Electronic Colloquium on Computational Complexity, Report No. 122 (2014)

Dual VP Classes

Eric Allender
Department of Computer Science, Rutgers University
Piscataway, NJ, USA
allender@cs.rutgers.edu

Anna Gál
Department of Computer Science, University of Texas
Austin, TX, USA
panni@cs.utexas.edu

Ian Mertz Department of Computer Science, Rutgers University
Piscataway, NJ, USA
iwmertz@gmail.com

September 30, 2014

Abstract

We consider arithmetic complexity classes that are in some sense dual to the classes $\mathsf{VP}(\mathbb{F}_p)$ that were introduced by Valiant. This provides new characterizations of the complexity classes ACC^1 and TC^1, and also provides a compelling example of a class of high-degree polynomials that can be simulated via arithmetic circuits of much lower degree.

1 Introduction

Semiunbounded fan-in circuits play an important role in computational complexity theory. Over the Boolean semiring, logarithmic depth polynomial-size semiunbounded fan-in circuits (with bounded fan-in AND gates and unbounded fan-in OR gates, with NOT gates only at the input level) characterize the complexity class LogCFL, also known as SAC^1, which has been the subject of numerous investigations [Ven91, GLS01, LMSV01, AL13, Pet02, RA00]. Over $\mathbb{F}_p$, logarithmic depth polynomial-size semiunbounded fan-in circuits (with bounded fan-in multiplication gates and unbounded fan-in addition gates) characterize the complexity class $\mathsf{VP}(\mathbb{F}_p)$, the study of which was initiated by Valiant [Val79]; these classes have received quite a bit of study since then (e.g., [Bür99, Bür00, GW96, KP11]).

Because LogCFL is closed under complement [BCD$^+$89], it can be characterized in terms of semiunbounded fan-in circuits by restricting either the AND gates or the OR gates to have bounded fan-in. It is unknown if there is any other algebraic structure for which a similar phenomenon occurs. In particular, it is not known how the complexity of functions in $\mathsf{VP}(\mathbb{F}_p)$ compares to that of the functions in the classes defined by

ISSN 1433-8092

logarithmic depth polynomial-size semiunbounded fan-in circuits with bounded fan-in addition gates and unbounded fan-in multiplication gates.

Part of our motivation in this study, is to understand the computational power of these semiunbounded fan-in circuit classes, which are in some sense dual to Valiant's classes $\mathsf{VP}(\mathbb{F}_p)$. We use the notation $\Lambda\mathsf{P}(\mathbb{F}_p)$ to refer to the class of problems characterized by logarithmic depth polynomial-size semiunbounded fan-in circuits with bounded fan-in addition gates and unbounded fan-in multiplication gates. Formal definitions appear in Section 2. We show that each class $\Lambda\mathsf{P}(\mathbb{F}_p)$ corresponds exactly to a particular subclass of ACC^1, and that the union over all p of $\Lambda\mathsf{P}(\mathbb{F}_p)$ is exactly equal to ACC^1.

After characterizing the computational power of $\Lambda\mathsf{P}(\mathbb{F}_p)$, we next turn our attention to the complexity class TC^1, characterized by polynomial-size threshold circuits of logarithmic depth. Reif and Tate [RT92] gave an alternative characterization of TC^1 in terms of unbounded fan-in arithmetic circuits of logarithmic depth where the circuits for inputs of size n operate over the field $\mathbb{F}_{p_n}$, where p_n is the n-th prime. (See also the discussion of Reif and Tate's work in [BCK+14].) Using the standard notation (reviewed in Section 2), this characterization can be stated as $\mathsf{TC}^1 = \#\mathsf{AC}^1(\mathbb{F}_{p_n})$. We show that no computational power is lost by restricting the fan-in of the + gates in this setting: We show that $\mathsf{TC}^1 = \mathsf{L}^{\Lambda\mathsf{P}(\mathbb{F}_{p_n})}$.

Immerman and Landau [IL95] conjectured that every problem in TC^1 is reducible to the problem of computing the determinant of integer matrices. This would have several consequences, including providing a characterization of TC^1 in terms of $\mathsf{VP}(\mathbb{Q})$. Buhrman et al. [BCK+14] have argued that the Immerman-Landau conjecture is unlikely, in that this would imply that arbitrary polynomials having degree $n^{O(\log n)}$ and polynomial-size arithmetic circuits mod p_n could be simulated by arithmetic circuits of *much lower degree* over $\mathbb{Q}$. This raises the question: When can high-degree polynomials over one algebra be simulated by low-degree polynomials over another algebra?

In partial answer to this question, we show that all of the problems in $\Lambda\mathsf{P}(\mathbb{F}_2)$, $\Lambda\mathsf{P}(\mathbb{F}_3)$, $\Lambda\mathsf{P}(\mathbb{F}_5)$, $\Lambda\mathsf{P}(\mathbb{F}_{17})$ and in general any $\Lambda\mathsf{P}(\mathbb{F}_q)$ where $q-1$ is a power of 2 (that is, where q is a Fermat prime) can be simulated by polynomial size "weakly semiunbounded fan-in" arithmetic circuits over $\mathbb{F}_2$ of depth $\log n$. These are circuits where the addition gates have unbounded fan-in, and the multiplication gates have fan-in $O(\log n)$. Thus these circuits compute polynomials over $\mathbb{F}_2$ having algebraic degree at most $n^{O(\log\log n)}$. Further, we show that *all* functions over $\mathbb{F}_2$ with polynomial size unbounded fan-in log-depth circuits can be represented by polynomial size weakly semi-unbounded fan-in log-depth circuits. This offers the most compelling example of which we are aware, in which a natural class of polynomials of degree $n^{O(\log n)}$ over one algebra (such as $\mathbb{F}_2$) can be simulated by polynomials having much smaller degree.

We show that similar Boolean simulations hold for the subclasses of ACC^1 defined by unbounded fan-in AND, OR, and MOD_m gates, for any fixed m. Namely, for logarithmic depth polynomial-size circuits of unbounded fan-in AND, OR and MOD_m gates, it causes no loss of power to restrict the fan-in of the AND and OR gates to be logarithmic.

2 Preliminaries

We assume that the reader is familiar with Boolean circuit complexity classes such as AC^0 and ACC^0; a good source for this background material is the excellent text by Vollmer [Vol99]. The following standard notation is used by Vollmer for circuit complexity classes, and we follow those conventions here:

Definition 1.

- AC^i *is the class of languages accepted by Dlogtime-uniform circuit families of polynomial size and depth* $O(\log^i n)$*, consisting of unbounded fan-in* AND*, and* OR *gates, along with* NOT *gates.*
- $\mathsf{AC}^i[m]$ *is defined as* AC^i*, but in addition unbounded fan-in* MOD_m *gates are allowed, which output 1 iff the number of input wires carrying a value of 1 is a multiple of* m*.*
- *For any finite set* $S \subset \mathbb{N}$*,* $\mathsf{AC}^i[S]$ *is defined analogously to* $\mathsf{AC}^i[m]$*, but now the circuit families are allowed to use* MOD_r *gates for any* $r \in S$*. It is known that, for any* $m \in \mathbb{N}, \mathsf{AC}^i(m) = \mathsf{AC}^i(S(m))$*, where* $S(m) = \{p : p$ *is prime and* p *divides* $m\}$ *[Smo87]. Thus, in particular* $\mathsf{AC}^i[6] = \mathsf{AC}^i[2,3]$ *and* $\mathsf{AC}^i = \mathsf{AC}^i[\emptyset]$*. (When it will not cause confusion, we omit unnecessary brackets, writing for instance* $\mathsf{AC}^i[2,3]$ *instead of* $\mathsf{AC}^i[\{2,3\}]$*.)*
- $\mathsf{ACC}^i = \bigcup_m \mathsf{AC}^i[m]$.
- TC^i *is the class of languages accepted by Dlogtime-uniform circuit families of polynomial size and depth* $O(\log^i n)$*, consisting of unbounded fan-in* MAJORITY *gates, along with* NOT *gates.*
- SAC^i *is the class of languages accepted by Dlogtime-uniform circuit families of polynomial size and depth* $O(\log^i n)$*, consisting of unbounded fan-in* OR *gates and bounded fan-in* AND *gates, along with* NOT *gates at the leaves.*

Note that the restriction that NOT gates appear at the leaves in SAC^i circuits is essential; if NOT gates were allowed to appear everywhere, then these classes would coincide with AC^i. Similarly, note that we do not bother to define a complexity class $\mathsf{SAC}^i[m]$, since a MOD_m gate with a single input is equivalent to a NOT gate, and thus $\mathsf{SAC}^i[m]$ would be the same as $\mathsf{AC}^i[m]$.

