Parameterized Valiant's Classes

Markus Bläser
Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
mblaeser@cs.uni-saarland.de

Christian Engels
IIT Bombay, Mumbai, India
christian@cse.ittb.ac.in

Abstract

We define a theory of parameterized algebraic complexity classes in analogy to parameterized Boolean counting classes. We define the classes VFPT and VW[t], which mirror the Boolean counting classes #FPT and #W[t], and define appropriate reductions and completeness notions. Our main contribution is the VW[1]-completeness proof of the parameterized clique family. This proof is far more complicated than in the Boolean world. It requires some new concepts like composition theorems for bounded exponential sums and Boolean-arithmetic formulas. In addition, we also look at two polynomials linked to the permanent with vastly different parameterized complexity.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases Algebraic complexity theory, parameterized complexity theory, Valiant's classes

Digital Object Identifier 10.4230/LIPIcs.IPEC.2019.3

Related Version `https://arxiv.org/abs/1907.12287`

Acknowledgements We thank Holger Dell for valuable comments on a first draft of this paper and Radu Curticapean and Marc Roth for helpful discussions.

1 Introduction

When Valiant invented the theory of computational counting and #P-completeness, he also defined an algebraic model for computing families of polynomials [24]. This was very natural, since many (Boolean) counting problems are evaluations of polynomials: Counting perfect matchings in bipartite graphs is the same as evaluating the permanent at the adjacency matrix, counting Hamiltonian tours in directed graphs is the same as evaluating the Hamiltonian cycle polynomial at the adjacency matrix, etc. There is a fruitful interplay between the Boolean and the algebraic world: algebraic methods like interpolation can be used to design counting algorithms as well as for proving hardness results. Proving lower bounds might be easier in the algebraic world and then we can use transfer theorems from the algebraic world to the Boolean world [3].

Parameterized counting complexity has been very successful in the recent years, see for instance [1, 8, 6, 7, 16, 23]. Parameterized complexity provides a more fine-grained view on #P-complete problems. There are problems like counting vertex covers of size k, which are fixed-parameter tractable, and others, which are presumably harder, like the problem of counting cliques of size k. Beside the classes VP and VNP (and subclasses of them), which correspond to time bounded computation in the Boolean world, there have also been definitions of algebraic classes that correspond to space bounded Boolean computation, see [18, 17, 20]. However, we are not aware of any parameterized classes in the algebraic world despite some algorithmic upper bounds being known, see for instance [4, 10].

1.1 Our Contribution

In this paper, we define a theory of parameterized algebraic complexity classes. While some of the definitions are rather obvious modifications of the Boolean ones and some of the basic theorems easily transfer from the Boolean world to the algebraic world, some concepts have to be modified. For instance, we cannot use projections to define hardness in general in the parameterized world, since they can only decrease the degree. On the one hand, one could choose the degree as a parameter, for instance, when computing the vertex cover polynomial. On the other hand, one could have parameterized families where the degree is always n, like the permanent on bounded (orientable[1]) genus graphs. One cannot compare these families with projections, although both of them turn out to be fixed parameter tractable. We could use c-reductions instead (which are the analogue of Turing reductions). However, these seem too powerful. We propose some intermediate concept, namely fpt-substitutions: We may replace the variables of a polynomial by other polynomials that are computed by small circuits (and not simply constants and variables like in the case of projections). This mirrors what is done in parsimonious reductions: the input is transformed by a polynomial time computable function but no post-processing is allowed.

Our main technical contribution is the VW[1]-completeness proof of the parameterized clique polynomial. This proof turns out to be far more complicated than in the Boolean world, since we are not counting satisfying assignments to Boolean circuits but we are computing sums over algebraic circuits. First we prove that one can combine two exponential sums into one sum. While this is very easy in the case of VNP, it turns out to be quite complicated in the case of VW[1]. Then we prove a normal form for so-called weft 1 circuits, the defining circuits for VW[1]. We go on with proving that the components consisting of all monomials that depend on a given number of variables of a polynomial computed by a weft 1 formula can be written as a bounded exponential sum over so-called Boolean-arithmetic expressions. We then show how to reduce such a sum to the clique problem.

In our final section, we study two polynomials based on cycle covers. In the first one, the covers consist of one cycle of length k and all other cycles being self loops. The second polynomial is similar, but allows all other cycles to be of constant length. We prove that the first problem is VW[1] complete while the second problem is hard for VW[t] for all t.

2 Valiant's Classes

We give a brief introduction to Valiant's classes, for further information we refer the reader to [2, 19]. Throughout the whole paper, K will denote the underlying ground field. An arithmetic circuit C is an acyclic directed graph such that every node has either indegree 0 or indegree 2. Nodes with indegree 0 are called input nodes and they are either labeled with a constant from K or with some variable. The other nodes are called computation gates, they are either labeled with $+$ (addition gate) or $*$ (multiplication gate). Every gate computes a polynomial in the obvious way. There is exactly one gate of outdegree 0, the polynomial computed there is the output of C. The size of a circuit is the number of edges in it. The depth of a circuit is length of a longest path from an input node to the output node. Later, we will also look at circuits in which the computation gates can have arbitrary fan-in.

The objects that we study will be polynomials over K in variables $X_1, X_2, \dots$ (Occasionally, we will rename these variables to make the presentation more readable.) We will denote

[1] We restrict ourselves to study orientable genus in this paper.

$\{X_1, X_2, \dots\}$ by X. The circuit complexity $C(f)$ of a polynomial f is the size of a smallest circuit computing f. We call a function $r\colon \mathbb{N} \to \mathbb{N}$ *p-bounded* if there is a polynomial p such that $r(n) \le p(n)$ for all n.

