be found in polynomial time.*

Proof of Theorem 5: We construct $\mathcal{T}(t)$ by induction on $\lfloor t \rfloor$. If $\lfloor t \rfloor = 1$, set $\mathcal{T}(t)$ to be a single node. Otherwise, let $\bar{b} = \langle b_1, \dots, b_n \rangle$ be the $(2t/3)$-universal sequence given by Lemma 8. We define $\mathcal{T}(t)$ by the following self-explanatory picture



The tree is an ordered one, edges going to the left of the central branch are labelled by *left* and the ones to the right by *right*. We denote the copies of $\mathcal{T}(b_i)$ to the left of the central branch by $\mathcal{T}^L(b_i)$ and the ones to the right by $\mathcal{T}^R(b_i)$. The bottom copy of $\mathcal{T}(2t/3)$ is denoted $\mathcal{T}_0$.

Let us first bound the size of $\mathcal{T}(t)$. We have $|\mathcal{T}(t)| = 1$ if $\lfloor t \rfloor = 1$, and by Lemma 8, if $\lfloor t \rfloor > 1$, the size of $\mathcal{T}$ is

$$|\mathcal{T}(t)| = |\mathcal{T}(s)| + 2\Big(\sum_{i=0}^{\lfloor \log s \rfloor} 2^i |\mathcal{T}(s/2^i)| \Big), \quad \text{where } s = 2t/3. \tag{1}$$

Looking for an upper bound $|\mathcal{T}(t)| \leq t^c, c > 1$, it is sufficient to satisfy

$$t^c \geq t^c (2/3)^c \Big(1 + 2\frac{2^{c-1}}{2^{c-1}-1} \Big).$$

Hence it is sufficient to have

$$(3/2)^c \geq \Big(1 + \frac{2^c}{2^{c-1}-1} \Big).$$

This is satisfied for $c = 4$. Obviously, the construction of the tree takes polynomial time.

Let us show that $\mathcal{T}(t)$ is t-universal. Fix a binary tree T of size $|T| \leq t$. We describe how to find a set V of the leaves of $\mathcal{T}(t)$ so that $T = \kappa(\mathcal{T}; V)$.

If $|T| = 1$, set V to be the leftmost leaf of $\mathcal{T}(t)$. Otherwise, let $|T| > 1$. Let v_0 be the node given by Fact 10 so that $|T|/3 < |T_{v_0}| \leq 2|T|/3$. Since $\mathcal{T}_0$ is $2t/3$ universal, we can find $V^{(0)}$ a subset of leaves of $\mathcal{T}_0$ such that

$$T_{v_0} = \kappa(\mathcal{T}_0; V^{(0)}) = \kappa(\mathcal{T}(t); V^{(0)}).$$

Let $\gamma = (v_m, \dots, v_1, v_0)$ be the directed path from the root of T to v_0. For $\ell \in [m]$, let u_ℓ be the child of v_ℓ in T that is not $v_{\ell-1}$, and let a_ℓ be the size of T_{u_ℓ}. Thus, $|T_{v_0}| + \sum_\ell a_\ell = |T| \leq t$ and so $\sum_\ell a_\ell \leq 2|T|/3 \leq 2t/3$. Since $\bar{b}$ is (2t/3)-universal, we can find $i_1 < i_2 < \cdots < i_m$ such that $a_1 \leq b_{i_1}, \dots, a_m \leq b_{i_m}$. If u_ℓ is the left child of v_ℓ, let $V^{(\ell)}$ be the subset of leaves of $\mathcal{T}^L(i_\ell)$ such that

$$T_{u_\ell} = \kappa(\mathcal{T}^L(i_\ell); V^{(\ell)}).$$

Such a set exists since $\mathcal{T}^L(i_\ell)$ is b_{i_ℓ}-universal and $a_\ell \leq b_{i_\ell}$. Otherwise, u_ℓ is the right child of v_ℓ; let $V^{(\ell)}$ be set of leaves of $\mathcal{T}^R(i_\ell)$ such that

$$T_{u_\ell} = \kappa(\mathcal{T}^R(i_\ell); V^{(\ell)}).$$

Define $V = V^{(0)} \cup \bigcup_{\ell \in [m]} V^{(\ell)}$. We claim that for every $\ell \in \{0, \dots, m\}$,

$$T_{v_\ell} = \kappa\Big(\mathcal{T}(t); V^{(0)} \cup \bigcup_{k \in \{0,\dots,\ell\}} V^{(k)} \Big).$$

This is straightforward to verify by induction on ℓ. The time it takes to find V is polynomial in t (similarly to (1)). ■

IV. COMPLETENESS

In this section we show that the permanent is complete even in the relationless world. The standard proof the permanent's completeness has three parts (see, for example, [3]). In the relationless world, the proof consists of three parts as well, but each one is slightly modified. We first describe the three steps.

