Small-space analogues of Valiant's classes

Meena Mahajan and B.V. Raghavendra Rao

The Institute of Mathematical Sciences, Chennai 600 113, India.
{meena,bvrr}@imsc.res.in

Abstract. In the uniform circuit model of computation, the width of a boolean circuit exactly characterises the "space" complexity of the computed function. Looking for a similar relationship in Valiant's algebraic model of computation, we propose width of an arithmetic circuit as a possible measure of space. We introduce the class VL as an algebraic variant of deterministic log-space L. In the uniform setting, we show that our definition coincides with that of VPSPACE at polynomial width.
Further, to define algebraic variants of non-deterministic space-bounded classes, we introduce the notion of "read-once" certificates for arithmetic circuits. We show that polynomial-size algebraic branching programs can be expressed as a read-once exponential sum over polynomials in VL, *i.e.* $\mathsf{VBP} \in \Sigma^R \cdot \mathsf{VL}$. We also show that $\Sigma^R \cdot \mathsf{VBP} = \mathsf{VBP}$, *i.e.* VBPs are stable under read-once exponential sums. Further, we show that read-once exponential sums over a restricted class of constant-width arithmetic circuits are within VQP, and this is the largest known such subclass of poly-log-width circuits with this property.

1 Introduction

In the arithmetic circuit model of computation, Valiant introduced the classes VP and VNP to capture the complexity of polynomial families ([15], see also [4]). Over Boolean computation these classes correspond roughly to P and NP; over arithmetic computation with Boolean inputs they correspond roughly to #LogCFL and #P. Given the rich structure within P and LogCFL, it is natural to ask for a complexity theory that can describe arithmetic computation at this level. In particular, there are two well-known hierarchies within polynomial-size Boolean circuit families: the NC hierarchy based on depth, modelling parallel time on a parallel computer, and the SC hierarchy based on width, modelling simultaneous time-space complexity of P machines. It is straightforward to adapt Valiant's definition of VP to NC. But an adaptation capturing a space-bound is more tricky, especially when dealing with sub-linear space. The main question is: what would be a "right" measure for space? Two obvious choices are: 1) the number of arithmetic "cells" or registers used during the course of computation (i.e., the unit-space model), and 2) the size of a succinct description of the polynomials computed at each cell. A third choice is the complexity of computing the coefficient function for polynomials in the family. All three of these space measures have been studied in the literature, [14, 6, 9, 8], with varying degrees of

success. In particular, the models [14, 9, 8], when adapted to logarithmic space, are too powerful to give meaningful insights into small-space classes, whereas the model of [6] as defined for log-space is too weak.

In this paper, we propose yet another model for describing space-bounded computations of families of polynomials. Our model is based on width of arithmetic circuits, and captures both succinctness of coefficients and ease of evaluating the polynomials. Special cases of this model have been studied in the past: in [10], such circuits over $\mathbb{Z}$ with Boolean inputs and an additional syntactic degree restriction are studied, while in [11, 7] such circuits over arbitrary rings but restricted to be syntactic multilinear are studied. We show that our notion of space $\mathsf{VSPACE}(s)$ coincides with that of [9, 8] at polynomial space with uniformity (Theorem 1), and so far avoids the pitfalls of being too powerful or too weak at logarithmic space.

Continuing along this line, we propose a way of describing non-deterministic space-bounded computation in this context. The specific motivation for this is to obtain an analogue of the class non-deterministic log-space NL as well as an analogue of the result that $\mathsf{VNP} = \Sigma \cdot \mathsf{VP}$. Again, there is a well-known model for NL that easily carries over to the arithmetic setting, namely polynomial-size branching programs BP. But we are unable to compare VBP with our version of VL. Our model here for NL is based on read-once certificates, which also provide the correct description of NL in terms of L in the Boolean world. We show that the arithmetization of this model, $\Sigma^R \cdot \mathsf{VL}$ does contain arithmetic branching programs (Theorem 2).

Surprisingly, we are unable to show a good upper bound on the complexity of read-once certified log-space polynomial families. This raises the question: Is the read-once certification procedure inherently too powerful? We show that this is not always the case; for branching programs, read-once-certification adds no power at all (Theorem 3). Similarly, for polylog-width circuits where the syntactic degree is bounded by a polynomial, read-once certification does not take us beyond VQP (Theorem 4). Further, if the circuit is multiplicatively disjoint and of constant width, then read-once certification does not take us beyond VP.

2 Preliminaries

We use standard definitions for complexity classes such as polynomial space PSPACE, NC, L, NL and LogCFL (see *e.g.* [17],[1]).