▶ **Definition 2.1.** *A sequence of polynomials $(f_n) \in K[X]$ is called a* p-family *if for all n,*
1. *$f_n \in K[X_1, \dots, X_{p(n)}]$ for some p-bounded function p and*
2. *$\deg f_n$ is p-bounded.*

▶ **Definition 2.2.** *The class* VP *consists of all p-families (f_n) such that $C(f_n)$ is p-bounded.*

Let $f \in K[X]$ be a polynomial and $s\colon X \to K[X]$ be a mapping that maps indeterminates to polynomials. Now, s can be extended in a unique way to an algebra endomorphism $K[X] \to K[X]$. We call s a *substitution*. (Think of the variables being replaced by polynomials.)

▶ **Definition 2.3.** 1. *Let $f, g \in K[X]$. f is called a* projection *of g if there is a substitution $r\colon X \to X \cup K$ such that $f = r(g)$. We write $f \le_p g$ in this case. (Since g is a polynomial, it only depends on a finite number of indeterminates. Therefore, we only need to specify a finite part of r.)*
2. *Let (f_n) and (g_n) be p-families. (f_n) is a* p-projection *of (g_n) if there is a p-bounded function $q\colon \mathbb{N} \to \mathbb{N}$ such that $f_n \le_p g_{q(n)}$. We write $(f_n) \le_p (g_n)$.*

Projections are very simple reductions. Therefore, we can also use them to define hardness for "small" complexity classes like VP. More powerful are so-called *c-reductions*, which are an analogue of Turing reductions. c-reductions are strictly more powerful than p-projections [15]. Let g be a polynomial in s variables. A g-oracle gate is a gate of arity s that on input $t_1, \dots, t_s$ outputs $g(t_1, \dots, t_s)$. The size of such a gate is s. $C^g(f)$ denotes the minimum size of a circuit with g-oracle gates that computes f. If G is a set of polynomials, then $C^G(f)$ is the minimum size of an arithmetic circuit that can use any g-oracle gates for any $g \in G$.

▶ **Definition 2.4.** *Let (f_n) and (g_n) be p-families. (f_n) is a* c-reduction *of (g_n) if there is a p-bounded function $q\colon \mathbb{N} \to \mathbb{N}$ such that $C^{G_{q(n)}}(f_n)$ is p-bounded, where $G_{q(n)} = \{g_i \mid i \le q(n)\}$. We write $(f_n) \le_c (g_n)$.*

▶ **Definition 2.5.** *A p-family (f_n) is in* VNP*, if there are p-bounded functions p and q and a sequence $(g_n) \in$* VP *of polynomials $g_n \in K[X_1, \dots, X_{p(n)}, Y_1, \dots, Y_{q(n)}]$ such that*

$$f_n = \sum_{e \in \{0,1\}^{q(n)}} g_n(X_1, \dots, X_{p(n)}, e_1, \dots, e_{q(n)}).$$

VP and VNP are algebraic analogues of the classes P and #P in the Boolean world. The permanent family (per_n) is complete for VNP under p-projections (over fields of characteristic distinct from two) and the problem of computing the permanent of a given $\{0,1\}$-matrix is complete for #P under parsimonious reductions.

3 Parameterized (Counting) Complexity

Parameterized counting complexity was introduced by Flum and Grohe [11]. We give a short introduction to fixed parameterized counting complexity. For more information on parameterized complexity, we refer the reader to [12, 9].

▶ **Definition 3.1.** *A* parameterized counting problem *is a function $F\colon \Sigma^* \times \mathbb{N} \to \mathbb{N}$.*

The idea is that an input has two components (x, k), $x \in \Sigma^*$ is the instance and the parameter k measures the "complexity" of the input.

▶ **Definition 3.2.** *A parameterized counting problem is* fixed parameter tractable *if there is an algorithm computing $F(x,k)$ in time $f(k)|x|^c$ for some computable function $f\colon \mathbb{N} \to \mathbb{N}$ and some constant c. The class of all fixed parameter tractable counting problems is denoted by* #FPT.

A parameterized counting problem is fixed-parameter tractable if the running time is polynomial in the instance size. The "combinatorial explosion" is only in the parameter k. In particular, the exponent of n does not depend on k. The classical example for a parameterized counting problem in #FPT is the vertex cover problem: Given a graph G and a natural number k, count all vertex covers of G of size k.

Fixed parameter tractable problems represent the "easy" problems in parameterized complexity. An indication that a problem is not fixed parameter tractable is that it is hard for the class #W[1]. Reductions that are used to define hardness are *parsimonious fpt-reductions*: Such a reduction maps an instance (x,k) to an instance (x',k') such that the value of the two instances is the same, the running time of the reduction is $f(k)|x|^c$ for some computable function f and a constant c, and there is a computable function g such that $k' \leq g(k)$. It is quite easy to see that the composition of two parsimonious fpt-reductions is again a parsimonious fpt-reduction and that #FPT is closed under parsimonious fpt-reductions.

We now define weft t formulas inductively.[2]

▶ **Definition 3.3.** *A* weft 0 *formula is a layered Boolean formula and the gates have fan-in two (over the basis $\wedge$, $\vee$, and $\neg$). A weft t formula is a layered Boolean formula where the gates have fan-in two, except one layer of gates that has unbounded fan-in. This formula has as inputs weft $t-1$ formulas.*

Weft t formulas have t layers of unbounded fan-in gates, and all other gate have bounded fan-in. Weft t formulas are the defining machine model of the #W[t] classes:

▶ **Definition 3.4.** *The class #W[t] are all parameterized counting problems that are reducible by parsimonious fpt-reductions to the following problem: Given a weft t formula C of constant depth and a parameter k, count all satisfying assignments of C that have exactly k 1s.*

A classical example of a counting problem, that is #W[1]-complete, is counting cliques of size k in a graph. Clique is used as a major complete problem for #W[1] by Flum and Grohe [11]. It is known that P = #P implies #FPT = #W[1]. Curticapean [5] proves that counting k-matchings, the parameterized analogue to the permanent, is #W[1]-hard (under Turing fptreductions).

