Balancing Bounded Treewidth Circuits

Maurice Jansen* and Jayalal Sarma M.N.*

Institute for Theoretical Computer Science,
Tsinghua University, Beijing, China.

Abstract. We use algorithmic tools for graphs of small treewidth to address questions in complexity theory. For both arithmetic and Boolean circuits, we show that any circuit of size $n^{O(1)}$ and treewidth $O(\log^i n)$ can be simulated by a circuit of width $O(\log^{i+1} n)$ and size n^c, where $c = O(1)$, if $i = 0$, and $c = O(\log \log n)$ otherwise. For our main construction, we prove that multiplicatively disjoint arithmetic circuits of size $n^{O(1)}$ and treewidth k can be simulated by bounded fan-in arithmetic formulas of depth $O(k^2 \log n)$. From this we derive an analogous statement for syntactically multilinear arithmetic circuits, which strengthens the central theorem of [14]. As another application, we derive that constant width arithmetic circuits of size $n^{O(1)}$ can be balanced to depth $O(\log n)$, provided certain restrictions are made on the use of iterated multiplication. Also from our main construction, we derive that Boolean bounded fan-in circuits of size $n^{O(1)}$ and treewidth k can be simulated by bounded fan-in formulas of depth $O(k^2 \log n)$. This strengthens in the non-uniform setting the known inclusion that $\mathrm{SC}^0 \subseteq \mathrm{NC}^1$. Finally, we apply our construction to show that Reachability and Circuit Value Problem for some treewidth restricted cases can be solved in LogDCFL.

1 Introduction

It is well-known that many hard graph theoretical problems become tractable when restricted to graphs of bounded treewidth[1]. If a graph with n nodes has bounded treewidth, there always exists a balanced tree decomposition of depth $O(\log n)$. This yields NC-algorithms for many problems, which are known to be NP-complete in general [6].

Consider the following question. Suppose one is given a circuit (Boolean or arithmetic) of size s and bounded fan-in, for which the underlying graph has bounded treewidth. Does this imply, as intuition might suggest, that there must exist an equivalent bounded fan-in circuit of size $poly(s)$ and depth $O(\log s)$? We show that in the Boolean case the situation is as expected, which yields the following theorem:

* This work was supported in part by the National Natural Science Foundation of China Grant 60553001, and the National Basic Research Program of China Grant 2007CB807900,2007CB807901. Email: maurice.julien.jansen@gmail.com, jayalal@tsinghua.edu.cn.

[1] For a definition see Section 2.

Theorem 1. *The class of languages accepted by non-uniform constant fan-in circuits of polynomial size and bounded treewidth equals non-uniform* NC^1.

Due to a celebrated result of Barrington [4], it is known that NC^1 can be simulated by constant width branching programs, which are skew circuits of constant width. A constant width circuit can be evaluated using $O(1)$ memory, and hence $\mathrm{SC}^0 = \mathrm{NC}^1$ (SC^0 is the class of Boolean functions computable by constant width circuits of *poly* size, c.f. [13]). Theorem 1 strengthens this statement in the non-uniform setting.

For arithmetic circuits, the short answer is that the equivalent circuit need not exist: a depth $O(\log s)$ circuit of bounded fan-in computes a polynomial of degree $s^{O(1)}$, but using repeated multiplication a bounded treewidth circuit of size s can easily compute a polynomial of degree 2^s. We rephrase the question to avoid this triviality.

The class of families of polynomials $\{p_n\}_{n\geq 1}$ of polynomial degree with polynomial size arithmetic circuits is known as VP (See e.g. [8]). In this paper, we let $\mathrm{VP}[tw = O(\log^i n)]$ stand for the class corresponding to polynomial size circuits of treewidth $O(\log^i n)$. Let VNC^1 denote the class corresponding to bounded fan-in arithmetic circuits of depth $O(\log n)$, which due to Brent's result [7] corresponds to poly-size arithmetic formulas. Our question becomes the following: is $\mathrm{VP}[tw = O(1)] \subseteq \mathrm{VNC}^1$?

One reason for considering this question, is that in case of an affirmative (and effective) answer it could be a useful tool for circuit building. Namely, one could during the design stage focus on using any of the well-known classes of bounded treewidth, and be guaranteed a nicely balanced formula can be obtained afterwards. Another reason, more on the complexity theoretic side, is that a flexible parameter like treewidth contributes to obtaining a more refined structural understanding of the difference between formulas and circuits, since it provides hierarchies of classes which bridge the two notions. Regarding this, it is a major open problem whether VNC_1 and VP are distinct. We only know of a separation in the multilinear world, due to the result by Raz [17].

Considering the notion of restricted treewidth circuits will also shed some light on issues regarding bounded width circuits. Arithmetic circuits of bounded treewidth provide a common generalization of formulas and of circuits of bounded width. One has the hierarchy of classes $\{\mathrm{VSC}^i\}_{i\geq 0}$, where VSC^i corresponds to arithmetic circuits of width $O(\log^i n)$ and size $n^{O(1)}$, for $i \geq 0$. This hierarchy is known to be sandwiched in between VNC_1 and VP. We prove the following (and the Boolean analogue):

Theorem 2.