An arithmetic circuit over a ring $\langle \mathbb{K}, +, \times, 0, 1 \rangle$ is a directed acyclic graph C, where vertices with non-zero in-degree are labelled from $\{+, \times\}$, and vertices of zero-in-degree (called *leaf* nodes) are labelled from $X \cup \mathbb{K}$, where $X = \{x_1 \ldots, x_n\}$ is the set of variable inputs to the circuit. An output node of C is a node of zero out-degree, and it computes a polynomial in $\mathbb{K}[X]$. (A circuit can have more than one output node, thus computing a set of polynomials.)

The following definitions apply to both arithmetic and boolean circuits, hence we simply use the term circuit. *Depth* of a circuit is the length of a longest path from a leaf node to an output node. *Size* of the circuit is the number of nodes

and edges in it. *Width* of a layered circuit is the maximum number of nodes at any particular layer. We assume that all output nodes appear at the last layer.

Polynomial size poly-log depth Boolean circuits form the class NC; NC^1 is the subclass of log-depth circuits and is known to be contained in L. Polynomial size poly-log width boolean circuits form the class SC; SC^0 is the subclass of constant-width circuits and SC^1 is the subclass of log-width circuits. It is known that SC^0 equals NC^1 ([2]) and uniform SC^1 equals L.

An arithmetic (resp. Boolean) circuit C is said to be *skew* if for every multiplication gate $f = g \times h$ (resp. $\wedge$ gate $f = g \wedge h$), either h or g is in $X \cup \mathbb{K}$. C is said to be *weakly skew* if for every $f = g \times h$, either the edge (g, f) or (h, f) is a bridge in the circuit, i.e removing the edge disconnects the circuit. Poly-size Boolean skew circuits are known to characterise NL([16]).

An *algebraic branching program* (BP for short) over a ring $\mathbb{K}$ is a layered directed acyclic graph, where edges are labelled from $\{x_1, \ldots, x_n\} \cup \mathbb{K}$. There are two designated nodes, s and t, where s has zero in-degree and t has zero out-degree. Size of a BP is the number of nodes and edges in it and width is the maximum number of nodes at any layer. Length of a BP is the number of layers in it. Depth of a BP B equals $1 + \mathsf{length}(B)$. The polynomial P computed by a BP is the sum of weights of all s-t paths in P, where weight of a path is the product of all edge labels in the path. We will also consider multi output BPs, where the above is generalised in the obvious way to several nodes $t_1, t_2, \ldots, t_m$ existing at the last level. Note that BPs can be simulated by skew circuits and vice versa with a constant blow up in the width.

VP denotes the class of families of polynomials $(f_n)_{n\geq 0}$ such that $\forall n \geq 0$

- $f_n \in \mathbb{K}[x_1, \ldots, x_{u(n)}]$, where $u \leq \mathsf{poly}(n)$
- $\mathsf{deg}(f_n) \leq \mathsf{poly}(n)$
- f_n can be computed by a polynomial size arithmetic circuit.

VP_e is the sub-class of VP corresponding to poly-size arithmetic formula (*i.e.* circuits with out-degree at most 1). If f_n can be computed by arithmetic circuits with resource bounds the same as NC^1 or SC^0 or SC^1, then we say the family is in VNC^1 or VSC^0 or VSC^1 respectively. It is known that VP_e is the same as VNC^1. If the circuits computing f_n have quasipolynomial size $2^{\log^c n}$, we say that $\{f_n\}$ is in the class VQP.

A polynomial family $(f_n)_{n\geq 0}$ is in VNP if there exists a family $(g_\ell)_{n\geq 0}$ in VP such that $f_n(X) = \sum_{e\in\{0,1\}^m} g_n(X, e)$, where m is bounded by $\mathsf{poly}(n)$.

We let VBP and VBWBP stand for classes corresponding to *poly*-size BPs of *poly* and *constant* width, respectively. Without loss of generality, we can treat these classes as skew circuits. ([13])

Let $\mathcal{C}$ be a complexity class defined in terms of Turing machines. A circuit family $(B_n)_{n\geq 0}$ is said to be $\mathcal{C}$-uniform, if the direct connection language for B_n can be decided in $\mathcal{C}$. (see [17])

3 Notion of space for arithmetic computations?

In the case of boolean computations, the notion of "width" of a circuit captures the notion of space in the Turing machine model (under certain uniformity assumptions). In the case of arithmetic computations, defining a notion of "space bounded computation" seems to be a hard task.

3.1 Previously studied notions

One possible measure for space is the number of arithmetic "cells" or registers used in the course of computation (i.e., the unit-space model). Michaux [14] showed that with this notion of space, any language that is decided by a machine in the Blum-Shub-Smale model of computation (a general model for algebraic computation capturing the idea of computation over reals, [3]; see also [4]) can also be computed using $O(1)$ registers. Hence there is no space-hierarchy theorem under this space measure.