Part I: simulating circuits by Boolean sums over formulas.: As in the standard proof, the first part is to show that circuits can be efficiently simulated by a Boolean sum over polynomial size formulas.

Denote by $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}{}^{e}$ the family of polynomials $\{f_n\}$ in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ so that the polynomials $\{g_n\}$ in the definition of $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ have polynomial size *formulas* (instead of polynomial size circuits). Formally, the first part of the proof is the following theorem, which is proved in Section IV-A

Theorem 11. $\mathsf{VP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}} \subseteq \mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}{}^{e}$*, over any field.*

Theorem 11 implies the following.

Corollary 12. $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}} = \mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}{}^{e}$*, over any field.*

Part II: universality.: The second part in our proof as well as in the standard proof is to show that the permanent is universal, that is, formulas can be efficiently simulated by permanents (a similar statement holds for determinant).

Lemma 13. *For every arithmetic formula Φ of size s, there exist a tree T with $t \leq s+1$ leaves and a $t \times t$ matrix M with entries that are either variables or field elements so that $\widehat{\Phi} = \mathrm{PERM}^{(T)}(M)$. A similar statement holds for the determinant, that is, there exists a $t \times t$ matrix M' with entries that are either variables or field elements so that $\widehat{\Phi} = \mathrm{DET}^{(T)}(M')$.*

Lemma 13 is proved in Section IV-B. It implies Theorem 4 by embedding M, M' in a matrix of $|\mathcal{T}(t)|$-dimension, as in the end of the proof of Theorem 2 below.

Part III: the permanent simulates Boolean sums.: In this part we show that we can write Boolean sum over one variable as permanent of a small matrix. This property differentiates the permanent from the determinant, and here we must assume that the underlying field has characteristic different from 2.

To state and prove this property, we need to add field elements to a given matrix. Let M be an $s \times s$ matrix. Let $M^{(1)}$ be a $s \times t$ matrix, $M^{(2)}$ be a $t \times s$ matrix and $M^{(3)}$ be a $t \times t$ matrix, so that all the entries in $M^{(1)}, M^{(2)}$ and $M^{(3)}$ are field elements. We call the matrix

$$M = \begin{bmatrix} M & M^{(1)} \\ M^{(2)} & M^{(3)} \end{bmatrix}$$

a *t-field-increment* of M. To define the permanent of M, we need to choose a canonical tree $P^{(t)}$ with t leaves (P stands for path): $P^{(1)}$ is the leaf v_1, and for $i > 1$, set $P^{(i)} = (v_i, P^{(i-1)})$ with a new leaf v_i.

The following theorem shows that the permanent simulates Boolean sums.

Proposition 14. *Assume that the underlying field is of characteristic different than two. Let M be an $s \times s$ matrix with entries that are variables, field elements, and a special variable e. Let T be a tree with s leaves. Let s_e be the number of entries in M that e occurs in.*

Then there exists a $(5s_e)$-field-increment M' of M so that

$$\begin{aligned} &\mathrm{PERM}^{(T')}(M'\big|_{e=0}) \\ &\qquad = \mathrm{PERM}^{(T)}(M\big|_{e=0}) + \mathrm{PERM}^{(T)}(M\big|_{e=1}), \end{aligned}$$

with $T' = (T, P^{(5s_e)})$. Every variable other than e occurs in H as many times as it occurs in M.

Proposition 14 is proved in Section IV-C.

The completeness proof.: Given these three parts, we can prove that the permanent is complete.