1. $\mathrm{VSC}^0 \subseteq \mathrm{VP}[tw = O(1)] \subseteq \mathrm{VSC}^1$.
2. $\mathrm{VSC}^i \subseteq \mathrm{VP}[tw = O(\log^i n)] \subseteq \mathrm{VSC}^{i+1}[size = n^{O(\log\log n)}]$, *for any* $i \geq 1$.

Arithmetic circuit width is a fundamental notion, but it is still ill-understood in its relation to other resources. We currently do not know of any "natural" problems characterized by bounded width arithmetic circuit classes. However,

as of recently, the notion has gained renewed attention from several researchers. It more or less embodies a notion of space in the arithmetic setting (See [15]). Mahajan and Rao [14] study the class $\mathrm{VSC}^0[deg = n^{O(1)}]$, which is obtained from VSC^0 by requiring formal degrees of circuits to be $n^{O(1)}$. Arvind, Joglekar, and Srinivasan [2] give lower bounds for *monotone* arithmetic circuits of constant width.

To make progress on the basic question whether $\mathrm{VP}[tw = O(1)] \subseteq \mathrm{VNC}^1$, we show that *multiplicatively disjoint*[2] circuits of size $n^{O(1)}$ and bounded treewidth can be simulated by bounded fan-in formulas of depth $O(\log n)$. In our notation this is stated as follows:

Theorem 3. $\text{md-VP}[tw = O(1)] = \mathrm{VNC}^1$.

Notice that without the treewidth restriction multiplicatively disjoint circuits of size $n^{O(1)}$ are known to compute all of VP [16]. From the above result, we derive the analogous statement for *syntactically multilinear* circuits. The resulting formulas will be syntactically multilinear (denoted with the prefix sm-) as well. This implies the lower bounds by Raz [17] hold all the way up to syntactically multilinear bounded treewidth circuits. We prove

Theorem 4. $\text{sm-VP}[tw = O(1)] = \text{sm-VNC}^1$.

Theorem 4 strengthens the main result of Mahajan and Rao [14], which states that *poly* size syntactically multilinear circuits of constant width can be simulated by *poly* size circuits of *log* depth (but it was explicitly left open whether the latter could be ensured to be syntactically multilinear). Considering the more general notion of treewidth in a way simplifies the proof due to the help of well-established algorithmic tools. We also remark that Theorem 3 strengthens Theorem 4 in [10], which states that $\text{md-VSC}^0 = \mathrm{VNC}^1$.

In [14] the fundamental question is raised whether $\mathrm{VSC}^0[deg = n^{O(1)}] \subseteq \mathrm{VNC}^1$. As was mentioned, they show this holds under the restriction of syntactic multilinearity. To make progress, we demonstrate a different restriction under which an efficient simulation by arithmetic $O(\log n)$ depth formulas is achievable. We apply Theorem 3 to give the following result for circuits with *bounded iterated multiplication chains* (For a definition see Section 4):

Theorem 5. *Constant width arithmetic circuits of size $n^{O(1)}$ with constant bounded iterated multiplication chains can be simulated by fan-in two arithmetic formulas of depth $O(\log n)$.*

As two additional applications of the above results, we consider the Circuit Value Problem (CVP) and the Reachability problem. Given the encoding of a Boolean circuit C and an input x the Circuit Value Problem is to test if $C(x) = 1$ or not. The general version of this problem is known to be P-complete. Several variants of this has been studied (See [9, 3, 12] and the references therein).

Given a (directed or undirected) graph $G = (V, E)$ and $s, t \in V$, reachability asks to test if t is reachable from s in G. reachability captures space

[2] We indicate this with the prefix md-. See Section 2 for a definition.

bounded computation in a natural way. For directed graphs it is complete for NL [11, 19]. The case of undirected graphs was settled recently by Reingold [18] by giving a log-space algorithm for the problem. This shows the problem is complete for L. There has been extensive research aimed at settling the complexity of testing reachability on restricted graphs (see [1] and references therein).

LogCFL and LogDCFL are the classes of languages that are logspace many-one reducible to non-deterministic and deterministic context-free languages, respectively. LogDCFL can be also characterized as the class of languages that can be recognized by a logspace Turing machine that is also provided with a stack, which runs in polynomial time. It follows by definition that $\mathrm{L} \subseteq \mathrm{LogDCFL} \subseteq \mathrm{LogCFL}$ and $\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{LogCFL}$. However, it is unknown how NL and LogDCFL can be compared. In essense, this asks for a trade-off, trading non-determinism with stack access. Given that directed reachability is an NL-complete problem, giving a LogDCFL upper bound achieves such a trade-off for a restricted class of NL-computations.

We prove the following theorem for the CVP and obtain a corollary for Reachability.

Theorem 6. CVP *for bounded treewidth and bounded fan-in circuits when the input also contains the tree decomposition, is in* LogDCFL.

Corollary 1. Reachability *for directed acyclic graphs of bounded treewidth and in-degree is in* LogDCFL*, provided the tree decomposition is given at the input.*

2 Preliminaries

We briefly recall basic circuit definitions. A *Boolean circuit* is a directed acyclic graph, with labels $\{0, 1, x_1, \ldots, x_n, \wedge, \vee, \neg\}$ on its nodes. Nodes with label from $\{0, 1, x_1, \ldots, x_n\}$ are called *input gates*, and designated nodes of zero out-degree are called the *output gates*. The *fan-in* of a gate is its in-degree. Formulas are circuits for which the out-degree of each gate is at most one. For size of a circuit we count the number of non-input gates. The depth is measured as the length of a longest directed path. Fan-in is assumed to be bounded. As in [14, 10], when we speak about the width of a circuit, we assume the circuit is layered, and it is taken to be the maximum number of nodes on a layer. Let us emphasize that we allow input gates (constant or variable labeled) to appear at all layers. The class NC^1 is the class of boolean functions on n bits which can be computed by boolean circuits of depth $O(\log n)$ and size $n^{O(1)}$. SC^i denotes the class of functions computed by polynomial size circuits of width $O(\log^i n)$.