Another possible measure is the size of a succinct description of the polynomials computed at each cell. In [6], Naurois introduced a notion of weak space in the Blum-Shub-Smale model, and introduced the corresponding log space classes $\mathsf{LOGSPACE}_W$ and PSPACE_W. This is in fact a way of measuring the complexity of succinctly describing the polynomials computed by or represented at each "real" cell. Though this is a very natural notion of "succinctness" of describing a polynomial, this definition has a few drawbacks:

1. $\mathsf{LOGSPACE}_W$ seems to be too weak to contain even NC^1 over $\mathbb{R}$, which is in contrast to the situation in the Boolean world.
2. The polynomials representable at every cell have to be "sparse", i.e., the number of monomials with non-zero coefficients should be bounded by some polynomial in the number of variables.

The second condition above makes the notion of weak space very restrictive if we adapt the definition to the Valiant's algebraic computation model. This is because the corresponding log-space class in this model will be computing only sparse polynomials, but in the non-uniform setting sparse polynomials are known to be contained in a highly restrictive class called skew formula ([11]), which is in fact a proper subclass of constant depth arithmetic circuits (i.e., VAC^0).

Koiran and Perifel ([9, 8]) suggested a notion of polynomial space for Valiant's ([15, 4]) classes. The main purpose of their definition was to prove a transfer theorem over $\mathbb{R}$ and $\mathbb{C}$. Under their definition Uniform-VPSPACE (the non-uniform counterpart can be defined similarly) is defined as the set of families (f_n) of multivariate polynomials $f_n \in F[x_1, \ldots, x_{u(n)}]$ with integer coefficients such that

- $u(n)$ is bounded by a polynomial in n.
- Size of coefficients of f_n is bounded by $2^{poly(n)}$.
- Degree of f_n is bounded by $2^{poly(n)}$.
- Every bit of the coefficient function of f_n is computable in PSPACE.

In [9], it was observed that the class VPSPACE is equivalent to the class of polynomials computed by arithmetic circuits of polynomial depth and exponential size. Such Boolean circuits compute exactly PSPACE, hence the name VPSPACE. Thus one approach to get reasonable smaller space complexity classes is to generalise this definition. We can consider $\mathsf{VSPACE}(s(n))$ to consist of families $(f_n)_{n\geq 1}$ of polynomials satisfying the following:

- $f_n \in \mathbb{Z}[x_1, \ldots, x_{u(n)}]$, where $u(n)$, the number of variables in f_n, is bounded by some polynomial in n.
- Degree of f_n is bounded by $2^{s(n)}$.
- The number of bits required to represent each of the coefficients of f_n is bounded by $2^{s(n)}$, *i.e.* the coefficients of f_n are in the range $[-2^{2^{s(n)}}, 2^{2^{s(n)}}]$
- Given n in unary, an index $i \in [1, 2^{s(n)}]$, and a monomial M, the ith bit of the coefficient of M in f_n is computable in $\mathsf{DSPACE}(s(n))$.

It is easy to see that with this definition, even the permanent function PERM_n is in log-space. Thus $\mathsf{VSPACE}(\log n)$ would be too big a class to be an arithmetic version of log-space. The reason here is that this definition, unlike that of [6], goes to the other extreme of considering only the complexity of coefficient functions and ignores the resource needed to add the monomials with non-zero coefficients. The relationship between the complexity of coefficient functions and the polynomials themselves is explored more thoroughly in [12].

3.2 Defining VPSPACE in terms of circuit width

In this section we propose width of a (layered) circuit as a possible measure of space for arithmetic computations.

Definition 1. *Let* $\mathsf{VWIDTH}(S)$ *(with* $S = S(n)$*) be the class of polynomial families* $(f_n)_{n\geq 0}$ *with the following properties,*

- *The number of variables* $u(n)$ *in* f_n *is bounded by* $\mathsf{poly}(n)$
- $f_n \in \mathbb{Z}[x_1, \ldots, x_{u(n)}]$*, i.e* f_n *has only integer coefficients*
- $\mathsf{deg}(f) \leq \max\{2^{S(n)}, \mathsf{poly}(n)\}$.
- *The coefficients of* f_n *are representable using* $\max\{2^{S(n)}, \mathsf{poly}(n)\}$ *many bits.*
- f_n *is computable by an arithmetic circuit of width* $S(n)$ *and size* $\leq \max\{2^{S(n)}, \mathsf{poly}(n)\}$.

Further, if the arithmetic circuits in the last condition are DSPACE(S)*-uniform, we call the family Uniform-*$\mathsf{VWIDTH}(S)$.