Proof of Theorem 2: The standard proof that the permanent is in VNP also implies that the permanent is in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ (alternatively, use Fact 1). It remains to show that every polynomial in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$ is reducible to permanent. Let $f = f_n$ be in $\mathsf{VNP}_{\bar{\mathsf{A}},\bar{\mathsf{C}}}$. Corollary 12 tells us that without loss of generality f is of the form

$$f(\bar{x}) = \sum_{\bar{e} \in \{0,1\}^p} g(\bar{e}, \bar{x}),$$

with p polynomial in n and g computable by a polynomial size formula. Lemma 13 tells us that there exists a polynomial size matrix M with entries that are either variables x_i, e_j and field elements so that

$$f(\bar{x}) = \sum_{\bar{b} \in \{0,1\}^p} \mathrm{PERM}(M\big|_{\bar{e}=\bar{b}}).$$

Proposition 14 tells us that there exist polynomial size matrices $H_1, \ldots, H_p$ and corresponding trees $T_1, \ldots, T_p$ so that

$$\begin{aligned} f &= \sum_{b_1,\ldots,b_{p-1}} \Big(\sum_{b_p} \mathrm{PERM}(M\big|_{e_p=b_p}) \Big) \\ &= \sum_{b_1,\ldots,b_{p-1}} \mathrm{PERM}^{(T_p)}(H_p) \\ &= \cdots = \sum_{b_1} \mathrm{PERM}^{(T_2)}(H_2\big|_{e_1=b_1}) = \mathrm{PERM}^{(T_1)}(H_1). \end{aligned}$$

Finally, let $\mathcal{T}$ be a $|T_1|$-universal tree of size polynomial in n, as given by Theorem 5. Let V be a subset of the leaves of $\mathcal{T}$ so that $\kappa(\mathcal{T}, V) = T_1$. Let M be a $|\mathcal{T}| \times |\mathcal{T}|$ matrix, whose rows and columns are labelled by leaves of $\mathcal{T}$, in the order defined by $\mathcal{T}$. Define the entries of M so that the restriction of M to the rows and columns in V are H_1, and in all the other rows and columns it is the identity matrix, namely, for every $\langle i, j \rangle$ not in $V \times V$, we have $M_{i,j} = 1$ if $i = j$ and $M_{i,j} = 0$ otherwise. Thus,

$$f = \mathrm{PERM}^{(T_1)}(H_1) = \mathrm{PERM}(M).$$

■

A. Simulating circuits by Boolean sums over formulas

In this section, we employ the method of Malod and Portier [13] to prove Theorem 11.

1) Parse trees: Given a circuit Ψ, we define a family of trees, which we call parse trees. They are intended to capture computation of monomials in Ψ. A *parse tree* T consists of nodes that are labelled by nodes of Ψ. The root of T is labelled by the output node of Ψ. If Ψ has only one node, T consists only of one node as well. If v is the output gate of Ψ, then

1) if $v = v_1 \times v_2$, then $T = (T_1, T_2)$, where T_1, T_2 are parse trees of Ψ_{v_1}, Ψ_{v_2}, and
2) if $v = v_1 + v_2$, then T is T_i with the edge $\langle v_i, v\rangle$ added to it, where T_i is a parse tree of Ψ_{v_i} and i is either 1 or 2.

A parse tree T computes a monomial $\widehat{T}$ in the obvious way: Let $\mu(T)$ be the minor of T that consists only of product gates and input gates, that is, after contracting all edges going into sum gates in T (μ stands for multiplicative part). The tree $\mu(T)$ is a binary tree. Every leaf v in $\mu(T)$ is an input gate in Ψ labelled by $\widehat{\Psi}_v$. The t leaves of $\mu(T)$ are ordered in a natural way: if $v = v_1 \times v_2$ in $\mu(T)$, then the leaves in $\mu(T)_{v_1}$ are smaller than the leaves in $\mu(T)_{v_2}$. Denote this order by $(v_1, v_2, \ldots, v_t)$. The monomial T computes is

$$\widehat{T} = \prod^{\mu(T)} (\widehat{\Psi}_{v_1}, \ldots, \widehat{\Psi}_{v_t}). \tag{2}$$

We say that a circuit Ψ is *multiplicatively disjoint* if for every $u = u_1 \times u_2$ in Ψ, the two circuits Ψ_{u_1} and Ψ_{u_2} are disjoint. When Ψ is multiplicatively disjoint, all the parse trees of Ψ are indeed trees. Multiplicatively disjoint circuits can be decomposed as follows.

Claim 15. *If Ψ is multiplicatively disjoint, then*

$$\widehat{\Psi} = \sum_T \widehat{T},$$

where T is a parse tree of Ψ.

Proof: The claim follows by induction on the size of Ψ. ■

The following proposition shows that circuits can be efficiently simulated by multiplicatively disjoint circuits. The proof proceeds in the same way as in [13], and will not be repeated here.