For *arithmetic circuits* over a ring R, nodes are labeled by ring constants, formal variables from a set X, and $\{+, \times\}$. We assume that the fan-in is bounded by two. The output of an arithmetic circuit is a polynomial in the ring $R[X]$, defined in the obvious way. The size of a circuit is taken to be the number of $\{+, \times\}$-gates. For a circuit Φ with designated output gate f, the polynomial computed by the output gate is denoted with $\lceil \Phi \rceil$. We denote the set of variables

used in Φ by $Var(\Phi)$. Similarly we use $Var(p)$, if p is a polynomial. Note that $Var(\lceil\Phi\rceil) \subseteq Var(\Phi)$. We call a polynomial f *multilinear* in some subset of the variables S, if the individual degree is at most one in f, for each variable in S (even if f has a constant term). An arithmetic circuit is called *syntactically multilinear* if for each multiplication gate the subcircuits originated at its inputs carry disjoint sets of variables. For a *multiplicatively disjoint* circuit, for every gate $f = g \times h$, the sub-circuits rooted at g and h are disjoint (as graphs). The *formal degree* of a circuit is defined inductively by taking variable and constant labeled gates to be of degree one. For addition gates one takes the maximum of the degrees of its inputs. For multiplication gates one takes the sum of the degrees. The degree of the circuit is taken to be the maximum degree of a gate.

For a *p-family of polynomials* $\{f_m\}_{m\geq 1}$, we have $f_m \in R[x_1, x_2, \ldots, x_{p(m)}]$, and $\deg(f_m) \leq q(m)$, for some polynomials p and q. Arithmetic circuit classes contain p-families. VP and VP_e are the classes of p-families computable by arithmetic circuits and formulas, respectively, of size $n^{O(1)}$ (See e.g. [8]). For $i \geq 0$, VSC^i is the class of all p-families computable by arithmetic circuits of width $O(\log^i n)$ and size $n^{O(1)}$. In [14] the class a-sSC^i is considered, which corresponds to width $O(\log^i n)$ circuits of size and formal degree $n^{O(1)}$. We will denote this class by $\mathrm{VSC}^i[deg = n^{O(1)}]$. The class VNC^i is the set of all p-families computable by arithmetic circuits of depth $O(\log^i n)$ and size $n^{O(1)}$.

Next we define various graph parameters. The *width* of a layered graph is the maximum number of vertices in any particular layer. A *tree decomposition* of a graph $G = (V, E)$ is given by a tuple $(T, (X_d)_{d\in V[T]})$, where T is a tree, each X_d is a subset of V called a *bag*, satisfying 1) $\bigcup_{d\in V[T]} X_d = V$, 2) For each edge $(u, v) \in E$, there exists a tree node d with $\{u, v\} \subseteq X_d$, and 3) For each vertex $u \in V$, the set of tree nodes $\{d : u \in X_d\}$ forms a connected subtree of T. Equivalently, for any three vertices $t_1, t_2, t_3 \in V[T]$ such that t_2 lies in the path from t_1 to t_3, it holds that $X_{t_1} \cap X_{t_3} \subseteq X_{t_2}$.

The *width* of the tree decomposition is defined as $\max_d |X_d| - 1$. The *treewidth* $\mathrm{tw}(G)$ of a graph G is the minimum width of a tree decomposition of G. For a rooted tree T, let $X_{\leq t} = \cup_{u\in S_t} X_u$, with $S_t = \{u : u = t \text{ or } t \text{ is an ancestor of } u\}$.

Lemma 1. *(Theorem 4.3 in [6]) Let $G = (V, E)$ be a graph with $|V| = n$ and treewidth at most k. Then G has a tree decomposition $(T, (X_d)_{d\in V[T]})$ of width $3k + 2$ such that T is a binary tree of depth at most $2\lceil \log_{\frac{5}{4}} n\rceil$.*

The following proposition is left as an easy exercise:

Proposition 1. *A leveled graph G of width k has treewidth at most $2k - 1$.*

3 Arithmetic Circuits of Bounded Treewidth

The treewidth of a circuit with underlying graph G is defined to be $\mathrm{tw}(G)$. Note that a circuit has treewidth 1 if and only if it is a formula. We introduce the class $\mathrm{VP}[tw = O(\log^i n)]$ as the class of p-families of polynomials $\{f_n\}_{n\geq 1}$ that can be computed by fan-in two arithmetic circuits of size $n^{O(1)}$ and treewidth

$O(\log^i n)$. As Theorem 2 states, these classes interleave (roughly) with the VSC^i classes. We postpone the proof of Theorem 2 as it uses developments of our main construction.