Remark 1. In [10], poly size circuits of log width and poly degree were introduced. The above definition generalises this definition to arbitrary width. A notable difference is that in [10] and [11], the degree bound was on the *syntactic degree* of the width-bounded circuits rather than on the degree of output polynomial. This was necessary to bound the degree of the output polynomial as well as the size of its coefficients. Here we do not deal with syntactic degree but independently bound the degree of the polynomial as well as the values of the coefficients.

We show in Theorem 1 below that with this definition, uniform VWIDTH(poly) coincides with uniform VPSPACE as defined in [9]; thus polynomial width indeed corresponds to polynomial space. Motivated by this equivalence, we define the following complexity classes:

Definition 2. $\mathsf{VSPACE}(S(n)) = \mathsf{VWIDTH}(S(n))$
Uniform-$\mathsf{VSPACE}(S(n))$= *Uniform*-$\mathsf{VWIDTH}(S(n))$

We denote the log-space class by VL; thus $\mathsf{VL} = \mathsf{VWIDTH}(\log n) = \mathsf{VSC}^1$.

The following containments and equalities follow directly from known results about width-constrained arithmetic circuits.

Lemma 1 ([5, 10, 11, 7]). $\mathsf{VBWBP} = \mathsf{VNC}^1 = \mathsf{VP}_e \subseteq \mathsf{VSPACE}(O(1)) = \mathsf{VSC}^0 \subseteq \mathsf{VL} = \mathsf{VSC}^1 \subseteq \mathsf{VP}$

Thus VL according to this definition is in VP and avoids the trivially "too-powerful" trap; also, it contains VNC^1 and thus avoids the "too weak" trap.

The following closure property is easy to see.

Lemma 2. *For every $S(n) > \log n$, the classes* $\mathsf{VSPACE}(S(n))$ *are closed under polynomially bounded summations and constant many products.*

3.3 Comparing VPSPACE and VWIDTH(poly)

This subsection is devoted to proving the following equivalence,

Theorem 1. *The class Uniform-*VPSPACE *([9]) and Uniform-*VWIDTH(poly) *are equivalent.*

The equivalence follows from the two lemmas below.

Lemma 3. *Uniform-*$\mathsf{VPSPACE} \subseteq$ *Uniform-*$\mathsf{VWIDTH}(\mathsf{poly})$.

The converse direction is a little more tedious, but essentially follows from the Lagrange interpolation formula for multivariate polynomials.

Lemma 4. *Uniform-*$\mathsf{VWIDTH}(\mathsf{poly}) \subseteq$ *Uniform-*$\mathsf{VPSPACE}$.

Lemma 4 requires that the VWIDTH family be uniform (with a direct-connection uniformity condition). If the VWIDTH family is non-uniform, this problem cannot be circumvented with polynomial advice, since the circuit has exp-size.

4 Read-Once certificates

In general, non-deterministic complexity classes can be defined via existential quantifiers. *e.g.* , $\mathsf{NP} = \exists \cdot P$. In the algebraic setting, Valiant ([15], [4]) introduced the class VNP (algebraic counterpart of NP) obtained as an "exponential" sum of values of a polynomial size arithmetic circuit. *i.e.* , $\mathsf{VNP} = \Sigma \cdot P$. Valiant also showed that $\mathsf{VNP} = \Sigma \cdot \mathsf{VP}_e$, which equals $\Sigma \cdot \mathsf{VNC}^1$.

If we consider smaller classes, NL is the natural non-deterministic version of L. However to capture it via existential quantifiers, we need to restrict the use of the certificate, since otherwise $\exists \cdot L = \mathsf{NP}$. It is known that with the notion of "read once" certificates (see, *e.g.* , [1], Chapter 4) one can express NL as an existential quantification over L. Analogously, we propose a notion of "read-once" certificates in the context of arithmetic circuits so that we can get meaningful classes by taking exponential sums over classes that are below VP.

Definition 3. *Let C be a layered arithmetic circuit with ℓ layers. Let $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ be the input variables of C. C is said to be "read-once certified" in Y if the layers of C can be partitioned into m blocks, such that each block reads exactly one variable from Y. That is, C satisfies the following:*

- *There is a fixed permutation $\pi \in S_m$ such that the variables of Y appear in the order $y_{\pi(1)}, \ldots, y_{\pi(m)}$ along any leaf-to-root path.*
- *There exist indices $0 = i_1 \leq \ldots \leq i_m \leq \ell$ such that the variable $y_{\pi(j)}$ appears only from layers $i_j + 1$ to i_{j+1}.*

We usually assume, without loss of generality, that π is the identity permutation.

Now we define the the exponential sum over read-once certified circuits.