The algebraic complexity classes $\mathsf{VP}(R)$ for various algebraic structures R were originally defined [Val79] in the context of nonuniform circuit complexity, as classes of families of n-variate polynomials of degree $n^{O(1)}$ that can be represented by polynomial-size algebraic circuits over R. In this paper, we focus on uniform circuit families, and thus we use the notation $\mathsf{VP}(R)$ to denote the families of polynomials that result when we impose a logspace-uniformity condition on the circuit families. In the original nonuniform setting, it was shown by [VSBR83] that the circuits defining polynomials in $\mathsf{VP}(R)$ can be assumed to have small depth. Later [AJMV98] a slightly improved characterization was provided, that works also in the context of uniform circuit complexity:

Theorem 1. *[VSBR83, AJMV98] For any commutative semiring* R*,* $\mathsf{VP}(R)$ *coincides with the class of families of polynomials over* R *represented by logspace-uniform circuit families of polynomial size and logarithmic depth with unbounded fan-in* $+$ *gates, and fan-in two* $\times$ *gates.*

Note that over $\mathbb{F}_p$, many different polynomials yield the same function. For example, since $x^3 = x$ in $\mathbb{F}_3$, every function on n variables has a polynomial of degree at most $2n$. Very likely there are functions represented by polynomials in $\mathsf{VP}(\mathbb{F}_3)$ of degree, say, n^5, but not by any polynomial of degree $2n$. On the other hand, there is a case to be made for focusing on the *functions* in these classes, rather than focusing on the *polynomials* that represent those functions. For instance, if the Immerman-Landau conjecture is true, and TC^1 is reducible to problems in $\mathsf{VP}(\mathbb{F}_{p_n})$ (for instance), it would suffice for every *function* in $\mathsf{TC}^1 = \#\mathsf{AC}^1(\mathbb{F}_{p_n})$ to have a representation in $\mathsf{VP}(\mathbb{F}_{p_n})$, even though the *polynomials* represented by $\#\mathsf{AC}^1(\mathbb{F}_{p_n})$ circuits have large degree, and thus cannot be in any VP class.

In the literature on VP classes, one standard way to focus on the *functions* represented by polynomials in VP is to consider what is called the *Boolean Part* of $\mathsf{VP}(R)$, which is the set of *languages* $A \subseteq \{0,1\}^*$ such that, for some sequence of polynomials (Q_n), for $x \in A$ we have $Q_{|x|}(x) = 1$, and for $x \in \{0,1\}^*$ such that $x \notin A$ we have $Q_{|x|}(x) = 0$.

When the algebra R is a finite field, considering the Boolean part of $\mathsf{VP}(R)$ captures the relevant complexity aspects, since the computation of any function represented by a polynomial in $\mathsf{VP}(R)$ (with inputs and outputs coming from R) is logspace-Turing reducible to some language in the Boolean Part of $\mathsf{VP}(R)$.

In this paper, we will be concerned exclusively with the "Boolean Part" of various arithmetic classes. For notational convenience, we will just refer to these classes using the "VP" notation, rather than constantly repeating the phrase "Boolean Part".

Following the standard naming conventions of [Vol99], for any Boolean circuit complexity class $\mathcal{C}$ defined in terms of circuits with AND and OR gates, we define the class $\#\mathcal{C}(\mathcal{R})$ to be the class of functions represented by algebraic circuits defined over the algebra R, where AND is replaced by $\times$, and OR is replaced by $+$ (and NOT gates are interpreted using the appropriate notion of negation in the algebra R). In particular, we will be concerned with the following two classes:

Definition 2. *Let R be any semiring. Then*

- *$\#\mathsf{AC}^1(R)$ is the class of functions $f\{0,1\}^* \to R$ represented by families of logspace-uniform circuits of unbounded fan-in $+$ and $\times$ gates having depth $O(\log n)$ and polynomial size.*
- *$\#\mathsf{SAC}^1(R)$ is the class of functions $f\{0,1\}^* \to R$ represented by families of logspace-uniform circuits of unbounded fan-in $+$ gates and $\times$ gates of fan-in two, having depth $O(\log n)$ and polynomial size.*

Input variables may be negated, using the appropriate notion of negation in the algebra R. Where no confusion will result, the notation $\#\mathcal{C}(\mathcal{R})$ will also be used to refer to the class of languages whose characteristic functions lie in the given class.

Hence from Theorem 1 we obtain:

Proposition 1. *Let p be a prime. Then $\mathsf{VP}(\mathbb{F}_p) = \#\mathsf{SAC}^1(\mathbb{F}_p)$.*

2.1 New Definitions: Λ-classes

In this section, we introduce and define classes that are dual to the $\#\mathsf{SAC}^1(R)$ classes discussed above. Define $\#\mathsf{SAC}^{1,*}(R)$ to be the class of functions $f\{0,1\}^* \to R$ represented by families of logspace-uniform circuits of unbounded fan-in $\times$ gates and $+$ gates of fan-in two, having depth $O(\log n)$ and polynomial size. Proposition 1 highlights the connection between VP and $\#\mathsf{SAC}^1$; thus we will utilize the convenient notation $\Lambda\mathsf{P}(R)$ to denote the dual notation, rather than the more cumbersome $\#\mathsf{SAC}^{1,*}(R)$.

Of course, the set of formal polynomials represented by $\Lambda\mathsf{P}$ circuits is not contained in any VP class, because $\Lambda\mathsf{P}$ contains polynomials of degree $n^{O(\log n)}$. However, as discussed in the previous section, we are considering the "Boolean Part" of these classes. More formally:

Definition 3. *Let p be a prime number. $\Lambda\mathsf{P}(\mathbb{F}_p)$ is the class of all languages $A \subseteq \{0,1\}^*$ with the property that there is a logspace-uniform family of circuits $\{C_n : n \in \mathbb{N}\}$ such that*

- *The depth of C_n is $O(\log n)$.*
- *Each C_n consists of input gates, $+$ gates, and $\times$ gates.*
- *Each $+$ gate has fan-in two, whereas there is no bound on the fan-in of the $\times$ gates.*
- *For each string x of length n, x is in A if and only if $C_n(x)$ evaluates to 1, when the $+$ and $\times$ gates are evaluated over $\mathbb{F}_p$. Furthermore, if $x \notin A$, then $C_n(x)$ evaluates to 0.*

Observe that $\Lambda\mathsf{P}(\mathbb{F}_p)$ is closed under logspace-Turing reductions; that is $\Lambda\mathsf{P}(\mathbb{F}_p) = \mathsf{L}^{\Lambda\mathsf{P}(\mathbb{F}_p)}$. Perhaps the easiest way to see this is to note that there are only polynomially-many queries that a logspace-Turing reduction can pose, on a given input x, since the query that is posed is determined entirely by the worktape configuration of the oracle Turing machine when it begins to write the query. These queries can be denoted $y_1, \ldots, y_{n^k}$ for some k. If $A \in \mathsf{L}^{\Lambda\mathsf{P}(\mathbb{F}_p)}$, then there is a language $B \in \mathsf{L}$ such that $x \in A$ iff $(x,z) \in B$ where z is the bit string of length n^k recording the oracle answers for each query y_i. The language B has a SAC^1 circuit with the property that, on each input (x,z), it has either zero or one accepting subtree. Although the argument showing that SAC^1 is closed under complement does not maintain a meaningful relationship regarding the number of accepting subtrees, it follows easily from the fact that L is closed under complement that B also has $\Lambda\mathsf{P}(\mathbb{F}_p)$ circuits. By connecting $\Lambda\mathsf{P}(\mathbb{F}_p)$ circuits computing the answer to each oracle query y_i to the input variables for z, one obtains a $\Lambda\mathsf{P}(\mathbb{F}_p)$ circuit for A. Thus we believe that we have ample justification for defining $\Lambda\mathsf{P}(\mathbb{F}_p)$ as a class of Boolean languages, as we have done.

We mention that $\Lambda\mathsf{P}$ classes over different fields of the same characteristic define the same class of languages.

Proposition 2. *Let p be a prime. Then $\Lambda\mathsf{P}(\mathbb{F}_p) = \Lambda\mathsf{P}(\mathbb{F}_{p^k})$.*

Proof. The finite field of size p^k is a vector space of dimension k over the field of size p, and thus can be represented by $k \times k$ matrices over $\mathbb{F}_p$. Thus each + and $\times$ gate of a $\Lambda\mathsf{P}(\mathbb{F}_{p^k})$ circuit can be replaced by subcircuits implementing matrix sum and product over $\mathbb{F}_p$. The resulting circuit is a $\Lambda\mathsf{P}(\mathbb{F}_p)$ circuit. □

It is also appropriate to use the VP and $\Lambda\mathsf{P}$ notation when referring to the classes defined by Boolean semiunbounded fan-in circuits with negation gates allowed at the inputs. With this notation, $\mathsf{VP}(B_2)$ corresponds to the Boolean class SAC^1, and $\Lambda\mathsf{P}(B_2)$ corresponds to the complement of SAC^1 (with bounded fan-in OR gates, unbounded fan-in AND gates and negation gates allowed at the inputs). It has been shown by [BCD$^+$89] that SAC^1 is closed under complement. Thus we close this section with the equality that serves as a springboard for investigating the $\Lambda\mathsf{P}$ classes.

Theorem 2. *[BCD$^+$89]* $\mathsf{VP}(B_2) = \Lambda\mathsf{P}(B_2)(= \mathsf{SAC}^1 = \mathsf{LogCFL})$.

We do not believe that $\mathsf{VP}(\mathbb{F}_p) = \Lambda\mathsf{P}(\mathbb{F}_p)$ for any prime p, but in Section 4 we discuss some other settings where it is plausible that VP and $\Lambda\mathsf{P}$ classes might have equivalent complexity.

3 Subclasses of ACC^1

In this section, we present our characterizations of ACC^1 in terms of the $\Lambda\mathsf{P}(\mathbb{F}_p)$ classes.

Theorem 3. *For any prime p, $\Lambda\mathsf{P}(\mathbb{F}_p) = \mathsf{AC}^1[S(p-1)]$. (Recall that $S(m)$ is defined in Definition 1.)*

Proof. ($\subseteq$): Consider a $\Lambda\mathsf{P}(\mathbb{F}_p)$ circuit C. We will create a circuit C' with gates that record the binary representation of the value at each gate g in C, using unbounded fan-in AND, OR and MOD_q gates, for each $q \in S(p-1)$. Since the input gates of C take on only binary values (by our definition of $\Lambda\mathsf{P}(\mathbb{F}_p)$), the input levels of C' and C coincide.