Theorem 7. *For any multiplicatively disjoint arithmetic circuit Φ of size s and treewidth k, there exists an equivalent formula Γ of size at most $s^{O(k^2)}$.*

Proof. Let Φ be a multiplicatively disjoint circuit of size s, and let $(T, (X_t)_{t\in V[T]})$ be a tree decomposition of Φ of width k. By Lemma 1, we can assume that T is a rooted binary tree of depth $d = O(\log s)$. We first preprocess T and Φ using Proposition 2 (Proof will appear in full version).

Proposition 2. *For every circuit Φ of size s, that has a tree decomposition $(T, (X_t)_{t\in V[T]})$ of width k and depth d, there exists a circuit Φ' of size at most $2s$, for which $\lceil\Phi\rceil = \lceil\Phi'\rceil$, with tree decomposition $(T', (X'_t)_{t\in V[T']})$ of width at most $k' = 3k+2$ and depth at most d, so that for any $t \in T'$, for any non-input gate $g \in X'_t$ with inputs g_1 and g_2, either both $g_1, g_2 \in X'_t$ or both $g_1, g_2 \notin X'_t$. In the latter case it holds that $g_1 \notin X'_{\leq t}$ iff $g_2 \notin X'_{\leq t}$.*

We assume wlog. that Φ and $(T, (X_t)_{t\in V[T]})$ satisfy the conditions of Proposition 2, as the increase in k and s due to preprocessing does not affect the bound we are aiming for. For any tree node $t \in T$ and $f \in X_t$, we define a circuit Φ_t, which is obtained from the subgraph $\Phi[X_{\leq t}]$, by turning all $g \in X_t$ that take both inputs from gates not in $X_{\leq t}$ into input gates with label z_g. For any $f \in X_t$, let $\Phi_{t,f}$ be the subcircuit of Φ_t rooted at gate f. At most $k+1$ new z-variables will be used at the tree node t. Crucially, observe that, since Φ is multiplicatively disjoint, any gate in $\Phi_{t,f}$ computes a polynomial that is multilinear in $\overline{z}$.

We will process the tree decomposition going bottom up. At a node t, we want to compute for each $f \in X_t$ a formula $\Gamma_{t,f}$ equivalent to $\Phi_{t,f}$. Wlog. we assume that the output gate of Φ is contained in X_r, for the root r of T. Hence, when done, we have a formula equivalent to Φ. In order to keep the size of the computed formulas properly bounded, we require a constant bound on the number of appearances of a z-variable in $\Gamma_{t,f}$. We achieve this by brute-force with Proposition 3, at the cost of blowing up the size by a factor of 2^{k+1}. To verify its correctness, observe that the lhs. and rhs. are multilinear polynomial in $F[\overline{x}][z_1, z_2, \ldots, z_{k+1}]$ taking identical values on $\{0,1\}^{k+1}$, and hence must be identical.

Proposition 3. *For any $f(\overline{x}, z_1, z_2, \ldots, z_{k+1})$ that is multilinear in $\overline{z}$, we have that $f = \sum_{b\in\{0,1\}^{k+1}} \left(\prod_{i\in[k+1]} (1-z_i)^{1-b_i} z_i^{b_i}\right) f(\overline{x}, b_1, b_2, \ldots, b_{k+1})$.*

The recursive procedure for computing the desired formula equivalent to $\Phi_{t,f}$ is given by Algorithm 1. Formally, for any $t \in T$, and $f \in X_t$, let $\Gamma_{t,f}$ be the formula output by the procedure call $Traceback(t, f)$. The following lemma proves its correctness:

Lemma 2. *For any $t \in T$, and any $f \in X_t$, $\lceil\Gamma_{t,f}\rceil = \lceil\Phi_{t,f}\rceil$.*

Proof. The proof will proceed by structural induction both on T and Φ. The statement can be easily verified for the two base cases: if t is a leaf of T, or f is an input gate in Φ. For the induction step, suppose t has children t_0 and t_1, and say $f = f_0 \circ f_1$, with $\circ \in \{+, \times\}$. We ignore line 17 of the procedure $Traceback$, since it does not modify the output of the computed formula.

In case both $f_0, f_1 \in X_t$, by induction hypothesis, $\lceil \Gamma_{t,f_0} \rceil = \lceil \Phi_{t,f_0} \rceil$ and $\lceil \Gamma_{t,f_1} \rceil = \lceil \Phi_{t,f_1} \rceil$. Observe that in this case $Traceback(t, f)$ returns $\Gamma_{t,f_0} \circ \Gamma_{t,f_1}$, so $\lceil \Gamma_{t,f} \rceil = \lceil \Gamma_{t,f_0} \circ \Gamma_{t,f_1} \rceil = \lceil \Gamma_{t,f_0} \rceil \circ \lceil \Gamma_{t,f_1} \rceil = \lceil \Phi_{t,f_0} \rceil \circ \lceil \Phi_{t,f_1} \rceil = \lceil \Phi_{t,f} \rceil$.