Proposition 16. *Let Φ be a relationless circuit of size s computing a polynomial f of degree r, then there is a multiplicatively disjoint relationless circuit Ψ of size $O(r^4 s)$ computing f as well.*

2) Parse trees are Boolean vectors: We use parse trees to simulate a circuit by a Boolean sum over a formula.

Proof of Theorem 11:

Let f be a polynomial in n variables of polynomial degree, and let Φ be a relationless circuit of polynomial size computing f. Let Ψ be the multiplicatively disjoint polynomial size circuit computing f given by Proposition 16, and let s be the size of Ψ, s being polynomial in n. Claim 15 implies that

$$f = \sum_T \widehat{T},$$

where T is a parse tree of Ψ. It thus remains to reduce the sum over parse trees to a Boolean sum. Intuitively, auxiliary Boolean variables will be used to identify parse trees of Ψ together with the monomials they compute.

Let $\mathcal{T} = \mathcal{T}(s)$ be the s-universal tree of polynomial size given by Theorem 5. Recall that we have a polynomial time algorithm that, given a tree T of size at most s, finds $V = V(T)$, a set of leaves of $\mathcal{T}$, so that $T = \kappa(\mathcal{T}; V)$. There may exist more than one such set V, we fix the set to be the one that the algorithm outputs. Let σ be the natural bijection between leaves of T and leaves in V, namely, if the leaves of T are $v_1, \ldots, v_m$ and the leaves in V are $u_1, \ldots, u_m$ (order the leaves according to the order defined by the trees), then $\sigma(v_i) = u_i$ for every $i \in [m]$.

We now define the auxiliary variables that we use. For every gate v in Ψ, let $a(v)$ be a variable, and let $\bar{a}$ be the vector of variables $a(v)$. For every leaf u of $\mathcal{T}$, let $b(u)$ be a variables, and let $\bar{b}$ be the vector of variables $b(u)$. For every leaf u of $\mathcal{T}$ and input gate w in Ψ, let $c(u, w)$ be a variable, and let $\bar{c}$ be the vector of variables $c(u, w)$. We are interested in zero-one assignments of $\bar{a}, \bar{b}, \bar{c}$.

Let ζ be a $0, 1$-assignment to the variables $\bar{a}, \bar{b}, \bar{c}$. We say that ζ is *good*, if

1) $T_\zeta = \{v : \zeta(a(v)) = 1\}$ is a parse tree of Ψ,
2) $V_\zeta = \{u : \zeta(b(u)) = 1\}$ is the set $V(\mu(T_\zeta))$ of leaves of $\mathcal{T}$, and
3) $\zeta(c(u, w)) = 1$ if and only if $u \in V(\mu(T_\zeta))$ and the leaf $\sigma^{-1}(u)$ of T_ζ is w.

For a leaf u of $\mathcal{T}$, let

$$L_u = (1 - b(u)) + \sum_w c(u, w) \widehat{\Psi}_w,$$

where w is an input gate of Ψ. If ζ is good, we thus have

$$\widehat{T}_\zeta = \prod^{\mathcal{T}} (L_{u_1}, \ldots, L_{u_m}) \Big|_\zeta \tag{3}$$

where $u_1, \ldots, u_m$ are the leaves of $\mathcal{T}$ and $\Big|_\zeta$ denotes substituting every variable e in $\bar{a}, \bar{b}, \bar{c}$ by $\zeta(e)$.

Given a Boolean assignment ζ, we can determine in polynomial time whether it is good. Cook's theorem tells us that there exists a polynomial size Boolean formula B in variables $\bar{a}, \bar{b}, \bar{c}$ and some additional variables $\bar{d}$ so that ζ is good if and only if there exists a zero-one assignment ζ_d to the variables $\bar{d}$ so that $B(\zeta, \zeta_d) = 1$. As we are interested in the value of B only for Boolean inputs, we can assume that B is an arithmetic formula. This also tells us that commutativity and associativity is not an issue. We can therefore write

$$f = \sum_T \widehat{T} = \sum_{\zeta,\zeta_d} B(\zeta,\zeta_d) \prod^{\mathcal{T}} (L_{u_1},\ldots,L_{u_m}) \Big|_{\zeta}$$

■

B. Universality

We now show that the permanent in universal in the relationless world; a similar claim holds for the determinant.

Claim 17. *For every arithmetic formula Φ, there exist $s \leq |\Phi|+1$, and an $s \times s$ matrix M with entries that are either variables or field elements and a tree T with s leaves so that $\widehat{\Phi} = \mathrm{PERM}^{(T)}(M)$. Moreover, M has the following property: for every $i \in [s-1]$, $M_{i,i+1} = 1$ and for every $j > i+1$, $M_{i,j} = 0$.*

Claim 17 immediately implies Lemma 13.