4 Parameterized Valiant's Classes

We now define fixed-parameter variants of Valiant's classes. Our families of polynomials will now have two indices. They will be of the form $(p_{n,k})$. Here, n is the number of indeterminates and k is the parameter.

▶ **Definition 4.1.** *A parameterized p-family is a family $(p_{n,k})$ of polynomials such that*
1. *$p_{n,k} \in K[X_1,\ldots,X_{q(n)}]$ for some p-bounded function q, and*
2. *the degree of $p_{n,k}$ is p-bounded (as a function of $n+k$).*

[2] The term "weft" originates from textile fabrication and has been used in Boolean parameterized complexity from its very beginning.

The most natural parameterization is by the degree: Let (p_n) be any p-family then we get a parameterized family $(p_{n,k})$ by setting $p_{n,k} = H_k(p_n)$. Here $H_k(f)$ denotes the homogeneous part of degree k of some polynomial f.[3] Since $\deg(p_n)$ is polynomially bounded, $p_{n,k}$ is zero when k is large enough. (This will usually be the case for any parameterization.) More generally, we will also allow that $p_{n,k} = H_{t(k)}(p_n)$ for some function t that solely depends on k.

Recall that a vertex cover C of a graph $G = (V, E)$ is a subset of V such that for every edge $e \in E$ at least one endpoint is in C.

▶ **Example 4.2.** Let $\mathcal{G} = (G_n)$ be a family of graphs such that G_n has n nodes. We will assume that the nodes of G_n are $\{1, \dots, n\}$.

1. The vertex cover family $(\mathrm{VC}^{\mathcal{G}}_n)$ with respect to $\mathcal{G}$ is defined as

$$\mathrm{VC}^{\mathcal{G}}_n = \sum_{C \subseteq \{1,\dots,n\}} \prod_{i \in C} X_i$$

where the sum is taken over all vertex covers C of G_n.

2. The parameterized vertex cover family $(\mathrm{VC}^{\mathcal{G}}_{n,k})$, with respect to $\mathcal{G}$, is defined as

$$\mathrm{VC}^{\mathcal{G}}_{n,k} = \sum_{\substack{C \subseteq \{1,\dots,n\} \\ |C|=k}} \prod_{i \in C} X_i$$

where we now sum over all vertex covers of size k of G_n. This is a homogeneous polynomial of degree k. (We will call both families $\mathrm{VC}^{\mathcal{G}}$. There is no danger of confusion, since we mainly deal with the parameterized family.)

Every node has a label X_i and for every vertex cover we enumerate (or more precisely, sum up) its weight, which is the product of the labels of the nodes in it. Above, every graph family defines a particular vertex cover family. We can also define a unifying vertex cover family.

▶ **Example 4.3.** Let $E_{i,j}, X_i$, $1 \le i < j \le n$, be variables over some field K. The *parameterized vertex cover polynomial* of size n is defined by

$$\mathrm{VC}_{n,k} = \sum_{\substack{C \subseteq \{1,\dots,n\} \\ |C|=k}} \prod_{\substack{i,j \notin C \\ i<j}} (1 - E_{i,j}) \prod_{i \in C} X_i.$$

The parameterized vertex cover family is defined as $(\mathrm{VC}_{n,k})$.

If we set the variables $E_{i,j}$ to values $e_{i,j} \in \{0, 1\}$ we get the vertex cover polynomial of the graph given by the adjacency matrix $(e_{i,j})$. The first product is 0 if there is an uncovered edge. More generally, if we take a family of graphs $\mathcal{G} = (G_n)$ such that G_n has n nodes and if we plug in the adjacency matrix of G_n into in each $\mathrm{VC}_{n,k}$ then we get the family $(\mathrm{VC}^{\mathcal{G}}_{n,k})$. $(\mathrm{VC}^{\mathcal{G}}_{n,k})$ is parameterized by the degree since we have $\mathrm{VC}^{\mathcal{G}}_{n,k} = H_k(\mathrm{VC}^{\mathcal{G}}_n)$. $(\mathrm{VC}_{n,k})$, however, is not parameterized by the degree as $\mathrm{VC}_{n,k}$ contains monomials of degree polynomial in n (independent of k).

Recall that a clique C of a graph is a subset of the vertices such that for every pair of nodes in C there is an edge between them.

[3] I.e., the sum of all monomials of degree k with their coefficients.

▶ **Example 4.4.** 1. Let $E_{i,j}, X_i$, $1 \le i,j \le n$, $i < j$, be variables over some field K. The *clique polynomial* of size n is defined by

$$\mathrm{Clique}_n = \sum_{C \subseteq \{1,\dots,n\}} \prod_{\substack{i,j \in C \\ i<j}} E_{i,j} \prod_{i \in C} X_i.$$

The clique family is defined as (Clique_n).

2. The parameterized clique family $(\mathrm{Clique}_{n,k})$ is defined by

$$\mathrm{Clique}_{n,k} = \sum_{\substack{C \subseteq \{1,\dots,n\} \\ |C|=k}} \prod_{\substack{i,j \in C \\ i<j}} E_{i,j} \prod_{i \in C} X_i.$$

(Again, we will call both families Clique.)

If we set the variables $E_{i,j}$ to values $e_{i,j} \in \{0,1\}$, we get the clique polynomial of the graph given by the adjacency matrix $(e_{i,j})$, since the first product checks whether C is a clique. For each clique, we enumerate a monomial $\prod_{i \in C} X_i$. X_i is the label of the node i. Clique is a polynomial defined on edges and nodes. This seems to be necessary, since the polynomial $\sum_{C \subseteq \{1,\dots,n\}} \prod_{i \in C} X_i = (1 + X_1) \cdots (1 + X_n)$, which is the "node-only" version of clique polynomial of the complete graph, is easy to compute. Therefore, we cannot expect that the "node-only" version of the clique family is hard for some class.