Now assume not both $f_0, f_1 \in X_t$. By Proposition 2, this means $f_0 \notin X_t$ and $f_1 \notin X_t$. Furthermore, we either have $f_0, f_1 \in X_{\leq t}$, or $\{f_0, f_1\} \cap X_{\leq t} = \emptyset$. In the latter case, $\lceil \Phi_{t,f} \rceil = z_f$, which is exactly what is returned by $Traceback(t, f)$. In the former case, say $f_0 \in X_{\leq t_{i_1}}$ and $f_1 \in X_{\leq t_{i_2}}$, for $i_1, i_2 \in \{0, 1\}$. Observe that by the tree decomposition properties $f \in X_{t_{i_1}}$, which makes the call of $Traceback(t_{i_1}, f)$ on line 11 valid. Note that $f_0 \notin X_{\leq t_{i_2}}$ and $f_1 \notin X_{\leq t_{i_1}}$, if $i_1 \neq i_2$. Hence, by the tree decomposition properties, if $i_1 \neq i_2$, there would exist a node t' with t_1 as ancestor such that $f, f_0 \in X_{t'}$, but $f_1 \notin X_{t'}$. Due to Proposition 2 this case does not arise.

The algorithm first computes $\Gamma = Traceback(t_{i_1}, f)$. By the induction hypothesis $\lceil \Gamma \rceil = \lceil \Phi_{t_{i_1},f} \rceil$. In $\Phi_{t_{i_1},f}$, whenever a gate g takes an input from a gate not in $X_{\leq t_{i_1}}$, i.e. by Proposition 2 this means both its inputs are not in $X_{\leq t_{i_1}}$, it appears as input node with label z_g. However, for the circuit $\Phi_{t,f}$ node g roots $\Phi_{t,g}$. Observe that this means that substituting $\lceil \Phi_{t,g} \rceil$ for each $z_g \in Var(\lceil \Phi_{t_{i_1},f} \rceil)$ in $\lceil \Phi_{t_{i_1},f} \rceil$ yields $\lceil \Phi_{t,f} \rceil$. Observe that the tree decomposition properties give us that $g \in X_t$, whenever we make the call on line 13 to compute Γ', and hence that this call is valid. By the induction hypothesis, $\lceil \Gamma' \rceil = \lceil \Phi_{t,g} \rceil$. Hence replacing, for all $z_g \in Var(\lceil \Gamma \rceil)$, each gate in Γ labeled with z_g by the formula Γ' gives a new formula Γ satisfying $\lceil \Gamma \rceil = \lceil \Phi_{t,f} \rceil$. □

We must bound the size of the formula $\Gamma_{t,f}$. The proof of the following lemma will appear in the full version of the paper.

Lemma 3. *Let $t \in T$ be a node at height h, then for any $f \in X_t$, $\Gamma_{t,f}$ has at most $\alpha^h 2^{k+1}$ many gates, where $\alpha = 2^{3k^2+9k+6}$.*

Since T has depth $O(\log s)$, we conclude the final formulas given at the root of T will be of size $s^{O(k^2)}$. □

The proof of Theorem 3 is now clear. Trivially $\text{VP}_e \subseteq \text{md-VP}[tw = O(1)]$. The converse follows from Theorem 7. Now use the fact that $\text{VP}_e = \text{VNC}^1$ [7].

3.1 Proof of Theorem 4

Observe that $\text{sm-VNC}^1 \subseteq \text{sm-VP}_e \subseteq \text{sm-VP}[tw = O(1)]$. For the other direction, let Φ be a syntactically multilinear circuit of treewidth k. We first modify it so that any gate g computing a field constant α is replaced by an input gate g' labeled with α. This can be done by removing edges fanning into g and

Algorithm 1 Recursive procedure for computing $\Gamma_{t,f}$

procedure $Traceback(t \in T,\ f \in X_t)$
if t is a leaf or f is an input gate in Φ **then**
  return a formula equivalent to $\Phi_{t,f}$ of size at most 2^{k+1} computed by 'brute force'.
else
  let t_0 and t_1 be the children of t in T, and say $f = f_0 \circ f_1$, with $\circ \in \{+, \times\}$.
  if both f_0 and f_1 are in X_t **then**
    let $\Gamma = Traceback(t, f_0) \circ Traceback(t, f_1)$.
  else
    // Neither f_0 nor f_1 is in X_t, by pre-processing.
    If f_0 and f_1 are not in $X_{\leq t}$ **return** a single node with label z_f. Otherwise, say $f_0 \in X_{\leq t_{i_1}}$ and $f_1 \in X_{\leq t_{i_2}}$, for $i_1, i_2 \in \{0, 1\}$.
    $\Gamma = Traceback(t_{i_1}, f)$.
    for all $z_g \in Var(\lceil \Gamma \rceil)$ **do**
      let $\Gamma' = Traceback(t, g)$.
      replace any gate in Γ labeled with z_g by the formula Γ'.
    end for
  end if
  Process Γ to make any z-variable occur at most 2^{k+1} times using Proposition 3.
  return Γ.
end if

relabeling. Hence the treewidth of the modified circuit is at most k. Next, any gate g labeled with a field constant α, with edges going to gates $f_1, f_2, \ldots, f_m$, is replaced by m separate copies of $g_1, g_2, \ldots, g_m$, each labeled with α, where we add edges (g_i, f_i), for all $i \in [m]$. This does not increase the treewidth, as it can be thought of as a two step procedure, neither of which increases treewidth: first removing the vertex g and attached edges, secondly, adding back the isolated copies. Observe that now we have obtained an equivalent circuit Φ' that is multiplicatively disjoint. Namely, for purpose of contradiction, suppose there exists a multiplication gate $f = f_1 \times f_2$ such that both f_1 and f_2 are reachable from some gate h. Then there exists such an h for which the paths to f_1 and f_2 are edge disjoint. For this h, since Φ' is syntactically multilinear, there cannot be variables in the subcircuit Φ'_h. Hence h is a gate computing a constant. Since the paths to f_1 and f_2 are edge disjoint, h must have out-degree at least two. This contradicts the fact that any gate computing a constant in Φ' has out degree one. The statement sm-VP$[tw = O(1)] \subseteq$ VNC$_1$ now follows from Theorem 7 and the fact that VP$_e$ = VNC^1 [7].