Proof: The lemma follows by induction on the size of Φ. If the size of Φ is one, set

$$M = \begin{bmatrix} 1 & 1 \\ 0 & \widehat{\Phi} \end{bmatrix}$$

and T a binary tree with two leaves. Thus, $\mathrm{PERM}^{(T)}(M) = \widehat{\Phi}$ and M has the claimed structure.

If $\Phi = \Phi_1 \times \Phi_2$, let $s_1, s_2, M_1, M_2, T_1, T_2$ be given by induction. Set $s = s_1 + s_2 \leq |\Phi| + 1$, set

$$M = \begin{bmatrix} M_1 & E \\ 0 & M_2 \end{bmatrix},$$

where E is a matrix that has a one in the lower left corner and zero elsewhere, and set $T = (T_1, T_2)$. Thus,

$$\begin{aligned} \mathrm{PERM}^{(T)}(M) &= \sum_{\pi'} \prod^{T} (M_{1,\pi(1)},\ldots,M_{s,\pi(s)}) \\ &\quad + \sum_{\pi_1,\pi_2} \prod^{T_1} (M_{1,\sigma(1)},\ldots,M_{s_1,\sigma(s_1)}) \\ &\qquad \cdot \prod^{T_2} (M_{1,\sigma(1)},\ldots,M_{s_2,\sigma(s_2)}) \\ &= 0 + \widehat{\Phi}_1 \cdot \widehat{\Phi}_2 = \widehat{\Phi}, \end{aligned}$$

where π' is a permutation so that $\pi'(s_1) = s_1 + 1$, π_1 is a permutation of $[s_1]$, and π_2 is a permutation of $[s_2]$. The reason why the sum over π' is zero is that by the pigeon hole principle, for every π', there exists $i > s_1$ so that $\pi'(i) \leq s_1$ and so $M_{i,\pi'(i)} = 0$.

If $\Phi = \Phi_1 + \Phi_2$, let $s_1, s_2, M_1, M_2, T_1, T_2$ be given by induction. Let $s = s_1 + s_2 + 1 \leq |\Phi| + 1$ and let M be the following $s \times s$ matrix

$$M = \begin{bmatrix} 1 & v & 0 & 0 \\ 0 & M_1 & v_1 & 0 \\ M_2[1] & 0 & v_2 & M_2[2^+], \end{bmatrix},$$

where v is a row vector with 1 in the leftmost entry and 0 elsewhere, v_1, v_2 are column vectors with 1 in the bottom entry and 0 elsewhere, $M_2[1]$ is the first column of M_2, and $M_2[2^+]$ is the matrix M_2 without the first column. Define a tree T with s leaves so that the following is satisfied for every polynomials,

$$\begin{aligned} &\prod^{T}(1, f_1, \ldots, f_{s_1}, 1, \ldots, 1) = \prod^{T_1}(f_1, \ldots, f_{s_1}) \quad \text{and} \\ &\prod^{T}(f_1, 1, \ldots, 1, f_2, \ldots, f_{s_2}) = \prod^{T_2}(f_1, \ldots, f_{s_2}) \,. \end{aligned} \tag{4}$$

To achieve this, let T' be the tree $((u_1, T_1), u_2)$, where u_1, u_2 are two new nodes. The tree T is obtained by attaching the tree T' to the leftmost leaf of the tree T_2.

The definition of M guarantees that the following properties of permutations π of $[s]$ are satisfied.

1) If $\pi(1) = 1$ and $\pi(s_1 + 1) = s_1 + 2$, then there exists $i \in [s]$ so that $M_{i,\pi(i)} = 0$.
2) If $\pi(1) = 1$, $\pi(s) = s_1 + 2$ and $M_{1,\pi(1)}, \ldots, M_{s,\pi(s)} \neq 0$, then $\pi(i) \in \{2, \ldots s_1 + 1\}$ for every $i \in \{2, \ldots s_1 + 1\}$ and $M_{i,\pi(i)} = 1$ for every $i \geq s_1 + 2$.
3) If $\pi(1) = 2$, $\pi(s_1 + 1) = s_1 + 2$ and $M_{1,\pi(1)}, \ldots, M_{s,\pi(s)} \neq 0$, then $\pi(i) \in \{1, s_1 + 3, \ldots s\}$ for every $i \in \{s_1 + 2, \ldots s\}$ and $M_{i,\pi(i)} = 1$, for every $1 \leq i \leq s_1 + 2$.
4) If $\pi(1) = 2$ and $\pi(s) = s_1 + 2$, then there exists $i \in [s]$ so that $M_{i,\pi(i)} = 0$.