Definition 4. *Let $\mathcal{C}$ be any arithmetic circuit complexity class. A polynomial family $(f_n)_{n\geq 0}$ is said to be in the class $\Sigma^R \cdot \mathcal{C}$, if there is a family $(g_n)_{n\geq 0}$ such that $f_n(X) = \sum_{Y \in \{0,1\}^{m(n)}} g_n(X, Y)$ and g_n is computed by a circuit of type $\mathcal{C}$ that is read-once certified in Y and $m(n) \leq \mathsf{poly}(n)$.*

We also use the term "read once exponential sum" over $\mathcal{C}$ to denote $\Sigma^R \cdot \mathcal{C}$.

For circuits of width polynomial or more, the restriction to read-once certification is immaterial: the circuit can read a variable once and carry its value forward to any desired layer via internal gates. This is equivalent to saying that for a P machine, read-once input is the same as two-way-readable input. Thus

Lemma 5. $\Sigma^R \cdot \mathsf{VP} = \Sigma \cdot \mathsf{VP} = \mathsf{VNP}$

Having seen that the read-once certificate definition is general enough for the case of large width circuits, we turn our focus on circuits of smaller width. Once the width of the circuit is substantially smaller than the number of bits in the certificate, the read-once property becomes a real restriction. If this restriction correctly captures non-determinism, we would expect that in analogy to $\mathsf{BP} = \mathsf{NL} = \Sigma^R \cdot \mathsf{L}$, we should be able to show that VBP equals $\Sigma^R \cdot \mathsf{VL}$. In a partial answer, we show in the following theorem one direction: read-once exponential sums over VL are indeed powerful enough to contain VBP.

Theorem 2. $\mathsf{VBP} \subseteq \Sigma^R \cdot \mathsf{VL}$.

In order to prove the above theorem, we consider a problem that is complete for VBP under projections. (see [4] for definition of a projection). Let $G_n = (V_n, E_n)$ (with $V_n = \{1, \ldots, n\}$) be the complete layered graph with variable $x_{i,j}$ as label

on the edge $(i,j) \in E_n$. Let $s = 1$ and $t = n$ denote two special nodes in G_n. Let $X = (x_{i,j})_{i,j \in \{1,\ldots,n\}}$. For any directed $s-t$ path $P = \langle v_0, v_1, \ldots, v_\ell, v_{\ell+1} \rangle$ in G_n, let M_P denote the monomial that is the product of the variables corresponding to edges in P. Let $PATH_G^n = \sum_P M_P$, where P is over all the $s-t$ paths in G_n.

Proposition 1. *(folklore)* $(PATH_G^n)_{n \geq 0}$ *is complete for* VBP *under projections.*

We prove theorem 2 by showing that $PATH_G^n \in \Sigma^R \cdot \mathsf{VL}$.

Proof (of theorem 2). Here onwards we drop the index n from G_n.

We define function $h_G(Y,Z) : \{0,1\}^{\lceil \log n \rceil} \times \{0,1\}^{n^2} \to \{0,1\}$ as follows. Assume that the variables in $Y = \{y_1, \ldots, y_k\}$ and $Z = \{z_{1,1}, \ldots, z_{n,n}\}$ take only values from $\{0,1\}$. $h_G(Y,Z) = 1$ if and only if $Z = z_{1,1}, \ldots, z_{n,n}$ represents a directed s-t path in G of length exactly ℓ, where ℓ written in binary is $y_1 \ldots y_k$, and Z reads off the entries of X in column-major order. Note that $s-t$ paths P in G are in one-to-one correspondence with assignments to Y, Z such that $h_G(Y,Z) = 1$. Hence

$$\begin{aligned} PATH_G^n = \sum_P M_P &= \sum_{Y,Z} h_G(Y,Z) \, [\text{ weight of path specified by } Y,Z] \\ &= \sum_{Y,Z} h_G(Y,Z) \prod_{i,j} (x_{i,j} z_{i,j} + (1 - z_{i,j})) \end{aligned}$$

There is a deterministic log-space algorithm A which computes $h_G(Y,Z)$ when Y, Z is given on a "read once" input tape (see [1]). Let C be the corresponding $\log n$ width boolean circuit. (w.l.o.g., all negation gates in C are at the leaves.) Call its natural arithmetization D. Since Y and Z are on a read-once input tape, it is easy to see that C, and hence D, are read-once certified in the variables from Y and Z. We can attach, parallel to D, constant-width circuitry that collects factors of the product $\prod_{i,j} (x_{i,j} z_{i,j} + (1 - z_{i,j}))$ as and when the $z_{i,j}$ variables are read, and finally multiplies this with $h_G(Y,Z)$. The resulting circuit remains $O(\log n)$-width, and remains read-once certified on Y, Z. □

While we are unable to show the converse, we are also unable to show a reasonable upper bound on $\Sigma^R \cdot \mathsf{VL}$. It is not even clear if $\Sigma^R \cdot \mathsf{VL}$ is contained in VP. One possible interpretation is that the Σ^R operator is too powerful and can lift up small classes unreasonably. We show that this is not the case in general; in particular, it does not lift up VBP and VBWBP.