Notice, that the parameterized clique family $(\mathrm{Clique}_{n,k})$ has variables standing in for vertices. These vertices seem to be necessary, as in the counting world, counting the number of k cliques and counting the number of k-independent sets are tightly related. Namely, the number of cliques is the number of independent sets on the complement graph. We want to keep this relationship as the problem is an important member of #W[1] and hence we incorporate the vertices.

$(\mathrm{Clique}_{n,k})$ is parameterized by the degree, since $\mathrm{Clique}_{n,k} = H_{\binom{k}{2}+k}(\mathrm{Clique}_n)$. Here is another example, beside the general vertex cover family, of a family that is parameterized by a different parameter:

▶ **Example 4.5.** Let $\mathcal{G} = (G_{n,k})$ be a family of bipartite graphs such that $G_{n,k}$ has n nodes on both sides and genus k, $k \le \lceil (n-2)^2/4 \rceil$.[4] Let $A_{n,k}$ be the $n \times n$-matrix that has a variable $X_{i,j}$ in position (i,j) if there is an edge between i and j in $G_{n,k}$ and a 0 otherwise. The *$\mathcal{G}$-parameterized permanent family* $\mathrm{per}^{\mathcal{G}} = (\mathrm{per}^{\mathcal{G}}_{n,k})$ is defined as $\mathrm{per}^{\mathcal{G}}_{n,k} = \mathrm{per}(A_{i,j})$.

There is another natural way to parameterize the permanent:

▶ **Example 4.6.** Given a $k \times n$-matrix $X = (X_{i,j})$ with variables as entries, the rectangular permanent is defined as

$$\mathrm{rper}_{n,k}(X) = \sum_{\substack{f\colon \{1,\dots,k\} \to \{1,\dots,n\} \\ f \text{ is injective}}} \prod_{i=1}^{k} X_{i,f(i)}.$$

When $k = n$ then this is the usual permanent. The *rectangular permanent family* is defined as $\mathrm{rper} = (\mathrm{rper}_{n,k})$.

[4] This is the genus of the $K_{n,n}$ [22].

We will give some more parameterizations of the permanent in Section 8 where we also prove some hardness results.

We now define fixed parameter variants of Valiant's classes.

▶ **Definition 4.7.** **1.** *A parameterized p-family $(p_{n,k})$ is in the class* VFPT *if $C(p_{n,k})$ is bounded by $f(k)p(n)$ for some p-bounded function p and some arbitrary function $f\colon \mathbb{N} \to \mathbb{N}$.*[5]

2. *The subclass of* VFPT *of all parameterized p-families that are parameterized by the degree is denoted by* VFPT_{deg}.

We will also say above that $C(p_{n,k})$ is *fpt-bounded.* We will see in one of the next sections that the vertex cover family and the bounded genus permanent are in VFPT. We will say that a family of circuits (C_n, k) has *fpt size* if the size is bounded by $f(k)p(n)$ for some function $f\colon \mathbb{N} \to \mathbb{N}$ and p-bounded function p.

▶ **Definition 4.8.** *A parameterized p-family $f = (f_{n,k})$ is an* fpt-projection *of another parameterized p-family $g = (g_{n,k})$ if there are functions $r, s, t\colon \mathbb{N} \to \mathbb{N}$ such that r is p-bounded and $f_{n,k}$ is a projection of $g_{r(n)s(k),k'}$ for some $k' \le t(k)$.*[6] *We write $f \le_p^{fpt} g$.*

▶ **Lemma 4.9.** *If $f \in$ VFPT (or VFPT_{deg}) and $g \le_p^{fpt} f$, then $g \in$ VFPT (or VFPT_{deg}, respectively).*

▶ **Lemma 4.10.** $\le_p^{fpt}$ *is transitive.*

One can define a notion of completeness. In the case of fpt-projections, the degree of the polynomial is the only meaningful parameter to consider: The permanent family on bounded genus graphs $\mathrm{per}^{\mathcal{G}}$ is in VFPT and so is (a variant of) the vertex cover family VC. However, every polynomial in the permanent family has degree equal to the number of nodes in the graph (independent of the genus) whereas the degree of the vertex cover polynomial depends on the degree. If a polynomial p is a projection of q, then $\deg p \le \deg q$. Therefore, $\mathrm{per}^{\mathcal{G}}$ cannot be an fpt-projection of VC. Now we can call a parameterized family f VFPT_{deg}*-complete* (under fpt-projections), if it is in VFPT_{deg} and for all $g \in \mathsf{VFPT}_{deg}$, $g \le_p^{fpt} f$.

For other parameters, we need a stronger notion of reduction. There are the so-called c-reductions, see [2], which are the analogue of Turing reductions in Valiant's world. This is the strongest kind of reduction one could use. However, the p-projections in Valiant's world seem to be weaker than parsimonious polynomial-time reductions in the Boolean world. Therefore, we propose an intermediate concept, which models parsimonious reductions in the algebraic world. In parsimonious reductions, the input instance is transformed by a polynomial time or fpt computable reduction, then the function we reduce to is evaluated, and the result that we get shall be the value of our given function evaluated at the original instance.

In the algebraic world, this is modeled as follows: We call a p-family $f = (f_n)$ with $f_n \in K[X_1, \dots, X_{p(n)}]$ a *p-substitution* of a p-family $g = (g_n)$ with $g_n \in K[X_1, \dots, X_{q(n)}]$ if there is a p-bounded function r, and for all n, there are $h_1, \dots, h_{q(r(n))}$ such that $f_n = g_{r(n)}(h_1, \dots, h_{q(r(n))})$ and $\deg(h_i)$[7] as well as $C(h_i)$ is p-bounded for all i. We write $f \le_s g$.