Each + gate g of C (of fan-in 2) can be computed with NC^0 circuitry using the Boolean gates of C', since the binary representation of both inputs to g consist of only $\log p = O(1)$ bits.

Now consider a $\times$ gate g of C, having unbounded fan-in: $g = \prod_i h_i$. For each $a \in \mathbb{F}_p$, C' will have a subcircuit computing the Boolean value $[g = a]$. (We will use the notation "$[B]$" to refer to the truth-value of B.) The i-th bit of the binary representation of g is thus obtained by taking the OR over all a of ($[g = a]$ AND (the i-th bit of a)).

The value $[g = 0]$ is obtained by simply checking if there is some i such that $h_i = 0$.

Now we show how to compute $[g = a]$ for $a \neq 0$. Let $p - 1 = \prod_{j=1}^{\ell} q_j^{e_j}$ where $S(p-1) = \{q_1, \ldots, q_\ell\}$. Let σ be a generator of the multiplicative group of $\mathbb{F}_p$. Then $g = \prod_i h_i = \prod_i \sigma^{\log h_i} = \sigma^{\sum_i \log h_i}$ where "$\log b$" denotes the unique element of $\mathbb{F}_p$ such that $\sigma^{\log b} = b$. Hence the value $[g = a]$ is equivalent to $[\log a \equiv \sum_i \log h_i \bmod (p-1)]$, which in turn is equivalent to the AND of the values $[\log a \equiv \sum_i \log h_i \bmod (q_j^{e_j})]$.

If $e_j = 1$ then the value $[\log a \equiv \sum_i \log h_i \bmod (q_j)]$ is easy to compute with a MOD_{q_j} gate, as follows. Map the binary representation of each h_i to the string $x_i = 1^{\log h_i} 0^{p - \log h_i}$. (Note that the mapping from h_i to x_i is computable in logspace-uniform NC^0.) Let X_a be the string that results from concatenating the string $1^{(p-1)-\log a}$ and

all of the strings x_i. Now observe that feeding X_a into a MOD_{q_j} gate computes the value $[\log a \equiv \sum_i \log h_i \bmod (q_j)]$.

If $e_j > 1$, then first observe that $[b \equiv 0 \bmod q_j^{e_j}]$ can be computed by checking if each of $b, \binom{b}{q_j}, \binom{b}{q_j^2}, \ldots \binom{b}{q_j^{e_j-1}}$ is equivalent to $0 \bmod q_j$. (See, e.g. [BT94, Fact 2.2].) Observe also that $\binom{b}{d}$ can be represented as the number of different AND gates of fan-in d that evaluate to 1, taking inputs from the string X_a. Thus all of these conditions can be checked in constant depth with unbounded fan-in AND gates and MOD_{q_j} gates.

Since C has depth $O(\log n)$, and C' consists of layers of constant-depth circuitry to replace each layer of gates in C, this completes the proof of this direction.

($\supseteq$): Given an $\mathsf{AC}^1[S(p-1)]$ circuit C, we show how to construct an algebraic circuit C' that is equivalent to C. Each gate g of C will have an equivalent gate g in C'. The input gates of C and of C' are exactly the same.

If g is a NOT gate in C, say $g = \neg h$, then in C' we will have $g = (h + (p-1)) \times (h + (p-1))$.

If g is an AND gate (say, $g = \wedge_i h_i$), then in C' we will have $g = \prod_i h_i$. OR gates will be handled the same way, using De Morgan's Laws.

Now consider the case when g is a MOD_{q_j} gate with inputs h_i. Thus g computes the value $[\sum_i h_i \equiv 0 \bmod q_j]$. Let σ be a generator of the multiplicative cyclic subgroup of size q_j. First map each h_i to the value $h'_i = 1 + ((\sigma + (p-1)) \times h_i$, and observe that $h'_i = \sigma^{h_i}$ for all $h_i \in \{0, 1\}$. Observe that $1 - \prod_i h'_i = 1 - \sigma^{\sum_i h_i}$ is equal to 0 if $\sum_i h_i$ is a multiple of q_j, and is non-zero otherwise. Thus $1 - (1 - \prod_i h'_i)^{p-1}$ is equal to the Boolean value $[\sum_i h_i \equiv 0 \bmod q_j]$.

It is easy to verify that C' has logarithmic depth, and uses only bounded fan-in $+$ gates, as well as unbounded fan-in $\times$ gates. □

Corollary 1. $\mathsf{ACC}^1 = \bigcup_p \Lambda\mathsf{P}(\mathbb{F}_p)$.

Proof. Let $A \in \mathsf{ACC}^1$. Thus $A \in \mathsf{AC}^1[m]$ for some modulus m.

By Dirichlet's Theorem, the arithmetic progression $m+1, 2m+1, \ldots$ contains some prime p. Thus $\mathsf{AC}^1[m] \subseteq \mathsf{AC}^1[S(p-1)] = \Lambda\mathsf{P}(\mathbb{F}_p)$. □

Note also that several of the $\Lambda\mathsf{P}(\mathbb{F}_p)$ classes coincide. This is neither known nor believed to happen with the $\mathsf{VP}(\mathbb{F}_p)$ classes.

Corollary 2.

- $\Lambda\mathsf{P}(\mathbb{F}_2) = \mathsf{AC}^1$.
- *If p is a Fermat prime (that is, $p-1$ is a power of 2, such as $p \in \{3, 5, 17, 257, 65{,}537\}$), then* $\Lambda\mathsf{P}(\mathbb{F}_p) = \mathsf{AC}^1[2]$.
- $\Lambda\mathsf{P}(\mathbb{F}_7) = \Lambda\mathsf{P}(\mathbb{F}_{13}) = \Lambda\mathsf{P}(\mathbb{F}_{19})$.
- *More generally, $S(p-1) = S(q-1)$ implies* $\Lambda\mathsf{P}(\mathbb{F}_p) = \Lambda\mathsf{P}(\mathbb{F}_q)$.

Augmenting the $\Lambda\mathsf{P}(\mathbb{F}_p)$ classes with unbounded fan-in addition gates increases their computation power only by adding MOD_p gates, as the following theorem demonstrates.

Theorem 4. *For each prime p, $\#\mathsf{AC}^1(\mathbb{F}_p) = \mathsf{AC}^1[\{p\} \cup S(p-1)]$.*

Proof. ($\subseteq$): Again, we use a gate-by-gate simulation, and record the binary value of each gate of g. Multiplication gates are handled as in the proof of Theorem 3. Consider now the case of an addition gate $g = \sum_i h_i$. First, we show how to compute the Boolean value $[g = a]$. Using this information to obtain the binary representation of the value of g is then handled the same way that this was done for multiplication gates in the proof of Theorem 3.

Using NC^0 circuitry, one can convert the binary representation of each h_i to the unary string $y_i = 1^{h_i}0^{p-h_i}$ (as in the proof of Theorem 3). Let Y_a be the string 1^{p-a} concatenated with all of the strings y_i. Feeding Y_a into a MOD_p gate computes the Boolean value $[g = a]$.

($\supseteq$): As in Theorem 3, we carry out a gate-by-gate simulation, whereby each gate g in a $\mathsf{AC}^1[\{p\} \cup S(p-1)]$ circuit C is equivalent to a gate (also called g) in a $\#\mathsf{AC}^1(\mathbb{F}_p)$ circuit C'. We only need to consider the case where g is a MOD_p gate with Boolean inputs h_i. In this case, note that $g = 1 + ((\sum_i h_i)^{p-1} \times (p-1))$. □

Corollary 3. $\mathsf{ACC}^1 = \bigcup_p \Lambda\mathsf{P}(\mathbb{F}_p) = \bigcup_p \#\mathsf{AC}^1(\mathbb{F}_p)$.

Corollary 4. *For any prime p there is a prime q such that $\#\mathsf{AC}^1(\mathbb{F}_p) \subseteq \Lambda\mathsf{P}(\mathbb{F}_q)$.*

Proof. By Dirichlet's Theorem, there is a prime q such that $q-1$ is a multiple of $p(p-1)$. The claim now follows immediately from Theorems 4 and 3. □

It will be useful to bear in mind that $\mathsf{VP}(\mathbb{F}_p)$ also has a simple characterization in terms of Boolean circuits. In order to present this characterization, we present a more general definition, which will be needed later.

Definition 4. *Let $m \in \mathbb{N}$, and let g be any function on $\mathbb{N}$. Define g-$\mathsf{AC}^1[m]$ to be the class of languages with logspace-uniform circuits of polynomial size and depth $O(\log n)$, consisting of unbounded-fan-in MOD_m gates, along with* AND *gates of fan-in $O(g(n))$. Clearly g-$\mathsf{AC}^1[m] \subseteq \mathsf{AC}^1[m]$.*

Observe that, since a MOD_m gate can simulate a NOT gate, g-$\mathsf{AC}^1[m]$ remains the same if OR gates of fan-in $O(g)$ are also allowed.

Corollary 5. *For every prime p, $\mathsf{VP}(\mathbb{F}_p) = 2\text{-}\mathsf{AC}^1[p] \subseteq \mathsf{AC}^1[p]$.*

Proof. Recall that $\mathsf{VP}(\mathbb{F}_p) = \#\mathsf{SAC}^1(\mathbb{F}_p)$. Thus we need only show how to simulate bounded fan-in $\times$ gates and unbounded fan-in $+$ gates. Bounded fan-in $\times$ gates can be simulated in $O(1)$ depth using AND and OR gates of fan-in two (since the values being multiplied are of size $O(1)$). Unbounded fan-in $+$ gates can be simulated using MOD_p gates, as in the proof of the preceding theorem.