Theorem 3. *1.* $\Sigma^R \cdot \mathsf{VBP} = \mathsf{VBP}$

2. $\Sigma^R \cdot \mathsf{VBWBP} = \mathsf{VBWBP}$

This theorem follows from Lemma 6 below, and from the facts that polynomial size weakly skew circuits can be transformed into (1) skew circuits of polynomial size([13]), and (2) skew circuits of polynomial size with a quadratic blowup in width ([7]) of . First, a definition.

Definition 5. *For $f \in \mathbb{K}[X, Y]$ with $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$, $E_Y(f)$ denotes the exponential sum of $f(X, Y)$ over all Boolean settings of Y. That is,*

$$E_Y(f)(X) = \sum_{e \subseteq \{0,1\}^m} f(X, e)$$

Lemma 6. *Let C be a layered skew arithmetic circuit on variables $X \cup Y$. Suppose C is* read-once *certified in Y. Let $w = \mathsf{width}(C)$, $s = \mathsf{size}(C)$ and $\ell =$ the number of layers in C. Let $f_1, \ldots, f_w$ denote the output gates (also the polynomials computed by them) of C. There exists a weakly skew circuit C', of size $O(mw^4 s)$ and width $4w$, that computes all the exponential sums $E_Y(f_1), \ldots, E_Y(f_w)$.*

Proof. We proceed by induction on $m = |Y|$. In the base case when $m = 1$, $E_Y(f_j)(X) = f_j(X, 0) + f_j(X, 1)$. Putting two copies of C next to each other, one with $y = 0$ and the other with $y = 1$ hardwired, and adding corresponding outputs, gives the desired circuit.

Assume now that the lemma is true for all skew circuits with $m' = |Y| < m$. Let C be a given circuit where $|Y| = m$. Let Y' denote $Y \backslash \{y_m\} = \{y_1, \ldots, y_{m-1}\}$. As per definition 3, the layers of C can be partitioned into m blocks, with the kth block reading only y_k from Y. Let $0 = i_1 \leq i_2 \leq \ldots \leq i_m \leq \ell$ be the layer indices such that y_k is read between layers $i_k + 1$ and i_{k+1}. Let $f_1, \ldots, f_w$ be the output gates of C.

We slice C into two parts: the bottom $m - 1$ blocks of the partition together form the circuit D, and the top block forms the circuit C_m. Let $g_1, \ldots, g_w$ be the output gates of D. These are also the inputs to C_m; we symbolically relabel the non-leaf inputs at level 0 and the outputs of C_m as $Z_1, \ldots Z_w$ and $h_1, \ldots, h_w$. Clearly, C_m and D are both skew circuits of width w. Further, each h_j depends on X, y_m and Z; that is, $h_1, \ldots, h_w \in R[Z_1, \ldots, Z_w]$ where $R = \mathbb{K}[X, y_m]$. Similarly, each g_j depends on X and Y'; $g_1, \ldots, g_w \in \mathbb{K}[X, Y']$. The values computed by C can be expressed as $f_j(X, Y) = h_j\left(X, y_m, g_1(X, Y'), \ldots, g_w(X, Y')\right)$.

Since C and C_m are skew circuits, and since the variables Z_j represent non-leaf gates of C, C_m must be linear in these variables. Hence each h_j can be written as $h_j(X, y_m, Z) = c_j + \sum_{k=1}^{w} c_{j,k} Z_k$, where the coefficients $c_j, c_{j,k} \in \mathbb{K}[X, y_m]$. Combining this with the expression for f_j, we have

$$\begin{aligned} f_j(X, Y) &= h_j\left(X, y_m, g_1(X, Y'), \ldots, g_w(X, Y')\right) \\ &= c_j(X, y_m) + \sum_{k=1}^{w} c_{j,k}(X, y_m) g_k(X, Y') \qquad \text{and hence} \end{aligned}$$

$$\begin{aligned} \sum_{e \in \{0,1\}^m} f_j(X, e) &= \sum_{e \in \{0,1\}^m} \left[c_j(X, e_m) + \sum_{k=1}^{w} c_{j,k}(X, e_m) g_k(X, e') \right] \\ &= 2^{m-1} \sum_{e_m = 0}^{1} c_j(X, e_m) + \sum_{k=1}^{w} \sum_{e \in \{0,1\}^m} c_{j,k}(X, e_m) g_k(X, e') \end{aligned}$$

$$= 2^{m-1} \sum_{e_m=0}^{1} c_j(X, e_m) + \sum_{k=1}^{w} \left(\sum_{e_m \in \{0,1\}} c_{j,k}(X, e_m) \right) \left(\sum_{e' \in \{0,1\}^{m-1}} g_k(X, e') \right)$$

$$\text{Thus } \; E_Y(f_j)(X) = 2^{m-1} E_{y_m}(c_j)(X) + \sum_{k=1}^{w} E_{y_m}(c_{j,k})(X) E_{Y'}(g_k)(X)$$

By induction, we know that there is a weakly skew circuit D' of width $4w$ and size $O((m-1)w^4 s)$ computing $E_{Y'}(g_k)(X)$ for all k simultaneously.