Using (4), we can thus write

$$\begin{aligned} &\mathrm{PERM}^{T}(M) \\ &= \sum_{\pi \in (ii)} \prod^{T} (M_{1,\pi(1)}, \ldots, M_{s,\pi(s)}) \\ &\quad + \sum_{\pi \in (iii)} \prod^{T} (M_{1,\pi(1)}, \ldots, M_{s,\pi(s)}) \\ &= \sum_{\pi \in (ii)} \prod^{T} (1, M_{2,\pi(2)}, \ldots, M_{s_1+1,\pi(s_1+1)}, 1, \ldots, 1) \\ &\quad + \sum_{\pi \in (iii)} \prod^{T} (M_{1,\pi(1)}, 1, \ldots, 1, M_{s_1+2,\pi(s_1+2)} M_{s,\pi(s)}) \\ &= \mathrm{PERM}^{(T_1)}(M_1) + \mathrm{PERM}^{(T_2)}(M_2) = \widehat{\Phi}. \end{aligned}$$

To prove the statement for the determinant, we may need to change signs of v, v_1, v_2 in the case of a sum gate. ■

C. The permanent simulates Boolean sums

To prove Proposition 14 we use the fact that the proposition holds in the commutative, associative case [3].

Proof of Proposition 14: Assume that the underlying field is of characteristic different than two. Let M be an $s \times s$ matrix whose entries are distinct variables, except a special variable e that appear s_e times in M. That is, we assume that either $M_{i,j} = x_{ij}$ or $M_{i,j} = e$, where $e, x_{ij} : i, j \in [s]$ are distinct variables, and e appears at s_e positions in M.

The standard proof in the commutative, associative world tells us that we can construct a $(5s_e)$-field-increment M' of M so that

$$\mathrm{PERM}(M'\big|_{e=0}) = \mathrm{PERM}(M\big|_{e=0}) + \mathrm{PERM}(M\big|_{e=1})\,. \tag{5}$$

This implies that every variable other than e occurs in $M'\big|_{e=0}$ as many times as it occurs in M. Equality (5) is an equality of commutative, associative polynomials. Nevertheless, we claim that due to the structure of the polynomials in (5), the equality holds also in the relationless setting. Let T be a tree with s leaves, and let $T' = (T, P^{(5s_e)})$. We claim that

$$\begin{aligned} &\mathrm{PERM}^{(T')}(M'\big|_{e=0}) \\ &\quad = \mathrm{PERM}^{(T)}(M\big|_{e=0}) + \mathrm{PERM}^{(T)}(M\big|_{e=1}). \end{aligned} \tag{6}$$

This holds since every relationless monomial that appears in (6) corresponds to a unique associative, commutative monomial, as we now explain. Let α be a commutative, associative monomial of the form

$$\alpha = x_{i_1 j_1} x_{i_2 j_2} \dots x_{i_k j_k}\,,$$

where $i_1 < \cdots < i_k \in [s]$ and $j_1, \dots j_k \in [s]$ are distinct. Let $u_1, \dots u_s$ be the leaves of T, ordered according to the ordering of T. Let T_α be the tree $\kappa(T; V)$, where $V = \{u_{i_1}, \dots u_{i_k}\}$. Let $\alpha^\star$ be the non-commutative non-associative monomial defined as

$$\alpha^\star = \prod^{T_\alpha}(x_{i_1 j_1}, x_{i_2 j_2}, \dots, x_{i_k j_k})\,.$$

First, we can see that every monomial that has a non-zero coefficient in (6) is of the form $\alpha^\star$, where α is as above. Second, the coefficient of $\alpha^\star$ on the left resp. right hand side in (6) is equal to the coefficient of α on the left resp. right hand side of (5). (For the left hand side, inspect the definition of field-increment.) Hence, (5) implies (6). ■

V. THE NON-ASSOCIATIVE COMMUTATIVE WORLD

We now show that in the non-associative, commutative world, VNP is strictly stronger than VP. We do so by constructing an explicit non-associative polynomial that requires exponential size circuits. In this section, all polynomials are non-associative commutative.