For the converse inclusion, consider a $2\text{-}\mathsf{AC}^1[p]$ circuit. Since a unary MOD_p gate is equivalent to a NOT gate, we can assume that the circuit has only fan-in two AND gates and unbounded fan-in MOD_p gates. Thus each Boolean AND gate can be simulated by a fan-in two multiplication gate, and the MOD_p gates can be simulated as in the proof of Theorem 4. □

We remark that the same proof shows that, for any $m \in \mathbb{N}$, $\mathsf{VP}(\mathbb{Z}_m) \subseteq 2\text{-}\mathsf{AC}^1[m]$. However, the converse inclusion is not known, unless m is prime.

We remark also that the proofs of Theorems 3 and 4 carry over also for depths other than $\log n$. (Related results for constant-depth unbounded-fan-in circuits can be found already in [Smo87, AAD00].)

Corollary 6. *For any prime p, $\#\mathsf{SAC}^{i,*}(\mathbb{F}_p) = \mathsf{AC}^i[S(p-1)]$ and $\#\mathsf{AC}^i(\mathbb{F}_p) = \mathsf{AC}^i[p \cup S(p-1)]$.*

3.1 Comparing $\Lambda\mathsf{P}$ and VP.

How do the $\Lambda\mathsf{P}$ and VP classes compare to each other?

As a consequence of Corollary 5 and Theorem 3, $\mathsf{VP}(\mathbb{F}_p) \subseteq \Lambda\mathsf{P}(\mathbb{F}_q)$ whenever p divides $q-1$. In particular, $\mathsf{VP}(\mathbb{F}_2) \subseteq \Lambda\mathsf{P}(\mathbb{F}_q)$ for any prime $q > 2$. No inclusion of any $\Lambda\mathsf{P}$ class in any VP class is known unconditionally, although $\Lambda\mathsf{P}(B_2)(= \mathsf{SAC}^1)$ is contained in every $\mathsf{VP}(\mathbb{F}_p)$ class in the nonuniform setting [GW96, RA00], and this holds also in the uniform setting under a plausible derandomization hypothesis [ARZ99].

No $\Lambda\mathsf{P}(\mathbb{F}_q)$ class can be contained in $\mathsf{VP}(\mathbb{F}_p)$ unless $\mathsf{AC}^1 \subseteq \mathsf{VP}(\mathbb{F}_p)$, since $\mathsf{AC}^1 = \Lambda\mathsf{P}(\mathbb{F}_2) \subseteq \Lambda\mathsf{P}(\mathbb{F}_3) \subseteq \Lambda\mathsf{P}(\mathbb{F}_q)$ for every prime $q \geq 3$. AC^1 is not known to be contained in any VP class, although we return to this topic again in Section 4

4 Threshold circuits and small degree

The inspiration for the results in this section comes from the following theorem of Reif and Tate [RT92] (as re-stated by Buhrman et al. [BCK+14]):

Theorem 5. $\mathsf{TC}^1 = \#\mathsf{AC}^1(\mathbb{F}_{p_n})$.

Here, the class $\#\mathsf{AC}^1(\mathbb{F}_{p_n})$ consists of the languages whose (Boolean) characteristic functions are computed by logspace-uniform families of arithmetic circuits of logarithmic depth with unbounded fan-in $+$ and $\times$ gates, where the arithmetic operations of the circuit C_n are interpreted over $\mathbb{F}_{p_n}$, where $p_1, p_2, p_3, \ldots$ is the sequence of all primes $2, 3, 5, \ldots$ That is, circuits for inputs of length n use the n-th prime to define the algebraic structure.

This class is closed under logspace-Turing reductions – but when we consider *other* circuit complexity classes defined using $\mathbb{F}_{p_n}$, it is *not* clear that these other classes are closed under logspace-Turing reductions.

As an important example, we mention $\mathsf{VP}(\mathbb{F}_{p_n})$. As we show below, this class has an important connection to $\mathsf{VP}(\mathbb{Q})$, which is perhaps the canonical example of a VP class. Vinay [Vin91] proved that $\mathsf{VP}(\mathbb{Q})$ has essentially the same computational power as $\#\mathsf{LogCFL}$ (which counts among its complete problems the problem of determining how many distinct parse trees a string x has in a certain context-free language). Here, we mention one more alternative characterization of the computational power of $\mathsf{VP}(\mathbb{Q})$.

Proposition 3. $\mathsf{L}^{\mathsf{VP}(\mathbb{F}_{p_n})} = \mathsf{L}^{\mathsf{VP}(\mathbb{Q})} = \mathsf{L}^{\#\mathsf{LogCFL}}$.

Proof. Consider the first equality. If one wants to compute the value of a $\mathsf{VP}(\mathbb{F}_{p_n})$ circuit on a given input of length n, in logspace one can first compute the value of p_n. Then

one can use a $\mathsf{VP}(\mathbb{Q})$ oracle to evaluate the $\mathsf{VP}(\mathbb{F}_{p_n})$ circuit over the rationals instead of over $\mathbb{F}_{p_n}$, obtaining an integer result. Then one can divide the result by p_n and obtain the remainder, which is the value of the circuit in $\mathbb{F}_{p_n}$, using the fact that division is computable in logspace [CDL01, HAB02].

Conversely, if one wants to evaluate a $\mathsf{VP}(\mathbb{Q})$ circuit on a given n-tuple of rationals, one can use the standard technique of computing the numerator and denominator separately; the circuits for these functions are also in $\mathsf{VP}(\mathbb{Q})$. Thus our task boils down to evaluating an integer-valued arithmetic circuit C_n. To do this, we use Chinese remaindering, and evaluate circuits (with some dummy variables) over the primes $p_n, p_{n+1}, \ldots, p_{n+n^c}$ for some constant c. Converting between Chinese remainder representation and binary representation can be accomplished in logspace [CDL01, HAB02], which completes the proof of the first equality.

For the second equality, we similarly use the fact that $\mathsf{VP}(\mathbb{Q})$ circuits with integer coefficients and inputs can be evaluated in $\#\mathsf{LogCFL}$, and appeal to [Vin91]. □

When we consider arithmetic circuits of superpolynomial algebraic degree (such as the $\Lambda\mathsf{P}$ classes), evaluating the circuits over the integers can produce outputs that require a superpolynomial number of bits to express in binary. Thus, when we consider such classes, it will always be in the context of structures (such as $\mathbb{F}_{p_n}$) where the output can always be represented in a polynomial number of bits.

Our first new result in this section, is to improve Theorem 5.

Theorem 6. $\mathsf{TC}^1 = \#\mathsf{AC}^1(\mathbb{F}_{p_n}) = \mathsf{L}^{\Lambda\mathsf{P}(\mathbb{F}_{p_n})}$.

Proof. We will show how to simulate a $\#\mathsf{AC}^1(\mathbb{F}_{p_n})$ circuit C, by making calls to an appropriate function in $\Lambda\mathsf{P}(\mathbb{F}_{p_n})$. The first step is to find a prime q that is not too much larger than p_n, such that $q-1$ is a multiple of $p_n(p_n-1)$. Xylouris [Xyl11] has shown that the sequence $1+p_n(p_n-1), 1+2p_n(p_n-1), 1+3p_n(p_n-1)\ldots$ contains a prime of size $O(p_n^{10.4})$. Thus our logspace procedure will begin by enumerating the elements of this sequence, and is guaranteed to find some such prime q. We will create an arithmetic circuit C' operating over $\mathbb{F}_q$ that will allow us to simulate C.

For each gate g of C and each $a \in \mathbb{F}_{p_n}$, C' will have a gate computing the Boolean value $[g = a]$. If g is an input gate, our logspace procedure will compute the value of each $[g = a]$ and provide these as the inputs to the circuit C'.

Let us now consider the case when g is a + gate, $g = \sum_i h_i$. Let γ be a generator of the cyclic subgroup of the multiplicative group of $\mathbb{F}_q$ of order p_n. Our circuit C' will have gates $h_{i,a}$ computing the value

$$h_{i,a} = ([h_i = a] \times (\gamma^a - 1) + 1).$$

Observe that $\prod_a h_{i,a}$ is equal to γ^{h_i}. C' will have a gate g' computing the value $g' = \prod_{i,a} h_{i,a}$. Note that g' is equal to $\gamma^{\sum_i h_i} = \gamma^g$ (since γ has order p_n). The value of the gate $[g = b]$ (for a given $b \in \mathbb{F}_{p_n}$) is thus $c_b^{-1} \times \prod_{\ell \neq b}(\gamma^\ell - g')$, where the constant $c_b = \prod_{\ell \neq b}(\gamma^\ell - b)$ can be computed in logspace and is thus available as a constant in C'.

It remains only to deal with the case when g is a $\times$ gate, $g = \prod_i h_i$. In C', the gate $[g = 0]$ is $1 - \prod_i (1 - [h_i = 0])$.

Let μ be a generator of the multiplicative group of $\mathbb{F}_{p_n}$, and let α be a generator of the subgroup of the multiplicative group of $\mathbb{F}_g$ of order $p_n - 1$. If g does not evaluate

to 0, then g is equal to μ^b for some b. Our circuit C' will have gates $h_{i,\ell}$ computing the values

$$h_{i,\ell} = ([h_i = \mu^\ell] \times (\alpha^\ell - 1) + 1).$$

Our circuit C' will have gates h'_i computing the value $h'_i = \prod_\ell h_{i,\ell}$. Observe that h'_i is equal to α^a if $h_i = \mu^a$, and h'_i is equal to 1 if $h_i = 0$.