[5] f need not be computable, since Valiant's model is nonuniform.
[6] k' might depend on n, but its size is bounded by a function in k. There are examples in the Boolean world, where this dependence on n is used.
[7] Note that a polynomial size circuit can construct superpolynomial degree polynomials by repeated squaring

Compared to a projection, we are now allowed to substitute polynomials of p-bounded complexity. We have that $\leq_s$ is transitive and that $p \leq_s q$ and $q \in \mathsf{VP}$ implies $p \in \mathsf{VP}$.

The parameterized analogue is defined as follows.

▶ **Definition 4.11.** *A parameterized p-family $f = (f_{n,k})$ with $f_{n,k} \in K[X_1, \dots, X_{p(n)}]$ is an* fpt-substitution *of another parameterized p-family $g = (g_{n,k})$ with $g_{n,k} \in K[X_1, \dots, X_{q(n)}]$ if there are functions $r, s, t \colon \mathbb{N} \to \mathbb{N}$ such that for all n, k, r is p-bounded and there exist polynomials $h_1, \dots, h_{q(r(n)s(k))} \in K[X_1, \dots, X_{p(n)}]$ such that*

$$f_{n,k} = g_{r(n)s(k),k'}(h_1, \dots, h_{q(r(n)s(k))})$$

for some $k' \leq t(k)$ and $\deg(h_i)$ as well as $C(h_i)$ are fpt-bounded (with respect to n and k) for all i. We write $f \leq_s^{fpt} g$.

The proof of the following two lemmas is almost identical to the proofs of Lemmas 4.9 and 4.10. The only difference is that fpt-substitutions do not preserve the degree.

▶ **Lemma 4.12.** *If $f \in \mathsf{VFPT}$ and $g \leq_s^{fpt} f$, then $g \in \mathsf{VFPT}$.*

▶ **Lemma 4.13.** *$\leq_s^{fpt}$ is transitive.*

To define an algebraic analogue of $\#\mathsf{W}[t]$, we study unbounded fan-in arithmetic circuits. These circuits have multiplication and addition gates of arbitrary fan-in. A gate with fan-in 2 will be called a gate of bounded fan-in, any other gate is a gate of unbounded fan-in. (Instead of 2, we can fix any other bound b.)

▶ **Definition 4.14.** *Let C be an arithmetic circuit. The* weft *of C is the maximum number of unbounded fan-in gates on any path from a leaf to the root.*

For $s, k \in \mathbb{N}$, $\left\langle {s \atop k} \right\rangle$ denotes the set of all $\{0, 1\}$-vectors of length s having exactly k 1s.

▶ **Definition 4.15.** 1. *A parameterized p-family $(f_{n,k})$ is in $\mathsf{VW}[t]$, if there is a p-family (g_n) of polynomials $g_n \in K[X_1, \dots, X_{p(n)}, Y_1, \dots, Y_{q(n)}]$ with p-bounded p and q such that g_n is computed by a constant depth unbounded fan-in circuit of weft $\leq t$ and polynomial size*[8] *and*

$$(f_{n,k}) \leq_s^{fpt} \Big(\sum_{e \in \left\langle {q(n) \atop k} \right\rangle} g_n(X_1, \dots, X_{p(n)}, e_1, \dots, e_{q(n)}) \Big). \tag{1}$$

2. *$\mathsf{VW}_{deg}[t]$ is the subset of all families in $\mathsf{VW}[t]$, that have the degree as the parameter.*

In essence, we emulate the Boolean $\#\mathsf{W}[t]$ definition. Instead of Boolean circuits of weft t we take an arithmetic circuit and instead of counting the number of assignments, we sum over all assignments. In addition, we only count the assignments that have weight k by adjusting the vectors we sum over, namely to $\{0, 1\}$-vectors with exactly k ones. While in the Boolean setting the closure is taken with respect to parsimonious fpt-reductions, in the arithmetic setting, we take fpt-substitutions. Hence, our definition seems to be the most appropriate analogue.

The clique family is in $\mathsf{VW}[1]$, since we can write it as

$$\mathrm{Clique}_{n,k} = \sum_{v \in \left\langle {n \atop k} \right\rangle} \prod_{\substack{i,j=1 \\ i<j}}^{n} (E_{i,j} v_i v_j + 1 - v_i v_j) \prod_{i=1}^{n} (X_i v_i + 1 - v_i).$$

[8] Note, that we do not need to require fpt-size, as we use an fpt sized reduction.

This formula has weft 1, since there are two unbounded product gates and none is a predecessor of the other. We replace the product over all C by a product over all vertices and use the v-vectors to switch variables on and off.

Like in the Boolean case, we will show that the parameterized clique family is complete for the class VW[1] (albeit for a stronger notion of reductions, namely fpt-c-reductions). It turns out that this proof is far more complicated than in the Boolean setting, since our circuits can compute arbitrary polynomials and not only Boolean values. Furthermore, multiplication and addition cannot be reduced to each other since there is no analog of de Morgan's law.

▶ **Definition 4.16.** *Let $f = (f_{n,k})$ and $g = (g_{n,k})$ be parameterized p-families. f* fpt-c-reduces *to g if there is a p-bounded function $q\colon \mathbb{N} \to \mathbb{N}$ and functions $s, t\colon \mathbb{N} \to \mathbb{N}$ such that $C^{G_{q(n)s(k),t(k)}}(f_{n,k})$ is fpt-bounded, where $G_{q(n)s(k),t(k)} = \{g_{i,j} \mid i \le q(n)s(k),\ \ j \le t(k)\}$. We write $f \le_c^{fpt} g$.*

The following two lemmas are proved like for $\le_c$ and VP. We replace oracle gates by circuits and use the fact that fpt-bounded functions are closed under composition.