To get the strengthened conclusion that sm-VP$[tw = O(1)] \subseteq$ sm-VNC$_1$, we will now indicate how to modify Algorithm 1 to ensure syntactic multilinearity. We use the notation of the proof of Theorem 7. Assume we have done preprocessing as indicated above. We know each circuit $\Phi_{t,f}$ is syntactically multilinear, for all $t \in T$, and $f \in X_t$. The goal is to establish inductively that each $\Gamma_{t,f}$ is syntactically multilinear, for all $t \in T$, and $f \in X_t$.

At the base case, i.e. line 3 of Algorithm 1, we can simply enforce the condition by brute force. At line 7, by induction Γ_{t,f_0} and Γ_{t,f_1} are syntactically multilinear. If $\circ = +$, then so is Γ. In case $\circ = \times$, whenever the formulas Γ_{t,f_0} and Γ_{t,f_1} share a variable α, since we know $\lceil \Gamma \rceil = \lceil \Phi_{t,f} \rceil$ is multilinear, α does not appear in at least one of the polynomials $\lceil \Gamma_{t,f_0} \rceil$ and $\lceil \Gamma_{t,f_1} \rceil$. Setting α to zero in the corresponding formula ensures Γ is syntactically multilinear.

We now argue how to correctly deal with the substitution on line 14, and the processing of z variables on line 17. Consider Γ as computed on line 11. We want to ensure it is in the following standard form: $\sum_{a\in\{0,1\}^{k+1}} \left(\prod_{i\in[k+1]} z_i^{a_i}\right) f_a(\overline{x})$, for certain polynomials $f_a \in F[X]$. For this we use the following modification of Proposition 3, which is obtained by multiplying out the factors $\prod_{i\in[k+1]}(1 - z_i)^{1-b_i} z_i^{b_i}$. For $a, a' \in \{0,1\}^{k+1}$, we say $a' \leq a$ iff $\{i : a'_i = 1\} \subseteq \{i : a_i = 1\}$. We denote the size of $\{i : a'_i = 1\}$ by $|a'|$. We leave the proof of the following proposition as an exercise:

Proposition 4. *Let $f(\overline{x}, z_1, z_2, \ldots, z_{k+1})$ be given that is multilinear in $\overline{z}$. Write $f(\overline{x}, z_1, z_2, \ldots, z_{k+1}) = \sum_{a\in\{0,1\}^{k+1}} \left(\prod_{i\in[k+1]} z_i^{a_i}\right) coef(f, z_1^{a_1} z_2^{a_2} \ldots z_{k+1}^{a_{k+1}})$, then it holds that $coef(f, z_1^{a_1} z_2^{a_2} \ldots z_{k+1}^{a_{k+1}}) = \sum_{a'\leq a} (-1)^{|a|-|a'|} f(\overline{x}, a')$.*

If we use the above proposition to process z-variables on line 17, then by induction, Γ on line 11 will indeed have the required form, or for simplicity one can also assume we do an extra step of z-variable processing. That is, assume we apply above proposition to get Γ in the required form. This requires at most $(2^{k+1})^2$ copies of Γ and blows up Γ by an inconsequential factor of $2^{O(k)}$. Observe that this leaves Γ syntactically multilinear.

Now consider line 14. First of all, any $z_g \in Var(\Gamma) \backslash Var(\lceil \Gamma \rceil)$ can be set to zero in Γ. For the remaining z-variables, we claim that for any pair $z_g, z_h \in Var(\lceil \Gamma \rceil)$, whenever $\Gamma_{t,g}$ and $\Gamma_{t,h}$ share a variable α, then $coef(\lceil \Gamma \rceil, m) = 0$, for any multilinear monomial m in the z-variables of Γ that contains both z_g and z_h. Hence we can remove these terms from the standard form of Γ, and avoid multilinearity conflicts among products between each of the substituted formulas.

We will verify this claim using the notion of a *proof tree*. A proof tree rooted at a gate g in a circuit C is any tree obtained by recursively selecting gates, starting with g, as follows: 1) at an addition gate select exactly one of its children, and 2) at a multiplication gate select both children. We will consider proof trees of $\Phi_{t_{i_1},f}$ rooted at f. For a subset Z of z-variables in $\Phi_{t_{i_1},f}$, we let $\mathrm{PTree}(Z)$ stand for the collection of proof trees rooted at f that have precisely the z-variables in Z appearing at its leaves. Given $T \in \mathrm{PTree}(Z)$, let $p(T)$ denote the product of all X variables appearing in T. The following proposition is easily proved by structural induction on the circuit $\Phi_{t_{i_1},f}$.