In C', there will be a gate g' that computes the following value: $g' = (1 - [g = 0]) \prod_i h'_i = ([g \neq 0]) \prod_i \alpha^{\log_\mu h_i} = ([g \neq 0]) \alpha^{\sum_i \log_\mu h_i} = ([g \neq 0]) \alpha^{\log_\mu g}$. Observe that, if $g \neq 0$, then $g = \mu^b$ for some b, and in this case g' evaluates to α^b. The value of the gate $[g = \mu^b]$ (for a given $b \in \mathbb{F}_{p_n}$) is thus $c_b^{-1} \times \prod_{\ell \neq b} (\alpha^\ell - g')$, where the constant $c_b = \prod_{\ell \neq b} (\alpha^\ell - \mu^b)$ can be computed in logspace and is thus available as a constant in C'. □

For completeness, we add one more relevant characterization of TC^1:

Theorem 7. $\mathsf{TC}^1 = \#\mathsf{AC}^1(\mathbb{F}_{p_n}) = \mathsf{L}^{\Lambda\mathsf{P}(\mathbb{F}_{p_n})} = \mathsf{AC}^1[p_n]$

Proof. We need only consider the last equality.

($\supseteq$): MAJORITY gates can simulate AND, OR, and MOD_{p_n} gates constant depth; thus this direction is easy.

($\subseteq$): Let ϵ be chosen so that $2n^\epsilon < p_n$ for every n. Any MAJORITY gate (of fan-in n^k) can be simulated by an AC^0-reduction to MAJORITY gates having fan-in n^ϵ [AK10]. Thus if $A \in \mathsf{TC}^1$, then A is accepted by a family of circuits if AND, OR, and MAJORITY gates, where the MAJORITY gates have fan-in at most n^ϵ. It suffices to show how to simulate a MAJORITY gate with inputs $h_1, \ldots, h_\ell$. Note that $\mathrm{MOD}_{p_n}(h_1, \ldots, h_\ell, 1^{p_n - b})$ computes the value $[b = \sum_i h_i]$. Thus the MAJORITY of the h_i is simply the OR, over all $b > \ell/2$ of the subcircuits computing $[b = \sum_i h_i]$. □

In order to set the context for the results of the next section, it is necessary to consider an extension of Theorem 6, involving arithmetic circuits over certain *rings*. Thus we require the following definition.

Definition 5. *Let (m_n) be any sequence of natural numbers (where each $m_n > 1$) such that the mapping $1^n \mapsto m_n$ is computable in logspace. We use the notation $\#\mathsf{AC}^1(\mathbb{Z}_{m_n})$ to denote the class of functions f with domain $\{0,1\}^*$ such that there is a logspace-uniform family of arithmetic circuits $\{C_n\}$ of logarithmic depth with unbounded fan-in $+$ and $\times$ gates, where the arithmetic operations of the circuit C_n are interpreted over $\mathbb{Z}_{m_n}$, and for any input x of length n, $f(x) = C_n(x)$. We use the notation $\#\mathsf{AC}^1(\mathbb{Z}_{\mathsf{L}})$ to denote the union, over all logspace-computable sequence of moduli (m_n), of $\#\mathsf{AC}^1(\mathbb{Z}_{m_n})$.*

Since the sequence of primes (p_n) is logspace-computable, $\mathsf{TC}^1 (= \#\mathsf{AC}^1(\mathbb{F}_{p_n}))$ is clearly contained in $\#\mathsf{AC}^1(\mathbb{Z}_{\mathsf{L}})$. Conversely, all of the functions in $\#\mathsf{AC}^1(\mathbb{Z}_{\mathsf{L}})$ are computable in TC^1. To see this, consider a function $f \in \#\mathsf{AC}^1(\mathbb{Z}_{\mathsf{L}})$. To evaluate $f(x)$ for an input of length n, first we compute the modulus m_n and the circuit C_n. To evaluate each gate g of C_n (in binary), first we compute the sum or product of the values that feed into g (which can be done in constant depth using threshold circuits) and then we reduce the result modulo m_n (which involves division, which can also be computed in constant depth). Thus, arithmetic circuits over the integers mod m_n for reasonable sequences of moduli m_n give yet another arithmetic characterization of TC^1.

4.1 Degree Reduction

In this subsection, we introduce a class of circuits that is intermediate between the unbounded fan-in circuit model and the semiunbounded fan-in model, for the purposes of investigating when arithmetic circuits of superpolynomial algebraic degree can be simulated by arithmetic circuits (possibly over a different algebra) with much smaller algebraic degree.

The starting point for this subsection is Theorem 4.3 in [AJMV98], which states that every problem in AC^1 is reducible to a function computable by polynomial-size arithmetic circuits of degree $n^{O(\log\log n)}$. In this section, we refine the result of [AJMV98], and put it in context with the theorems about TC^1 that were presented in the previous subsection. Those results show that TC^1 reduces to semiunbounded fan-in arithmetic circuits in the $\Lambda\mathsf{P}(\mathbb{F}_{p_n})$ model, but leave open the question of whether TC^1 also reduces to semiunbounded fan-in arithmetic circuits in the $\mathsf{VP}(\mathbb{F}_{p_n})$ model (which coincides with $\mathsf{VP}(\mathbb{Q})$). We are unable to answer this question, but we do show that some interesting inclusions can be demonstrated if we relax the VP model, by imposing a less-stringent restriction on the fan-in of the $\times$ gates.

Definition 6. *Let (m_n) be any sequence of natural numbers (where each $m_n > 1$) such that the mapping $1^n \mapsto m_n$ is computable in logspace. $\#\mathsf{WSAC}^1(\mathbb{Z}_{m_n})$ is the class of functions represented by logspace-uniform arithmetic circuit families $\{C_n\}$, where C_n is interpreted over $\mathbb{Z}_{m_n}$, where each C_n has size polynomial in n, and depth $O(\log n)$, and where the $+$ gates have unbounded fan-in, and the $\times$ gates have fan-in $O(\log n)$. Thus these circuits are not semiunbounded, but have a "weak" form of the semiunbounded fan-in restriction. We use the notation $\#\mathsf{WSAC}^1(\mathbb{Z}_{\mathsf{L}})$ to denote the union, over all logspace-computable sequence of moduli (m_n), of $\#\mathsf{AC}^1(\mathbb{Z}_{m_n})$. In the special case when $m_n = p$ for all n, we obtain the class $\#\mathsf{WSAC}^1(\mathbb{F}_p)$.*

We refrain from defining a weakly semiunbounded analog of the $\Lambda\mathsf{P}$ classes, because it is easy to show that they are equivalent to the $\Lambda\mathsf{P}$ classes, since AC^0 circuits can add logarithmically-many numbers, given in binary.

We improve on [AJMV98, Theorem 4.3] by showing AC^1 is contained in $\#\mathsf{WSAC}^1(\mathbb{F}_2)$; note that all polynomials in $\#\mathsf{WSAC}^1(\mathbb{F}_p)$ have degree $n^{O(\log\log n)}$, and note also that the class of functions considered in [AJMV98] is not obviously even in TC^1. In addition, we improve on [AJMV98] by reducing not merely AC^1, but also $\mathsf{AC}^1[p]$ for any prime p. This includes $\Lambda\mathsf{P}(\mathbb{F}_p)$ for any p such that $S(p-1) \subseteq \{2\}$.

Theorem 8. *Let p be any prime. Then $\mathsf{AC}^1[p] = \#\mathsf{WSAC}^1(\mathbb{F}_p)$.*

Proof. The inclusion $\#\mathsf{WSAC}^1(\mathbb{F}_p) \subseteq \mathsf{AC}^1[p]$ is straightforward. The proof of Corollary 5 shows how to simulate semiunbounded fan-in circuits over $\mathbb{F}_p$ by $\mathsf{AC}^1[p]$ circuits. We merely need to add to that construction, to show how to handle multiplication gates of logarithmic fan-in. Let g be a multiplication gate computing the product of the gates $h_1, \ldots, h_{c\log n}$. As in the proof of Corollary 5, the simulating $\mathsf{AC}^1[p]$ circuit will record the binary representation of the value of each of the gates h_i. Thus the value of g depends on only $O(\log n)$ binary bits of the simulating circuit, and the bits of the binary representation of the value of g can be computed by a logspace-uniform DNF expression. This yields the desired $\mathsf{AC}^1[p]$ circuit.

For the proof of the converse inclusion, the main technical ingredient involved is the following lemma from [AJMV98]. (In [AJMV98] the lemma is stated only for MOD_2, but the proof carries over to any MOD_m gate with only trivial changes. For completeness, a detailed proof may be found in Appendix A.)

Lemma 1. *[AJMV98] Let m be any natural number, $m > 1$. For each $\ell \in \mathbb{N}$, there is a family of constant-depth, polynomial-size, probabilistic circuits consisting of unbounded-fan-in MOD_m gates, AND gates of fan-in $O(\log n)$, and $O(\log n)$ probabilistic bits, computing the OR of n bits, with error probability $< 1/n^\ell$.*

Now we follow closely the proof of [AJMV98, Theorem 4.3].

Take an $\mathsf{AC}^1[p]$ circuit, replace all AND gates by OR and MOD_p gates (using DeMorgan's laws), and then replace each OR gate in the resulting circuit with the subcircuit guaranteed by Lemma 1 (for l chosen so that n^l is much larger than the size of the original circuit), with the *same* $O(\log n)$ probabilistic bits re-used in each replacement circuit. The result is a probabilistic, polynomial-size circuit that, with high probability, provides the same output as the original circuit. Note that replacing AND gates by $\times$ and replacing each MOD_p gate g having wires from h_i with a subcircuit of the form $1 - (\sum_i h_i)^{p-1}$, one obtains an arithmetic circuit over the integers, whose value mod p is equal to the output of the original $\mathsf{AC}^1[p]$ circuit with high probability. (This is one place where we use the fact that p is prime.) The circuit has depth $O(\log n)$, and has unbounded fan-in + gates, and all $\times$ gates have fan-in $O(\log n)$, and thus it is a weakly semiunbounded fan-in circuit.