To compute $E_{y_m}(c_j)(X)$, note that a copy of C_m with all leaves labelled Z_k replaced by 0 computes exactly $c_j(X, y_m)$. So the sum can be computed as in the base case, in width $w+1$ and size $2(\mathsf{size}(C_m)+1)$. Multiplying this by 2^{m-1} in the standard way adds nothing to width and 2 to size, so overall width is $w+1$ and size is at most $2s+4$.

To compute $E_{y_m}(c_{j,k})(X)$, we modify C_m as follows: replace leaves labelled Z_k by the constant 1, replace leaves labelled $Z_{k'}$ for $k' \neq k$ by 0, leave the rest of the circuit unchanged, and let h_j be the output gate. This circuit computes $c_j(X, y_m) + c_{j,k}(X, y_m)$. Subtracting $c_j(X, y_m)$ (as computed above) from this gives $c_{j,k}(X, y_m)$. Now, the sum can be computed as in the base case. Again, to compute $E_{y_m}(c_{j,k})(X)$, we use two copies of the difference circuit with $y_m = 0$ and $y_m = 1$ hardwired, and add their outputs. It is easy to see that this circuit has width $w+2$ and size at most $4(w+2)\mathsf{size}(C_m) \leq 4(w+2)s$.

Putting together these circuits naively may increase width too much. So we position D' at the bottom, and carry w wires upwards from it corresponding to its w outputs. Alongside these wires, we position circuitry to accumulate the terms for each f_j and to carry forward already-computed f_k's. The width in this part is w for the wires carrying the outputs of D', w for wires carrying the values $E_Y(f_j)$, $w+2$ for computing the terms in the sum above (they are computed sequentially so the width does not add up), and 2 for computing partial sums in this process, overall at most $3w+4$. Thus the resulting circuit has width at most $\max\{\mathsf{width}(D'), 3w+4\} \leq 4w$.

To bound the size of the circuit, we bound its depth in the part above D' by d; then size is at most $\mathsf{size}(D') + \mathsf{width}^2 \times d$. The circuit has w modules to compute the $E_Y(f_j)$s. The depth of each module can be bounded by the depth to compute $E_{y_m}(c_j)$ plus w times the depth to compute any one $E_{y_m}(c_{j,k})$, that is, at most $(2s+4) + w^2 \times 4(w+2)s$. So $d \leq w^2(2s+4+4sw(w+2)) = \theta(w^4 s)$, and the size bound follows. □

5 Read-once exponential sums of some restricted circuits

In this section, we explore how far the result of Theorem 3 can be pushed to larger classes within VP. In effect, we ask whether the technique of Lemma 6 is

applicable to larger classes of circuits. Such a question is relevant because we do not have any bound (better than VNP) even for $\Sigma^R \cdot \mathsf{VSC}^0$ and $\Sigma^R \cdot \mathsf{VL}$.

One generalization we consider is multiplicative disjointness. An arithmetic circuit C is said to be multiplicatively disjoint (md) if every multiplication gate operates on sub-circuits which are not connected to each other. (See [13].) *i.e.* if $g = u \times v$ then the sub-circuits rooted at u and v are disjoint. Multiplicative disjointness generalises skewness and weak-skewness.

A further generalization is polynomial syntactic degree. Let C be an arithmetic circuit computing the polynomial $f \in \mathbb{K}[X]$. The *syntactic degree* of C is the degree of the polynomial $f' \in \mathbb{K}[X, y]$ computed by the circuit C' which is obtained by replacing all the leaves in C labelled by a constant from $\mathbb{K}$ with the variable y. The syntactic degree is an upper bound on the actual degree of the polynomial computed, though the two can differ significantly. All multiplicatively disjoint circuits have syntactic degree bounded by their size.

We add a prefix *md-* to denote the multiplicative disjoint version of a class. *e.g.* *md*-VSC^0 denotes class of all polynomials that are computed by constant width arithmetic circuits of polynomial size which are also multiplicatively disjoint. For $i \geq 0$, let VsSC^i denote the sub-class of families of polynomials in VSC^i whose witness circuits also have syntactic degree bounded by $\mathsf{poly}(n)$. Analogous classes sSC^i in the Boolean and counting worlds have been studied in [10].