Proposition 5. *For any multilinear monomial m in z-variables used in $\Phi_{t_{i_1},f}$, it holds that $coef(\lceil \Phi_{t_{i_1},f} \rceil, m) = \sum_{T\in PTree(Z)} p(T)$, where Z is the set of z-variables of m.*

Recall that by induction $\lceil \Gamma \rceil = \lceil \Phi_{t_{i_1},f} \rceil$. Now consider any multilinear monomial m in z-variables of $\lceil \Phi_{t_{i_1},f} \rceil$ with both z_g and z_h in it, where $\Gamma_{t,g}$ and $\Gamma_{t,h}$ share a variable α. For purpose of contradiction suppose $coef(\lceil \Phi_{t_{i_1},f} \rceil, m) \neq 0$. By Proposition 5 this means there exists a proof tree in $\Phi_{t_{i_1},f}$ rooted at f that contains both z_g and z_h. This implies g and h are reachable from a single multiplication gate r in $\Phi_{t_{i_1},f}$, and hence also in $\Phi_{t,f}$. Observe that our construction satisfies the property that for any $t \in V[T]$ and $f \in X_t$, $Var(\Gamma_{t,f}) \subseteq Var(\Phi_{t,f})$. Hence α appears in both $\Phi_{t,g}$ and $\Phi_{t,h}$. Observe that both α's must be reachable from r in $\Phi_{t,f}$. This contradicts the fact that $\Phi_{t,f}$ is syntactically multilinear.

Similarly, one can verify that whenever for a variable $z_g \in Var(\lceil \Gamma \rceil)$, the formula $\Gamma_{t,g}$ contains a variable α, then $coef(\lceil \Gamma \rceil, m)$ does not contain α for any monomial m containing z_g. Hence any occurrence of α in the formula $\sum_{a' \leq a} (-1)^{|a|-|a'|} \Gamma(\overline{x}, a')$ used to compute $coef(\lceil \Gamma \rceil, m)$ can be replaced by zero.

We conclude that under above modifications, Algorithm 1 yields a syntactically multilinear formula $\Gamma_{t,f}$ equivalent to $\Phi_{t,f}$. The proof is completed with the observation of [14] that Brent's construction [7], which shows $\mathrm{VP}_e \subseteq \mathrm{VNC}^1$, preserves syntactic multilinearity. □

3.2 Evaluation over a Finite Field and Boolean Implications

The observation is that Algorithm 1, when applied over $GF(2)$ to an arbitrary n-input arithmetic circuit Φ, will result in a formula Γ such that for any $a \in GF(2)^n$, $\lceil \Phi \rceil(a) = \lceil \Gamma \rceil(a)$. For this, no assumptions regarding the multiplicative disjointness of Φ is needed. One can prove this condition using structural induction similarly as in Lemma 2. For the processing of the z-variables on line 17, observe that we have the following adaption of Proposition 3:

Proposition 6. *Let $f(x_1, \ldots, x_n, z_1, \ldots, z_{k+1})$ be a polynomial over $GF(2)$, and let $g = \sum_{b \in \{0,1\}^{k+1}} \left(\prod_{i \in [k+1]} (1 - z_i)^{1-b_i} z_i^{b_i} \right) f(x_1, x_2, \ldots, x_n, b_1, b_2, \ldots, b_{k+1})$. Then for any $a \in GF(2)^{n+k+1}$, $f(a) = g(a)$.*

One can generalize this to an arbitrary finite field F of size q, by similarly using brute force on line 17 of Algorithm 1 to make sure any z-variables appears at most q^{k+1} times in Γ. Consequently, we have the following theorem:

Theorem 8. *Let F be a finite field, and let $q = |F|$. For any arithmetic circuit Φ over F of size s and treewidth k, there exists a formula Γ over F of size at most $s^{O(k^2 \log q)}$ such that Φ and Γ evaluate to identical values for inputs in F.*

A proof of the following proposition will appear in the full version.

Proposition 7. *For every Boolean circuit C of fan-in two and treewidth k, there is an arithmetic circuit C' over $GF(2)$ of treewidth $3k$ such that $\forall x \in \{0,1\}^n$, $C(x) = 1$ if and only if $C'(x) = 1$.*

Proof of Theorem 1 Given a Boolean circuit of size s and treewidth k of bounded fan-in (wlog. assume fan-in two), first convert it into an arithmetic circuit over $GF(2)$ using Proposition 7. Now apply Theorem 8 to obtain an arithmetic formula Γ over $GF(2)$ of size $s^{O(k^2)}$. Balance this formula down to depth $O(k^2 \log s)$ using [7]. Now do the reverse construction of arithmetization and code out an $\{\wedge, \vee, \neg\}$-formula computing the same function. The final circuit has depth $O(k^2 \log s)$. Thus we have proven Theorem 1. □

We can use a similar reduction to derive a Boolean analogue of Theorem 2. The proof will appear in the full version of the paper. Let TWC^i denote the class of Boolean functions computed by Boolean circuits of treewidth $O(\log^i n)$.

Theorem 9. *The following two statements hold in the non-uniform setting: 1)* $\mathrm{SC}^0 \subseteq \mathrm{TWC}^0 \subseteq \mathrm{SC}^1$*, and 2)* $\forall i \geq 1, \mathrm{SC}^i \subseteq \mathrm{TWC}^i \subseteq \mathrm{SC}^{i+1}[size = n^{O(\log \log n)}]$.