Create $n^{O(1)}$ copies of this probabilistic circuit, one copy for each sequence of probabilistic bits; call these circuits $D_1, D_2, \ldots, D_{n^c}$. Note that each D_i computes a value in $\{0, 1\}$. Note also that $1 - D_i$ is also computable in $\#\mathsf{WSAC}^1(\mathbb{F}_p)$. Thus we can feed these values into an arithmetic NC^1 circuit computing MAJORITY (using the fact that all functions in NC^1 are in $\#\mathsf{NC}^1$ [CMTV98]). The resulting circuit is equivalent to our original $\mathsf{AC}^1[p]$ circuit. □

We especially call attention to the following corollary, which shows that, over $\mathbb{F}_2$, polynomial size logarithmic depth arithmetic circuits of degree $n^{O(\log n)}$ and of degree $n^{O(\log \log n)}$ represent precisely the same functions!

Corollary 7. $\#\mathsf{WSAC}^1(\mathbb{F}_2) = \#\mathsf{AC}^1(\mathbb{F}_2) = \mathsf{AC}^1[2] = \Lambda\mathsf{P}(\mathbb{F}_3)$.

Proof. The containment $\#\mathsf{WSAC}^1(\mathbb{F}_2) \subseteq \#\mathsf{AC}^1(\mathbb{F}_2)$ is immediate from the definition (since $\#\mathsf{WSAC}^1(\mathbb{F}_2)$ circuits are a restricted form of $\#\mathsf{AC}^1(\mathbb{F}_2)$ circuits). The second equality is from Theorem 4. The equality $\mathsf{AC}^1[2] = \Lambda\mathsf{P}(\mathbb{F}_3)$ is from Theorem 3. The inclusion $\mathsf{AC}^1[2] \subseteq \#\mathsf{WSAC}^1(\mathbb{F}_2)$ is from Theorem 8. □

If we focus on the Boolean classes, rather than on the arithmetic classes, then we obtain a remarkable collapse.

Theorem 9. *Let $m \in \mathbb{N}$. Then $\mathsf{AC}^1[m] = \log\text{-}\mathsf{AC}^1[m]$. (Recall the definition of $g\text{-}\mathsf{AC}^1[m]$ from Definition 4.)*

Proof. The proof of Theorem 8 begins with the statement of Lemma 1, which holds for any modulus m. The proof then uses Lemma 1 to replace a general $\mathsf{AC}^1[m]$ circuit by

an equivalent probabilistic circuit with unbounded fan-in MOD_m gates and AND gates with logarithmic fan-in, using only $O(\log n)$ probabilistic bits.

The proof of Theorem 8 proceeds to modify this to obtain an arithmetic circuit. Instead, we simply make polynomially-many copies of this Boolean circuit (one copy for each probabilistic sequence), and take the majority vote of these copies. □

Using Theorem 4 it follows that arithmetic AC^1 circuits over any finite field $\mathbb{F}_p$ can be simulated by Boolean circuits with MOD gates and small fan-in AND gates. It remains open whether this in turn leads to small-degree arithmetic circuits over $\mathbb{F}_p$ when $p > 2$, and also whether the fan-in of the AND gates can be sublogarithmic, without loss of power.

When m is composite, Theorem 9 can be improved to obtain an even more striking collapse, by invoking the work of Hansen and Koucký [HK10].

Theorem 10. *Let m not be a prime power. Then $\mathsf{AC}^1[m] = 2\text{-}\mathsf{AC}^1[m]$.*

Proof. Let $p \neq q$ where $\{p, q\} \subseteq S(m)$. It suffices to show how to construct a family of $2\text{-}\mathsf{AC}^1[m]$ circuits to simulate a given $\mathsf{AC}^1[m]$ circuit family.

Hansen and Koucký showed [HK10, Lemma 3.5] that, for every $c > 1$ there is a constant-depth probabilistic circuit composed of MOD_{pq} gates that computes the OR of n variables, using only $O(\log n)$ probabilistic bits, and having error probability less than $1/n^c$. Thus we can replace each unbounded fan-in AND and OR gate in the $\mathsf{AC}^1[m]$ with the corresponding circuit (possibly with negation gates) guaranteed by [HK10]. The MOD_{pq} gates can be replaced with MOD_m gates via standard techniques, as in the proof of Theorem 3. By choosing a suitably large value for c, the resulting probabilistic circuit simulates the original circuit with small error probability.

Now, as in the proof of Theorem 9 we can make polynomially-many copies of the probabilistic circuit, hardwiring in different values for the probabilistic bits, and take the majority vote. □

It might be useful to have additional examples of algebras, where some degree reduction can be accomplished. Thus we also offer the following theorem:

Theorem 11. *Let p be any prime. Then $\mathsf{AC}^1[p] \subseteq \mathsf{L}^{\#\mathsf{WSAC}^1(\mathbb{Z}_\mathsf{L})}$.*

Proof. We start with the sequence of circuits $D_1, D_2, \ldots, D_{n^c}$ created in the proof of the preceding theorem. We now make use of the "Toda polynomials" introduced in [Tod91]. For example, there is an explicit construction in [BT94] of a polynomial P_k of degree $2k - 1$ such that $(y \bmod p) \in \{0, 1\}$ implies $P_k(y) \bmod p^k = y \bmod p$. It is observed in [AG94] that, for $k = O(\log n)$, the polynomial P_k can be implemented via logspace-uniform constant-depth circuits over the integers. Thus, by replacing each multiplication gate with a tree of fan-in two, the polynomial can be implemented by a semiunbounded fan-in circuit of logarithmic depth. Applying this polynomial to the output of each circuit D_i, we obtain a $\#\mathsf{WSAC}^1(\mathbb{Z})$ circuit whose value mod p is the same as the output of the original $\mathsf{AC}^1[p]$ circuit with high probability, and with the additional property that the output of the circuit, when represented in p-ary notation, has all of the $c \log n$ low-order symbols of the result equal to zero (except possibly the lowest-order symbol). We will choose c to be the constant such that there are $c \log n$ probabilistic bits). Call the resulting circuit E_i.

Now create a circuit whose output gate computes $\sum_i E_i$. The output gate of the resulting $\#\mathsf{WSAC}^1(\mathbb{Z})$ circuit records a number whose low-order $c \log n$ positions (in p-ary notation) records the number of the n^c copies that output 1. If this number is greater than $n^c/2$, then the original circuit accepted its input; otherwise it rejected its input.

In order to compute this number using $\#\mathsf{WSAC}^1(\mathbb{Z}_{\mathsf{L}})$ instead of $\#\mathsf{WSAC}^1(\mathbb{Z})$, we use this logspace-computable sequence of moduli: $m_n = p^n$. Evaluating the arithmetic over $\mathbb{Z}_{p^n}$ gives the number represented by the low-order n positions of the result, in p-ary notation. A logspace oracle machine, upon being given this number (say, in binary notation) can compute the value of this number modulo $p^{1+c \log n}$ and determine if that number is greater than $n^c/2$, and can thereby determine if the original circuit accepted its input. □

It is natural to wonder whether this theorem can be extended, to allow composite moduli. A direct application of the techniques of [AG94, BT94, Yao90] requires multiple applications of the Toda polynomials, and this in turn results in circuits of superlogarithmic depth.

Using Theorems 3 and 4 we obtain the following.

Corollary 8. *If p is a Fermat prime, then $\Lambda\mathsf{P}(\mathbb{F}_p) \subseteq \mathsf{L}^{\#\mathsf{WSAC}^1(\mathbb{Z}_{\mathsf{L}})}$.*

5 Conclusions, Discussion, and Open Problems

We have introduced the complexity classes $\Lambda\mathsf{P}(R)$ for various algebraic structures R, and have shown that they provide alternative characterizations of well-known complexity classes. Furthermore, we have shown that arithmetic circuit complexity classes corresponding to polynomials of degree $n^{O(\log \log n)}$ also yield new characterizations of complexity classes, such as the equality

$$\mathsf{AC}^1[p] = \log - \mathsf{AC}^1[p] = \#\mathsf{WSAC}^1(\mathbb{F}_p).$$

Furthermore, in the case when $p = 2$, we obtain the additional collapse

$$\#\mathsf{AC}^1(\mathbb{F}_2) = \mathsf{AC}^1[2] = \log - \mathsf{AC}^1[2] = \#\mathsf{WSAC}^1(\mathbb{F}_2),$$

showing that algebraic degree $n^{O(\log n)}$ and $n^{O(\log \log n)}$ have equivalent expressive power, in this setting.

We have obtained the following new characterizations of ACC^1:

$$\mathsf{ACC}^1 = \bigcup_p \#\mathsf{AC}^1(\mathbb{F}_p) = \bigcup_p \Lambda\mathsf{P}(\mathbb{F}_p) = \bigcup_m 2\text{-}\mathsf{AC}^1[m].$$

That is, although ACC^1 corresponds to unbounded fan-in arithmetic circuits of logarithmic depth, and to unbounded fan-in Boolean circuits with modular counting gates, no power is lost if the addition gates have bounded fan-in (in the arithmetic case) or if only the modular counting gates have unbounded fan-in (in the Boolean case). It remains unknown if $\bigcup_m \mathsf{VP}(\mathbb{Z}_m)$ captures all of ACC^1. (That is, in the arithmetic case, is the $\Lambda\mathsf{P}$ model really stronger than the VP model?)