Let $\{S_n\}$ be a fixed family of binary trees such that S_n has n leaves and the family is constructible in polynomial time. Otherwise the particular structure of S_n is not important. Let V_n be the polynomial of degree $2n$ and variables z_0, z_1

$$\mathsf{V}_n = \sum_{\zeta < \zeta'} \prod^{S_n}(z_{\zeta(1)}, \dots, z_{\zeta(n)}) \prod^{S_n}(z_{\zeta'(1)}, \dots, z_{\zeta'(n)})\,,$$

where $\zeta, \zeta' \in \{0,1\}^n$ and "$<$" stands for lexicographic ordering.

We are going to prove an exponential lower bound on the circuit-size of V_n. It would perhaps be more natural to define V_n as $\sum_\zeta \prod^{S_n}(z_{\zeta(1)}, \dots, z_{\zeta(n)}) \prod^{S_n}(z_{\zeta(1)}, \dots, z_{\zeta(n)})$. This would however allow us to prove the lower bound only if char $\mathbb{F} \neq 2$.

The first step of the proof is given by the following linear lower bound.

Lemma 18. *Assume that*

$$\sum_{i,j \in [k] : i < j} x_i x_j = \sum_{i \in [m]} f_i g_i\,,$$

where f_i, g_i are homogeneous polynomials of degree 1. Then m is at least $(k-1)/2$. This holds over any field.

Proof: The lemma is true even in the associative, commutative setting. The integer m is at least the minimum rank of a $k \times k$ matrix A which satisfies

$$A_{ii} = 0 \ \text{ and } \ A_{ij} + A_{ji} = 1$$

for every $i, j \in [k]$. Setting $B = A + A^t$, we obtain a matrix B with 0 on the diagonal and 1 everywhere else. Hence B has rank at least $k-1$, and A has rank at least $(k-1)/2$. ■

For a monomial α, let us say that it *respects* the tree S_n, if $\alpha = \prod^{S_n}(z_{\zeta(1)}, \dots, z_{\zeta(n)})$, where $\zeta \in \{0,1\}^n$; we write $\alpha = z(\zeta)$. A polynomial g *respects* S_n if every monomial α which has a non-zero coefficient in g respects S_n. In particular, g is a homogeneous polynomial of degree n (or g is zero). We use the following simple claim. It is this point where we essentially use non-associativity.

Claim 19. *Assume that β is a monomial with a non-zero coefficient in V_n and $\beta = \alpha_1 \alpha_2$, where $0 < \deg \alpha_1, \deg \alpha_2 < 2n$. Then both α_1 and α_2 respect S_n.*

We need one other claim, whose proof is the same as in the standard associative commutative case. We say that a polynomial is homogeneous if all its monomials with non-zero coefficients have the same degree. We say that a circuit Φ is homogeneous, if every node v in Φ computes a homogeneous polynomial.

Claim 20. *Assume that f is a homogeneous polynomial of degree d, and that f is computed by a circuit of size s. Then f can be computed by a homogeneous circuit of size $O(d^2 s)$.*

We are ready to prove the lower bound.

Proposition 21. *Over any field, every non-associative circuit computing V_n has size $2^{\Omega(n)}$.*

Proof: Let Φ be a circuit of size s computing V_n. Losing a factor of at most $O(n^2)$, we can assume that Φ is homogeneous. Let $v_1, \dots, v_m$ be the set of product nodes in Φ such that $v_i = u_i \times w_i$, $\deg v_i = 2n$ and

$\deg u_i, \deg w_i < 2n$. Thus $m \leq s$ and it is easy to show that

$$\mathsf{V}_n = \sum_{i \in [m]} a_i \widehat{\Phi}_{u_i} \widehat{\Phi}_{w_i} .$$

where a_i is a field element.

Let us show that there exist polynomials f_i, g_i, for $i \in [m]$, which respect S_n and

$$\mathsf{V}_n = \sum_{i \in [m]} f_i g_i . \quad (7)$$

For a polynomial $g = \sum_j b_j \alpha_j$ with every b_j a field element, define g' as the polynomial

$$g' = \sum_{j : \alpha_j \text{ respects } S_n} b_j \alpha_j .$$

The polynomial g' respects S_n. Claim 19 implies

$$\mathsf{V}_n = \sum_{i \in [m]} a_i \widehat{\Phi}'_{u_i} \widehat{\Phi}'_{w_i} ,$$

and (7) follows.