4 Constant Width Circuits

We make the following definition: an *iterated multiplication chain* of length ℓ in a circuit Φ is given by a sequence of gates $g_0, g_1, \ldots, g_\ell$, where all are multiplication gates, except possibly g_0, such that both inputs of g_i are reachable from g_{i-1}, for all $i \in [\ell]$. We denote the length of a longest iterated multiplication chain in Φ by $\mathcal{M}(\Phi)$. Note that if $\mathcal{M}(\Phi) = 0$, then Φ is multiplicatively disjoint. The following theorem will be proved in the full version of the paper:

Theorem 10. *For any leveled arithmetic circuit Φ of size s and width w, there exists equivalent formula of depth $d = O(w^4 \mathcal{M}(\Phi) \log s)$ and size at most 2^d.*

Theorem 5 immediately follows from Theorem 10. Note that conversely one has the inclusion $\mathrm{VNC}^1 \subseteq \mathrm{VSC}^0[\mathcal{M} = O(1)]$, due to [5].

5 Application to Circuit Evaluation and Reachability

We use Proposition 7 to reduce the proof of Theorem 6 to the following proposition. We note this reduction can be computed within *logspace*.

Proposition 8. *Given an arithmetic circuit C over $GF(2)$ together with its tree decomposition $(T, (X_d)_{d \in V[T]})$ of constant width k and an input $x \in \{0,1\}^n$, testing whether $C(x) = 1$ can be done in* LogDCFL.

Proofsketch. The proof proceeds by analyzing $Traceback$. We are given the circuit C and an input x. We replace each gate of C labeled by x_i with its Boolean value. Next we run $Traceback$ to compute an equivalent formula. We claim this can be implemented in *poly*-time and $O(\log n)$ workspace, provided we use a stack (whose space usage is not counted towards the space bound). This claim will be substantiated in the paper's full version. □

A proof of following proposition will appear in the full version of the paper. Corollary 1 follows from it.

Proposition 9. *Given a DAG $G = (V, E)$ of bounded treewidth and bounded in-degree and $s, t \in V$, we can obtain a circuit C of bounded treewidth and an input x such that $C(x) = 1$ if and only if t is reachable from s in the graph G.*

References

1. E. Allender. Reachability problems: An update. In *Computation and Logic in the Real World*, volume 4497 of *LNCS*, pages 25–27. 2007.
2. V. Arvind, P. Joglekar, and S. Srinivasan. On lower bounds for constant width arithmetic circuits. In *Proc. of ISAAC 2009*, LNCS 5878, pages 637–646, 2009
3. D. Barrington, C.-J. Lu, P. Miltersen, and S. Skyum. On monotone planar circuits. In *IEEE Conference on Computational Complexity*, pages 24–31, 1999.
4. D. M. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In *Proc. 18th STOC*, pages 1–5, 1986.
5. M. Ben-Or and R. Cleve. Computing algebraic formulas using a constant number of registers. In *Proc. 20th STOC*, pages 254–257, 1988.
6. H. L. Bodlaender. NC-algorithms for graphs with small treewidth. In *in Proc. 14th WG 1989, LNCS 344*, pages 1–10. 1989.
7. R. Brent. The parallel evaluation of general arithmetic expressions. *J. Assn. Comp. Mach.*, 21:201–206, 1974.
8. P. Bürgisser, M. Claussen, and M. Shokrollahi. *Algebraic Complexity Theory*. Springer Verlag, 1997.
9. L. M. Goldschlager. The monotone and planar circuit value problems are logspace comp lete for P. *SIGACT News*, 9(2):25–29, 1977.
10. M. Jansen and B. Rao. Simulation of arithmetical circuits by branching programs with preservation of constant width and syntactic multilinearity. In Proc. of the 4th International Comp. Science Symp. in Russia (CSR2009), LNCS Vol. 5675, pages 179–190. , 2009.
11. N. Jones. Space-bounded reducibility among combinatorial problems. *J. Comp. Sys. Sci.*, 11:68–85, 1975. Corrigendum *J. Comp. Sys. Sci.* 15:241, 1977.
12. N. Limaye, M. Mahajan, and J. Sarma. Upper bounds for monotone planar circuit value and variants. *Computational Complexity*, 18(3):377–412, 2009.
13. M. Mahajan. Polynomial size log depth circuits: between NC^1 and AC^1. Technical Report 91, BEATCS Computational Complexity Column, 2007.
14. M. Mahajan and B. Rao. Arithmetic circuits, syntactic multilinearity, and the limitations of skew formulae. In *Proc. MFCS*, LNCS 5162, pages 455–466, 2008.
15. M. Mahajan and B. Rao. Small-space analogues of Valiant's classes. In *Proc. 17th FCT*, LNCS Volume 5699, pages 455–466, 2009.
16. G. Malod and N. Portier. Characterizing valiant's algebraic complexity classes. *J. Complex.*, 24(1):16–38, 2008.
17. R. Raz. Separation of multilinear circuit and formula size. In *Proc. 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 344–351, 2004.
18. O. Reingold. Undirected st-connectivity in log-space. In *Proc. 37th Annual ACM Symposium on the Theory of Computing*, pages 376–385, 2005.
19. W. Savitch. Maze recognizing automata and nondeterministic tape complexity. *J. Comput. Syst. Sci.*, 7(4):389–403, 1973.
20. L. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12:641–644, 1983.