Examining the proof of Lemma 6, we see that the main barrier in extending it to these larger classes is that when we slice C into D and C_m, C_m is no longer linear in the "slice variables" Z. However, for *md*-circuits, C_m is multilinear in Z. As far as computing the coefficients $c_{j,\alpha}$ goes, where α describes a multilinear monomial, this is not a problem; in [7], it is shown that for such circuits the coefficient function can be computed efficiently. There is a cost to pay in size because the number of multilinear monomials is much larger. To handle this, we modify the inductive step, slicing C not at the last block but at a level that halves the number of Y variables read above and below it. This works out fine for constant-width, but results in quasipolynomial blow-up in size for larger widths.

Formally, we show the following:

Lemma 7. *Let C be a layered multiplicatively disjoint circuit of width w and size s on variables $X \cup Y$. Let ℓ be the number of layers in C. Suppose C is read-once certified in Y. Let $f_1 \ldots, f_w$ be the output gates of C. Then, there is an arithmetic circuit C' of size $sm^{O(w)}$ which computes $E_Y(f_1), \ldots, E_Y(f_w)$.*

For VsSC circuits, the "upper half" circuit is not even multilinear. So we need to explicitly account for each monomial up to the overall degree, and compute the coefficient of each. We show that this is possible, if a quasipolynomial blow-up in size is allowed. Formally,

Lemma 8. *Let C be a layered arithmetic circuit size of s on the variables $X \cup Y \cup Z$. Let d be the syntactic degree bound on C and w be its width. Let $f \in R[Z]$ be a polynomial computed by C, where $R = \mathbb{K}[X, Y]$. Let $t = \langle t_1, \ldots, t_w \rangle$ be a degree sequence for variables from Z. Then $\mathsf{coeff}_f(Z^t)$ can be computed by a circuit of width $w + 2$ and size $O((d+1)^{2w})$, where $Z^t = \prod_{k=1}^{w} Z^{t_k}$.*

As a consequence of Lemmas 7, 8, we have the following:

Theorem 4. – $\Sigma^R \cdot md\text{-}\mathsf{VSC}^0 \subseteq \mathsf{VP}$.
– $\Sigma^R \cdot md\text{-}\mathsf{VSC} \subseteq \mathsf{VQP}$.
– *For all* $i \geq 0$, $\Sigma^R \cdot \mathsf{VsSC}^i \subseteq \mathsf{VQP}$.

References

1. S. Arora and B. Barak. *Complexity Theory: A Modern Approach*. To be published, 2009.
2. D. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.
3. L. Blum, F. Cucker, M. Shub, and S. Smale. *Complexity and Real Computation*. Springer, 1997.
4. P. Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*. Algorithms and Computation in Mathematics. Springer-Verlag, 2000.
5. H. Caussinus, P. McKenzie, D. Thérien, and H. Vollmer. Nondeterministic NC^1 computation. *Journal of Computer and System Sciences*, 57:200–212, 1998.
6. P. J. de Naurois. A Measure of Space for Computing over the Reals. In *CiE*, pages 231–240, 2006.
7. M. Jansen and B. V. R. Rao. Simulation of arithmetical circuits by branching programs preserving constant width and syntactic multilinearity. In *CSR*, 2009. To Appear.
8. P. Koiran and S. Perifel. VPSPACE and a Transfer Theorem over the Complex Field. In *MFCS*, pages 359–370, 2007.
9. P. Koiran and S. Perifel. VPSPACE and a Transfer Theorem over the Reals. In *STACS*, pages 417–428, 2007.
10. N. Limaye, M. Mahajan, and B. V. R. Rao. Arithmetizing classes around NC^1 and L. In *STACS*, pages 477–488, 2007. full version in ECCC TR07-087.
11. M. Mahajan and B. V. R. Rao. Arithmetic circuits, syntactic multilinearity and skew formulae. In *MFCS, LNCS vol. 5162*, pages 455–466, 2008. full version in ECCC TR08-048.
12. G. Malod. The complexity of polynomials and their coefficient functions. In *IEEE Conference on Computational Complexity*, pages 193–204, 2007.
13. G. Malod and N. Portier. Characterizing Valiant's algebraic complexity classes. In *MFCS*, pages 704–716, 2006.
14. C. Michaux. Une remarque à propos des machines sur $\mathbb{R}$ introduites par Blum, Shub et Smale. *Comptes Rendus de l'Académie des Sciences de Paris*, 309(7):435–437, 1989.
15. L. G. Valiant. Completeness classes in algebra. In *Symposium on Theory of Computing STOC*, pages 249–261, 1979.
16. H. Venkateswaran. Circuit definitions of nondeterministic complexity classes. *SIAM J. Comput.*, 21(4):655–670, 1992.
17. H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York Inc., 1999.