▶ **Lemma 4.17.** *If $f \in$ VFPT and $g \le_c^{fpt} f$, then $g \in$ VFPT.*

▶ **Lemma 4.18.** *$\le_c^{fpt}$ is transitive.*

So we have two different notions to define #W[t]-hardness. Presumably, they are different, see [15].

5 VFPT

▶ **Theorem 5.1.** *For every family of graphs $\mathcal{G} = (G_n)$, where G_n has n nodes, $\mathrm{VC}_{n,k}^{\mathcal{G}}$ is in VFPT$_{deg}$.*

▶ Remark 5.2. It is unlikely that the general family $\mathrm{VC}_{n,k}$ is in VFPT. Take any graph $G = (V, E)$ on n nodes and m edges and compute $\mathrm{VC}_{n,k}$ on this graph, i.e., set all edge variables to zero that do not occur in E.. Now, for $i < j$, we set

$$E_{i,j} = \begin{cases} 1 - S & \text{if } \{i, j\} \in E, \\ 0 & \text{otherwise,} \end{cases}$$

and $X_i = T$ for all i. Then we get a bivariate polynomial. This polynomial contains a monomial $S^i T^j$ iff there is a vertex cover of size j in G not covering i edges, or, equivalently, covering $m - i$ edges. Note that since the polynomial is now bivariate, we can easily compute its coefficients using interpolation. While the (Boolean decision version of) vertex cover is in FPT, it turns out [14] that the more general question whether there is a set of nodes of size k covering at least t edges is W[1]-hard (with parameter k). Therefore, it seems to be unlikely that $\mathrm{VC}_{n,k}$ has circuits of fpt size.

Mahajan and Saurabh [21] define another variant of the vertex cover polynomial. We multiply each cover by a product over the uncovered edges. They multiply by a product over the covered edges. Both polynomials are essentially equivalently, one can turn one into the other by dividing through the product over all edges, doing a variables transform, and removing divisions.

The *sun graph* $S_{n,k} = (V, E)$ on n nodes is defined as follows: The first k nodes form a clique. And every other node is connected to the nodes $1, \dots, k$, but to no other nodes, that is, the nodes $k + 1, \dots, n$ form an independent set. (Note that there are other definitions of sun graphs in the literature, but all of them look like a sun when drawn appropriately.) Every graph G with n nodes that contains a vertex cover of size k is a subgraph of $S_{n,k}$. To see this, we map the nodes of the vertex cover of G to the nodes of the clique and the remaining nodes of G to the other $n-k$ nodes. Note that there are cannot be any edges in G between the nodes outside of the vertex cover.

We define $\mathrm{VC}^s_{n,k}$ like $\mathrm{VC}_{n,k}$ but on the graph $S_{n,k}$ instead of K_n, i.e., all edge variables not in $S_{n,k}$ are set to zero. The difference to VC is, that we now have some idea where the vertex cover is located (like it is in the Boolean case where we can find a potential set for instance by computing a maximum matching). Therefore, we can obtain:

▶ **Theorem 5.3.** $\mathrm{VC}^s \in \mathsf{VFPT}$.

Both parameterized permanent families turn out to be fixed parameter tractable.

▶ **Theorem 5.4** ([13])**.** *For every family of bipartite graphs $\mathcal{G} = (G_{n,k})$ such that $G_{n,k}$ has n nodes and genus k, $\mathrm{per}^{\mathcal{G}}$ is in* VFPT.

▶ **Theorem 5.5** ([25])**.** $\mathrm{rper} \in \mathsf{VFPT}$.

Kernelization is an important concept in parameterized complexity. In the algebraic setting, VFPT can also be characterized by kernels with size only bounded by k.

We also develop a notion of kernelization. We refer the reader to the full version.

6 The VW-hierarchy

We start with proving some basic facts about the $\mathsf{VW}[t]$ classes, in analogy to the Boolean world.

▶ **Lemma 6.1.** $\mathsf{VFPT} = \mathsf{VW}[0]$ *and* $\mathsf{VFPT}_{deg} = \mathsf{VW}_{deg}[0]$.

Proof. The proof is obvious, since $\mathsf{VW}[0]$ and $\mathsf{VW}_{deg}[0]$ are defined as the closure under fpt-substitutions, so we can compute problems in VFPT simply by using the reduction. ◀

The following lemma is obvious.

▶ **Lemma 6.2.** *For every t,* $\mathsf{VW}[t] \subseteq \mathsf{VW}[t+1]$ *and* $\mathsf{VW}_{deg}[t] \subseteq \mathsf{VW}_{deg}[t+1]$.

We call a parameterized p-family f $\mathsf{VW}[t]$*-hard* (under fpt-substitutions), if for all $g \in \mathsf{VW}[t]$, $g \leq_s^{fpt} f$. f is called $\mathsf{VW}[t]$*-complete* (under fpt-substitutions) if in addition, $f \in \mathsf{VW}[t]$. If the same way, we can also define hardness and completeness under fpt-c-reductions.

For the classes $\mathsf{VW}_{deg}[t]$, it is reasonable to study hardness and completeness under fpt-projections. We call a parameterized p-family f $\mathsf{VW}_{deg}[t]$*-hard* (under fpt-projections), if for all $g \in \mathsf{VW}[t]$, $g \leq_p^{fpt} f$. f is called $\mathsf{VW}_{deg}[t]$*-complete* (under fpt-projections) if in addition, $f \in \mathsf{VW}[t]$.

▶ **Lemma 6.3.** *If f is* $\mathsf{VW}[t+1]$*-complete under fpt-substitutions and $f \in$* $\mathsf{VW}[t]$*, then* $\mathsf{VW}[t] = \mathsf{VW}[t+1]$*. In the same way, if f is* $\mathsf{VW}_{deg}[t+1]$*-complete under fpt-substitutions or fpt-projections and $f \in$* $\mathsf{VW}_{deg}[t]$*, then* $\mathsf{VW}_{deg}[t] = \mathsf{VW}_{deg}[t+1]$.