We use Lemma 18 to bound m in (7) from below. For every $\zeta \in \{0,1\}^n$, introduce a new variable x_ζ, and let $X = \{x_\zeta : \zeta \in \{0,1\}^n\}$. If $g = \sum_\zeta b_\zeta z(\zeta)$ is a polynomial in variables z_0, z_1 which respects S_n, let $g^\star$ be the polynomial $\sum_\zeta b_\zeta x_\zeta$ in variables X. The polynomial $g^\star$ is a homogeneous polynomial of degree 1 (or the zero polynomial). Equation (7) and the definition of V_n imply that

$$\sum_{\zeta, \zeta' \in \{0,1\}^n : \zeta < \zeta'} x_\zeta x_{\zeta'} = \sum_{i \in [m]} f_i^\star g_i^\star .$$

By Lemma 18, this implies that $m \geq (2^n - 1)/2$. ■

Since we assume that S_n can be constructed in time polynomial in n, V_n is in $\mathsf{VNP}_{\bar{\mathsf{A}},\mathsf{C}}$ by Fact 1. This and Proposition 21 imply Theorem 6.

ACKNOWLEDGMENT

We wish to thank Swastik Kopparty and Leslie Valiant for helpful discussions, and Valentine Kabanets for useful references.

REFERENCES

[1] V. Arvind and S. Srinivasan. On the hardness of the noncommutative determinant. ECCC TR09-103, 2009.

[2] M. Braverman, S. Cook, P. McKenzie, R. Santhanam and D. Wehr. Pebbles and branching programs for tree evaluation. MFCS, pages 175 – 186, 2009.

[3] P. Burgisser. Completeness and reduction in algebraic complexity theory. Springer-Verlag Berlin Heidelberg 2000.

[4] Algebras with polynomial identities and computing the determinant. S. Chien and A. Sinclair. SIAM Journal on Computing 37, pages 252 – 266, 2007.

[5] S. Chein, L. Rasmussen and A. Sinclair. Clifford algebras and approximating the permanent. Journal of Computer and Systems Sciences 67, pages 263-290, 2003.

[6] F. R. K. Chung, R. L. Graham and D. Coppersmith. On trees containing all small trees. The Theory of Applications of Graphs, John Wiley and Sons, pages 265–272, 1981.

[7] H. Cohn, R. Kleinberg, B. Szegedy and C. Umans. Group-theoretic algorithms for matrix multiplication. FOCS 05', pages 379–388, 2005.

[8] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation 9, pages 251–280, 1990.

[9] J. Friedman and N. Pippenger. Expanding graphs contain all small trees. Combinatorica 7(1), pages 71–76, 1987.

[10] M. Goldberg and E. Lifshitz. On minimal universal trees. Mat. Zametki 4, pages 371 – 378, 1968.

[11] P. Hrubes, A. Wigderson and A. Yehudayoff. Non-commutative circuits and the sum-of-squares problem. Manuscript, 2009.

[12] L. Hyafil. On the parallel evaluation of multivariate polynomials. SIAM J. Comput. 8(2), pages 120 – 123, 1979.

[13] G. Malod and N. Portier. Characterizing Valiant's algebraic complexity classes. J. Complexity 24(1), pages 16–38, 2008.

[14] N. Nisan. Lower bounds for non-commutative computation. STOC 91, pages 410–418, 1991.

[15] N. Nisan and A. Wigderson. Lower bounds on arithmetic circuits via partial derivatives. Computational Complexity, vol. 6, pages 217–234, 1996.

[16] A. A. Razborov and S. Rudich. Natural proofs. STOC 94', pages 204–213, 1994.

[17] P. M. Spira. On time-hardware complexity tradeoffs for Boolean functions. 4.Hawaii Symp.on Syst.Sc., 525-527.

[18] L. G. Valiant. General context-free recognition in less than cubic time. JCSS 102, pages 308–315, 1975.

[19] L. G. Valiant. Completeness classes in algebra. STOC '79, pages 249–261, 1979.

[20] L. G. Valiant. Why is Boolean complexity theory difficult? Poceedings of the London Mathematical Society symposium on Boolean function complexity, pages 84 – 94, 1992.

[21] L. G. Valiant and V. V. Vazirani NP is as easy as detecting unique solutions. STOC '85, pages 458463, 1985.

[22] L. G. Valiant, S. Skyum, S. Berkowitz and C. Rackoff. Fast parallel computation of polynomials using few processors. SIAM J. Comput. 12(4), pages 641–644, 1983.