We believe that it is fairly likely that several of our theorems can be improved. For instance:

- Perhaps Theorems 9 and 10 can be improved, to show that for all m, $\mathsf{AC}^1[m] = 2\text{-}\mathsf{AC}^1[m]$. Note that this is already known to hold if m is not a prime power. By Corollary 5 this would show that $\mathsf{VP}(\mathbb{F}_p) = \mathsf{AC}^1[p]$ for all primes p. It would also show that $\#\mathsf{AC}^1(\mathbb{F}_2) = \mathsf{VP}(\mathbb{F}_2) = \Lambda\mathsf{P}(\mathbb{F}_p)$ for every Fermat prime p. (We should point out that this would imply that $\mathsf{AC}^1 \subseteq \mathsf{VP}(\mathbb{F}_p)$ for every prime p, whereas even the weaker inclusion $\mathsf{SAC}^1 \subseteq \mathsf{VP}(\mathbb{F}_p)$ is only known to hold non-uniformly [GW96].)

- Can Corollary 8 be improved to hold for all primes p, or even for $\Lambda\mathsf{P}(\mathbb{F}_{p_n})$? The latter improvement would show that $\mathsf{TC}^1 \subseteq \mathsf{L}^{\#\mathsf{WSAC}^1(\mathbb{Z}_\mathsf{L})}$.

- Perhaps one can improve Theorem 11, to achieve a simulation of degree $n^{O(1)}$. Why should $n^{O(\log\log n)}$ be optimal? Perhaps this could also be improved to hold for composite moduli?

- If some combinations of the preceding improvements are possible, TC^1 would reduce to $\mathsf{VP}(\mathbb{Q})$, which would be a significant step toward the Immerman-Landau conjecture.

We began this investigation, wondering if the equality $\mathsf{VP}(B_2) = \Lambda\mathsf{P}(B_2)$ could carry over to any other algebraic structure. We think that it appears as if $\mathsf{VP}(\mathbb{F}_p)$ and $\Lambda\mathsf{P}(\mathbb{F}_p)$ are incomparable for every non-Fermat prime $p > 2$, since $\mathsf{VP}(\mathbb{F}_p) = 2\text{-}\mathsf{AC}^1[p]$ and $\Lambda\mathsf{P}(\mathbb{F}_p) = 2\text{-}\mathsf{AC}^1[S(p-1)]$. That is, these classes correspond to circuits with access to modular counting gates for completely different sets of primes. For Fermat primes we have $\Lambda\mathsf{P}(\mathbb{F}_p) = \log\text{-}\mathsf{AC}^1[2]$ and again the VP and $\Lambda\mathsf{P}$ classes seem incomparable.

For the special case of $p = 2$, we have $\mathsf{VP}(\mathbb{F}_2) = 2\text{-}\mathsf{AC}^1[2]$ and $\Lambda\mathsf{P}(\mathbb{F}_2) = \mathsf{AC}^1$. We hold out some hope that $\mathsf{VP}(\mathbb{F}_2) = \mathsf{AC}^1[2]$, in which case it would appear that the VP class could be *more* powerful than the $\Lambda\mathsf{P}$ class – but based on current knowledge it also appears possible that the VP and $\Lambda\mathsf{P}$ classes are incomparable in this case too.

Acknowledgments The first and third authors acknowledge the support of NSF grants CCF-0832787 and CCF-1064785. The second author was supported in part by NSF grant CCF-1018060. We also acknowledge stimulating conversations with Meena Mahajan, which occurred at the 2014 Dagstuhl Workshop on the Complexity of Discrete Problems (Dagstuhl Seminar 14121), and illuminating conversations with Stephen Fenner and Michal Koucký, which occurred at the 2014 Dagstuhl Workshop on Algebra in Computational Complexity (Dagstuhl Seminar 14391). We also thank Rutgers colleagues Richard Bumby, John Miller and Steve Miller, for helpful pointers to the literature, as well as helpful feedback from Pascal Koiran.

References

[AAD00] M. Agrawal, E. Allender, and S. Datta. On TC^0, AC^0, and arithmetic circuits. *Journal of Computer and System Sciences*, 60:395–421, 2000.

[AG94] E. Allender and V. Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23:1026–49, 1994.

[AJMV98] E. Allender, J. Jiao, M. Mahajan, and V. Vinay. Non-commutative arithmetic circuits: Depth reduction and size lower bounds. *Theoretical Computer Science*, 209:47–86, 1998.

[AK10] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *Journal of the ACM*, 57:14:1 – 14:36, 2010.

[AL13] E. Allender and K.-J. Lange. Symmetry coincides with nondeterminism for time-bounded auxiliary pushdown automata. *Theory of Computing*, 2013. To appear; an earlier version appeared in CCC 2010.

[ARZ99] E. Allender, K. Reinhardt, and S. Zhou. Isolation, matching, and counting: Uniform and nonuniform upper bounds. *Journal of Computer and System Sciences*, 59(2):164–181, 1999.

[BCD+89] A. Borodin, S. A. Cook, P. W. Dymond, W. L. Ruzzo, and M. Tompa. Two applications of inductive counting for complementation problems. *SIAM Journal on Computing*, 18:559–578, 1989. See Erratum in SIAM J. Comput. 18, 1283.

[BCK+14] Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In *ACM Symposium on Theory of Computing (STOC)*, pages 857–866, 2014.

[BT94] R. Beigel and J. Tarui. On ACC. *Computational Complexity*, 4:350–366, 1994. Special issue on circuit complexity.

[Bür99] Peter Bürgisser. On the structure of Valiant's complexity classes. *Discrete Mathematics & Theoretical Computer Science*, 3(3):73–94, 1999.

[Bür00] Peter Bürgisser. Cook's versus Valiant's hypothesis. *Theoretical Computer Science*, 235(1):71–88, 2000.

[CDL01] A. Chiu, G.I. Davida, and B. Litow. Division in logspace-uniform NC^1. *RAIRO Theoretical Informatics and Applications*, 35:259–276, 2001.

[CMTV98] Hervé Caussinus, Pierre McKenzie, Denis Thérien, and Heribert Vollmer. Nondeterministic NC^1 computation. *Journal of Computer and System Sciences*, 57(2):200–212, 1998.

[CRS95] Suresh Chari, Pankaj Rohatgi, and Aravind Srinivasan. Randomness-optimal unique element isolation with applications to perfect matching and related problems. *SIAM Journal on Computing*, 24(5):1036–1050, 1995.

[GG81] Ofer Gabber and Zvi Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, 22(3):407–420, 1981.

[GLS01] Georg Gottlob, Nicola Leone, and Francesco Scarcello. The complexity of acyclic conjunctive queries. *Journal of the ACM*, 48(3):431–498, 2001.

[GW96] Anna Gál and Avi Wigderson. Boolean complexity classes vs. their arithmetic analogs. *Random Struct. Algorithms*, 9(1-2):99–111, 1996.

[HAB02] William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65:695–716, 2002.

[HK10] Kristoffer Arnsfelt Hansen and Michal Koucký. A new characterization of ACC^0 and probabilistic CC^0. *Computational Complexity*, 19(2):211–234, 2010.

[IL95] N. Immerman and S. Landau. The complexity of iterated multiplication. *Information and Computation*, 116:103–116, 1995.

[IZ89] Russell Impagliazzo and David Zuckerman. How to recycle random bits. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 248–253, 1989.

[KP11] Pascal Koiran and Sylvain Perifel. Interpolation in Valiant's theory. *Computational Complexity*, 20(1):1–20, 2011.

[LMSV01] Clemens Lautemann, Pierre McKenzie, Thomas Schwentick, and Heribert Vollmer. The descriptive complexity approach to LOGCFL. *Journal of Computer and System Sciences*, 62(4):629–652, 2001.

[Pet02] Holger Petersen. The membership problem for regular expressions with intersection is complete in LOGCFL. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, number 2285 in Lecture Notes in Computer Science, pages 513–522. Springer, 2002.

[RA00] K. Reinhardt and E. Allender. Making nondeterminism unambiguous. *SIAM Journal on Computing*, 29:1118–1131, 2000.

[RT92] J. Reif and S. Tate. On threshold circuits and polynomial computation. *SIAM Journal on Computing*, 21:896–908, 1992.

[Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *ACM Symposium on Theory of Computing (STOC)*, pages 77–82, 1987.

[Tod91] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20:865–877, 1991.

[Val79] L.G. Valiant. Completeness classes in algebra. In *Proc. 11th ACM STOC*, pages 249–261, 1979.

[Ven91] H. Venkateswaran. Properties that characterize LOGCFL. *Journal of Computer and System Sciences*, 43:380–404, 1991.

[Vin91] V Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proceedings of 6th Structure in Complexity Theory Conference*, pages 270–284, 1991.

[Vol99] H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer-Verlag New York Inc., 1999.

[VSBR83] L.G. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM Journal on Computing*, 12(4):641–644, 1983.

[Xyl11] T. Xylouris. On the least prime in an arithmetic progression and estimates for the zeros of Dirichlet L-functions. *Acta Arithmetica*, 150:65–91, 2011.

[Yao90] Andrew Chi-Chih Yao. On ACC and threshold circuits. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 619–627, 1990.

6 Appendix A: Proof of Lemma 1

In this section, we present a detailed proof of Lemma 1, showing the adjustments that need to be made, in order to deal with arbitrary MOD_m gates.