It is open in the Boolean case whether $\mathsf{W}[t] = \mathsf{W}[t+1]$ or $\#\mathsf{W}[t] = \#\mathsf{W}[t+1]$ implies a collapse of the corresponding hierarchy. Maybe the algebraic setting can provide more insights.

▶ **Theorem 6.4.** *If* $\mathsf{VFPT} \neq \mathsf{VW}[1]$ *then* $\mathsf{VP} \neq \mathsf{VNP}$.

If one takes the defining problems for $\mathsf{VW}[t]$ (sums over $\{0,1\}$ vectors with k 1s of weft t circuits) instead of clique, one can prove the same theorem for arbitrary classes $\mathsf{VW}[t]$ in place of $\mathsf{VW}[1]$. The proof only gets technically a little more complicated.

7 Hardness of Clique

Our main technical result is the $\mathsf{VW}[1]$-hardness of Clique. The proof is technically much more intricate than in the Boolean setting. We will first give a short outline.

- First, we prove as a technical tool that two bounded exponential sums over a weft t circuit can be expressed by one exponential sum over a (different) weft t circuit. In the case of VNP, a similar proof is easy: Instead of summing over bit vectors of length p and then of length q, we can sum over bit vectors of length $p+q$ instead. If the number of ones is however bounded, this does not work easily anymore. It turns out that for the most interesting class $\mathsf{VW}[1]$ of the VW-hierarchy, the construction is astonishingly complicated.
- Next, we prove a normal form for weft 1 circuits. Every weft 1 circuit can be replaced by an equivalent weft 1 circuit that has five layers: The first layer is a bounded summation gate, the second layer consist of bounded multiplication gates, the third layer is the only layer of unbounded gates, the fourth layer again consists of bounded addition gates and the fifth layer of bounded multiplication gates.
- Then we introduce *Boolean-arithmetic formulas*: A Boolean-arithmetic formula is a formula of the form

  $$B(X_1,\dots,X_n)\cdot\prod_{i=1}^{n}(R_iX_i+1-X_i)$$

  where B is an arithmetization of some Boolean formula and the R_i some polynomial or even rational function (over a different set of variables). For each satisfying $\{0,1\}$-assignment e to B, that is, $B(e)=1$, the right hand side produces one product and the e_i (assigned to the X_i variables) switch the factors R_i on or off. For a polynomial f, the monomials of support size k are all monomials that depend on exactly k variables. The sum of all these monomials is denoted by $S_k(f)$. A central result for the hardness proof is that when f is computed by a circuit of weft 1, then we can write $S_k(f)$ as a bounded sum over a weft 1 Boolean arithmetic expression, that is, $S_k(f)=\sum_{e\in\left\langle {p(n,k) \atop q(k)}\right\rangle}\mathcal{B}(e)\cdot\prod_{i=1}^{p(n,k)}(R_ie_i+1-e_i)$.
- Finally, we prove that Clique is $\mathsf{VW}[1]$-complete under fpt-c-reductions (or under fpt-substitutions that allow rational expressions). Given some bounded sum over a polynomial $g_n(X_1,\dots,X_p,Y_1\dots,Y_q)$ computed by a weft 1 circuit, we view g_n as a polynomial over the Y-variables, the coefficients of which are polynomials in the X-variables. Then $S_0(g),\dots,S_k(g)$ are the parts of g_n that contribute to the sum when summing over all bit vectors with k ones. We can write this as a bounded sum over a Boolean-arithmetic formula. The concept of Boolean-arithmetic formulas allows us to reuse parts of the Boolean hardness proof.

Note that once we have the $\mathsf{VW}[1]$-hardness of Clique, we get further hardness results for free.

8 Hardness of the Permanent and Cycle Covers

In this section we will highlight the vast difference between the provable complexity of the following two problems. Having cycle covers where one cycle is of length k and all other cycles are self loops and the complexity of all cycle covers where one cycle is length k and all other cycle covers are of length some fixed constant c. As always in this paper, we will look at corresponding polynomials to this problem.

8.1 Hardness of the k-permanent

We are adapting the proof from Curticapean and Marx [8] to show the hardness of the following parameterization of the permanent. Notice that the theorem from [8] is not enough for us. It is in general unclear how and in which way the cycles transfer in the theorem while we need an explicit fpt-projection.

We define the k-permanent polynomial as follows. Let S'_n be the set of all permutations on n elements that map $n-k$ elements to itself. Then

$$\mathrm{per}_k = \sum_{\sigma \in S'_n} \prod_{i \in [n]} x_{i,\sigma(i)}.$$

Notice, we do not include the selected vertices, as all vertices are in the cycle cover and hence the two problems are equivalent.

▶ **Corollary 8.1.** (per_k) *is* $\mathsf{VW}[1]$ *hard under fpt-c-reductions.*

8.2 Bounded length Cycle Covers

▶ **Definition 8.2.** *We define* $\mathrm{per}_{\leq c,k}$*, the bounded length k-permanent, to be the following polynomial over all cycle covers where one cycle has length k and all other cycles have length bounded by some constant $c \geq 3$.*

$$\mathrm{per}_{\leq c,k} = \sum_{\sigma \in S''_n} \prod_{i \in [n]} x_{i,\sigma(i)}$$

where S''_n is the set of permutations σ on n elements such that σ has one cycle of length k and all other cycles have length $\leq c$.

With this, we can prove the following theorem.

▶ **Theorem 8.3.** *For all t, there exists a constant c such that* $\mathrm{per} \leq c, k$ *is hard for* $\mathsf{VW}[t]$ *under fpt-c-reductions.*

References

1 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Narrow sieves for parameterized paths and packings. *J. Comput. Syst. Sci.*, 87:119–139, 2017. `doi:10.1016/j.jcss.2017.03.003`.