Here is a reminder of the statement of Lemma 1: *Let m be any natural number, $m > 1$. For each $\ell \in \mathbb{N}$, there is a family of constant-depth, polynomial-size, probabilistic circuits consisting of unbounded-fan-in* MOD_m *gates,* AND *gates of fan-in $O(\log n)$, and $O(\log n)$ probabilistic bits, computing the OR of n bits, with error probability $< 1/n^{\ell}$.*

Proof. Our presentation here is a slight adjustment of the proof in [AJMV98]. There are no significant changes in the proof, which relies crucially on the fact that one can replace an OR gate with a MOD gate, when there is a guarantee that at most one of the inputs to the OR gate evaluates to 1.

The construction in [CRS95] gives a depth 5 probabilistic circuit that computes the NOR correctly with probability at least $\frac{1}{2}$ and uses $O(\log n)$ random bits. More precisely, using the terminology of [CRS95], let $m' = \lceil \log n \rceil$, let $S = \{1, \ldots, m'\}$, and let $\mathcal{F}$ be the collection of subsets of S, such that $A \in \mathcal{F}$ iff the bit string k of length $m' = \lceil \log n \rceil$ representing the characteristic sequence of A corresponds to a binary number $k \leq n$ such that the k-th bit of the input sequence $x_1, \ldots, x_n$ has value 1. That is, the OR of $x_1, \ldots, x_n$ evaluates to 1 iff $\mathcal{F}$ is not empty. The strategy of [CRS95] is to use probabilistic bits to define a way of assigning a "weight" to each set $A_k \in \mathcal{F}$ so that if $\mathcal{F}$ is not empty, then with high probability there is a unique element of $\mathcal{F}$ having minimum weight. The next paragraph explains how this is done.

Let $c = \lceil \log m' \rceil$ and let $t = \lceil m'/c \rceil$. For any $1 \leq i \leq m'$ and $0 \leq j \leq t-1$, define $b_{i,j}$ as follows:

$$b_{i,j} = \begin{cases} 2^{i-jc} & \text{if } jc < i \leq (j+1)c \\ 0 & \text{otherwise} \end{cases}$$

(It may help the reader's intuition to consider an m'-bit sequence $k = k_1, \ldots, k_{m'}$. Divide this sequence into blocks; Block(j) has positions $k_{jc+1}, k_{jc+2}, \ldots, k_{(j+1)c}$. Clearly, $k_{m'}$ is in Block(k_{t-1}). Now, if $k_i \notin$ Block(j), then $b_{i,j} = 0$, else $b_{i,j} = 2^{i-jc}$. Note that $i - jc$ is the position of k_i within Block(j).)

Choose t numbers $r_0, \ldots, r_{t-1}$ in the range $1 \leq r_j \leq 50 \log^5 n$ uniformly and independently at random (and note that this amounts to choosing $O(\log n)$ random bits). Finally, define w_i to be equal to $\sum_{j=0}^{t-1} b_{i,j} r_j$. The weight of a set A is then $\sum_{i \in A} w_i$. The

analysis in Proposition 2 of [CRS95] shows that if $\mathcal{F}$ is not empty, then with probability at least .99, there is a unique minimal weight set in $\mathcal{F}$.

This paragraph explains how to implement this system as a uniform constant-depth circuit. Note first that for any $k \leq n$ and for any fixed $p \leq \log^7 n$, there is a depth 2 circuit of MOD_m gates and small-fan-in AND gates that evaluates to 1 iff the weight of A_k is equal to p. Here A_k is that subset of S whose characteristic sequence is the binary representation of k. (To see this, note that the only inputs to this circuit are the $O(\log n)$ probabilistic bits. Thus the DNF for this function can be computed in logspace, and the OR gate at the root can be replaced by a MOD_m gate with $m-1$ additional 1 inputs. Here we are making use of the fact that there can only be one of the AND gates that feed into to the MOD_m gate that returns 1, namely the one where the weight of $A_k = p$.)

Taking the AND of this circuit with the input bit x_k results in a depth three circuit that evaluates to 1 iff $A_k \in \mathcal{F}$ and the weight of A_k is equal to p. Thus there is a polynomial-size depth-4 circuit with a MOD_m gate at the root (with $m-1$ additional 1 inputs) that evaluates to 1 iff the number of sets in $\mathcal{F}$ that have weight p is equivalent to 1 mod m. Hence there is a uniform depth-5 circuit with an OR at the root that evaluates to 1 iff there is some weight p such that the number of sets in $\mathcal{F}$ having weight p is equivalent to 1 mod m. By the remarks in the preceding paragraph, if the OR of $x_1, \ldots, x_n$ evaluates to 1, then with probability at least .99, our depth-5 circuit will also. (Clearly, if the OR is zero, then the depth-5 circuit also evaluates to zero.) If we replace the OR gate at the root with AND and negate each of the MOD_m gates that feed into that OR gate (recalling that a unary MOD_m gate is a NOT gate) we obtain our desired circuit for the NOR function. Let us denote this circuit by $C(x, r)$.

It remains only to reduce the error probability from $\frac{1}{100}$ to $\frac{1}{n^l}$, without using too many additional probabilistic bits. Consider a graph with vertices for each of our $O(\log n)$-bit probabilistic sequences, the edge relation is given by the construction of an expander graph presented in [GG81], where each vertex has degree five. Inspection of [GG81] shows that, in logspace, one can take as input one of our original probabilistic sequences r as well as a new probabilistic sequence $s \in \{1,2,3,4,5\}^{c\ell \log n}$ (for some constants c and ℓ) and output the vertex r' reached by starting in vertex r and following the sequence of edges indicated by s. Since this function depends on only $O(\log n)$ bits, the DNF for this function can be computed in logspace, and (as above) can be implemented using a MOD_m gate and AND gates of small fan-in. Let this circuit be denoted by $R(r, s)$.

Thus we can construct a constant-depth circuit that computes the AND for all $i \leq cl \log n$ of $C(x, R(r, s[1..i]))$ (where $s[1..i]$ denotes the prefix of s of length i, where r and s are probabilistically chosen. By Section 2 of [IZ89], this circuit computes the NOR correctly with probability $1 - \frac{1}{n^l}$. Adding a MOD_m gate at the root allows us to compute the OR, as desired. This completes the proof of the lemma. □

7 Appendix B: diagram of new macro and micro inclusions

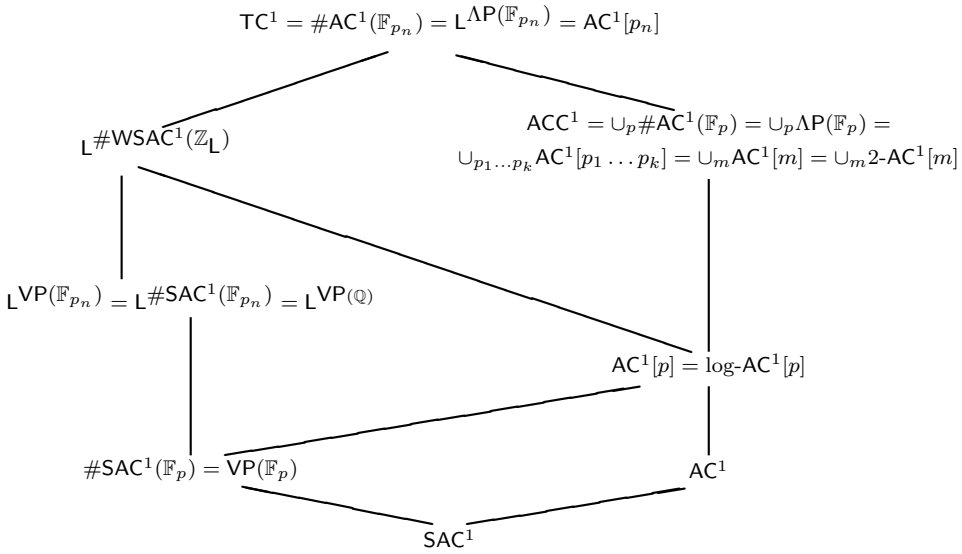

Figure 1: Macro inclusions within TC^1

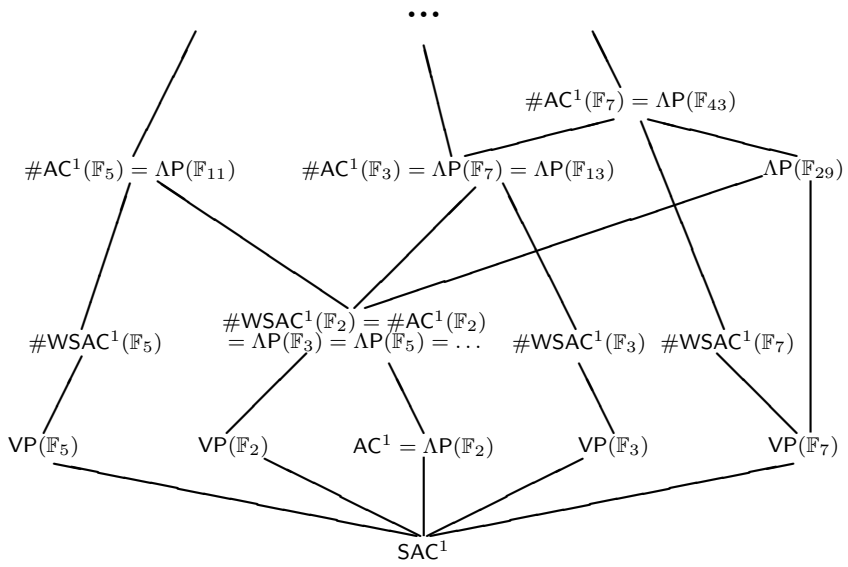

Figure 2: Micro inclusions within ACC^1

ECCC ISSN 1433-8092
http://eccc.hpi-web.de