2 Peter Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*. Springer, 2000.

3 Peter Bürgisser. Cook's versus Valiant's hypothesis. *Theor. Comput. Sci.*, 235(1):71–88, 2000. `doi:10.1016/S0304-3975(99)00183-8`.

4 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic. *Discrete Applied Mathematics*, 108(1-2):23–52, 2001. `doi:10.1016/S0166-218X(00)00221-3`.

5 Radu Curticapean. Counting Matchings of Size k Is W[1]-Hard. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, volume 7965 of *Lecture Notes in Computer Science*, pages 352–363. Springer, 2013. doi:10.1007/978-3-642-39206-1_30.

6 Radu Curticapean, Holger Dell, Fedor V. Fomin, Leslie Ann Goldberg, and John Lapinskas. A Fixed-Parameter Perspective on #BIS. *CoRR*, abs/1702.05543, 2017. arXiv:1702.05543.

7 Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 210–223. ACM, 2017. doi:10.1145/3055399.3055502.

8 Radu Curticapean and Dániel Marx. Complexity of Counting Subgraphs: Only the Boundedness of the Vertex-Cover Number Counts. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 130–139. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.22.

9 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer, 2013.

10 Uffe Flarup, Pascal Koiran, and Laurent Lyaudet. On the Expressive Power of Planar Perfect Matching and Permanents of Bounded Treewidth Matrices. In Takeshi Tokuyama, editor, *Algorithms and Computation, 18th International Symposium, ISAAC 2007, Sendai, Japan, December 17-19, 2007, Proceedings*, volume 4835 of *Lecture Notes in Computer Science*, pages 124–136. Springer, 2007. doi:10.1007/978-3-540-77120-3_13.

11 Jörg Flum and Martin Grohe. The Parameterized Complexity of Counting Problems. *SIAM J. Comput.*, 33(4):892–922, 2004. doi:10.1137/S0097539703427203.

12 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

13 Anna Galluccio and Martin Loebl. On the Theory of Pfaffian Orientations. I. Perfect Matchings and Permanents. *Electr. J. Comb.*, 6, 1999. URL: http://www.combinatorics.org/Volume_6/Abstracts/v6i1r6.html.

14 Jiong Guo, Rolf Niedermeier, and Sebastian Wernicke. Parameterized Complexity of Vertex Cover Variants. *Theory Comput. Syst.*, 41(3):501–520, 2007. doi:10.1007/s00224-007-1309-3.

15 Christian Ikenmeyer and Stefan Mengel. On the relative power of reduction notions in arithmetic circuit complexity. *Inf. Process. Lett.*, 130:7–10, 2018. doi:10.1016/j.ipl.2017.09.009.

16 Mark Jerrum and Kitty Meeks. The parameterised complexity of counting even and odd induced subgraphs. *Combinatorica*, 37(5):965–990, 2017. doi:10.1007/s00493-016-3338-5.

17 Pascal Koiran and Sylvain Perifel. VPSPACE and a transfer theorem over the complex field. In Ludek Kucera and Antonín Kucera, editors, *Mathematical Foundations of Computer Science 2007, 32nd International Symposium, MFCS 2007, Ceský Krumlov, Czech Republic, August 26-31, 2007, Proceedings*, volume 4708 of *Lecture Notes in Computer Science*, pages 359–370. Springer, 2007. doi:10.1007/978-3-540-74456-6_33.

18 Pascal Koiran and Sylvain Perifel. VPSPACE and a transfer theorem over the reals. In Wolfgang Thomas and Pascal Weil, editors, *STACS 2007, 24th Annual Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, February 22-24, 2007, Proceedings*, volume 4393 of *Lecture Notes in Computer Science*, pages 417–428. Springer, 2007. doi:10.1007/978-3-540-70918-3_36.

19 Meena Mahajan. Algebraic Complexity Classes. *CoRR*, abs/1307.3863, 2013. arXiv:1307.3863.

20 Meena Mahajan and B. V. Raghavendra Rao. Small-Space Analogues of Valiant's Classes. In Miroslaw Kutylowski, Witold Charatonik, and Maciej Gebala, editors, *Fundamentals of Computation Theory, 17th International Symposium, FCT 2009, Wroclaw, Poland, September*

2-4, 2009. Proceedings, volume 5699 of *Lecture Notes in Computer Science*, pages 250–261. Springer, 2009. doi:10.1007/978-3-642-03409-1_23.

21 Meena Mahajan and Nitin Saurabh. Some Complete and Intermediate Polynomials in Algebraic Complexity Theory. In Alexander S. Kulikov and Gerhard J. Woeginger, editors, *Computer Science - Theory and Applications - 11th International Computer Science Symposium in Russia, CSR 2016, St. Petersburg, Russia, June 9-13, 2016, Proceedings*, volume 9691 of *Lecture Notes in Computer Science*, pages 251–265. Springer, 2016. doi:10.1007/978-3-319-34171-2_18.

22 Gerhard Ringel. Das Geschlecht des vollständigen paaren Graphen. *Abh. Math. Sem. Univ. Hamburg 28*, pages 139–150, 1965.

23 Marc Roth. Counting Restricted Homomorphisms via Möbius Inversion over Matroid Lattices. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 63:1–63:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ESA.2017.63.

24 Leslie G. Valiant. Completeness Classes in Algebra. In Michael J. Fischer, Richard A. DeMillo, Nancy A. Lynch, Walter A. Burkhard, and Alfred V. Aho, editors, *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 249–261. ACM, 1979. doi:10.1145/800135.804419.

25 Virginia Vassilevska and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 455–464. ACM, 2009. doi:10.1145/1536414.1536477.