Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 131 (2013)

Collapsing Exact Arithmetic Hierarchies

Nikhil Balaji and Samir Datta

Chennai Mathematical Institute
{nikhil,sdatta}@cmi.ac.in

Abstract. We provide a uniform framework for proving the collapse of the hierarchy $\mathsf{NC}^1(\mathcal{C})$ for an exact arithmetic class $\mathcal{C}$ of polynomial degree. These hierarchies collapse all the way down to the third level of the AC^0-hierarchy, $\mathsf{AC}^0_3(\mathcal{C})$. Our main collapsing exhibits are the classes

$$\mathcal{C} \in \{\mathsf{C}_=\mathsf{NC}^1, \mathsf{C}_=\mathsf{L}, \mathsf{C}_=\mathsf{SAC}^1, \mathsf{C}_=\mathsf{P}\}.$$

$\mathsf{NC}^1(\mathsf{C}_=\mathsf{L})$ and $\mathsf{NC}^1(\mathsf{C}_=\mathsf{P})$ are already known to collapse [1,18,19].
We reiterate that our contribution is a framework that works for *all* these hierarchies. Our proof generalizes a proof from [8] where it is used to prove the collapse of the $\mathsf{AC}^0(\mathsf{C}_=\mathsf{NC}^1)$ hierarchy. It is essentially based on a polynomial degree characterization of each of the base classes.

1 Introduction

Collapsing hierarchies has been an important activity for structural complexity theorists through the years [12,21,14,23,18,17,4,11]. We provide a uniform framework for proving the collapse of the NC^1 hierarchy over an exact arithmetic class. Using our method, such a hierarchy collapses all the way down to the AC^0_3 closure of the class.

Our main collapsing exhibits are the NC^1 hierarchies over the classes $\mathsf{C}_=\mathsf{NC}^1$, $\mathsf{C}_=\mathsf{L}$, $\mathsf{C}_=\mathsf{SAC}^1$, $\mathsf{C}_=\mathsf{P}$. Two of these hierarchies, viz. $\mathsf{NC}^1(\mathsf{C}_=\mathsf{L})$, $\mathsf{NC}^1(\mathsf{C}_=\mathsf{P})$, are already known to collapse ([1,19,18]) while a weaker collapse is known for a third one viz. that of $\mathsf{AC}^0(\mathsf{C}_=\mathsf{NC}^1)$. We reiterate that our contribution is a simple proof that works for all these hierarchies. Our proof is a generalization of a proof from [8] who used it to prove the collapse of the $\mathsf{C}_=\mathsf{NC}^1$ hierarchy and is essentially based on a polynomial degree characterization of each of the corresponding arithmetic classes.

The most well known amongst the exact arithmetic circuit hierarchy collapses is the collapse of the NC^1 hierarchy over $\mathsf{C}_=\mathsf{L}$. This was proved by using the linear algebraic properties of $\mathsf{C}_=\mathsf{L}$ an elegant and non-trivial argument by Allender, Beals, and Ogihara [1]. We find it remarkable that our proof does not use any linear algebra (apart from the characterization of GapL functions as being exactly the functions computed by weakly skew circuits [24], which involves a certain amount of linear algebra) to prove the collapse to $\mathsf{AC}^0_3(\mathsf{C}_=\mathsf{L})$.

The collapse of the NC^1 hierarchy over $\mathsf{C}_=\mathsf{P}$ to a constant level of the AC^0 hierarchy follows easily from the results by Ogihara from [19,18]. Our proof is quite orthogonal to the proofs there.

We would like to point out a notational quirk that we carry over from [8] by consistently calling the finite level of AC^0 circuit to which we collapse the various classes as

ISSN 1433-8092

AC^0_3 where we actually mean two layers of Boolean gates i.e $\wedge, \vee$. The reason for the subscript 3 being that we include negation gates while counting the depth contrary to popular usage.

1.1 Historical Perspective

Counting classes have been studied from the early days of complexity theory. Gill and Simon [10,22] were the first to introduce the class PP(Probabilistic polynomial time) which consists of all languages decidable by a nondeterministic Turing machine in polynomial time in which at least half of the computation paths lead to acceptance. This class gained importance due to Toda's theorem (which states that a polynomial time machine with access to a PP oracle is at least as powerful as the entire polynomial hierarchy) and Beigel, Reingold and Spielman [3] who proved that it is closed under intersection.

$\mathsf{C}_=\mathsf{P}$ is the humbler cousin of PP though it happens to be one of our main protagonists. It was introduced by Simon [22] and consists of languages where the acceptance and rejection probability is the same.

Fenner, Fortnow and Kurtz[9] introduced the notion of Gap to aid the study of structural complexity of these counting classes. For a class $\mathcal{C}$, captured by nondeterministic Turing machines, denote by $\mathsf{Gap}\mathcal{C}$ the class of functions expressible as the difference between the number of accepting and rejecting computations of some nondeterministic Turing machine. Then $\mathsf{C}_=\mathsf{P}$ is just those languages which have a zero gap (with an NP-machine).

The class $\mathsf{C}_=\mathsf{L}$ and the $\mathsf{NC}^1, \mathsf{AC}^0$ hierarchies over it have also received attention in literature [1,20,13]. In [1] it was first proved that the NC^1 hierarchy over $\mathsf{C}_=\mathsf{L}$ collapses all the way down to the first level of the hierarchy, $\mathsf{L}^{\mathsf{C}_=\mathsf{L}}$. Further collapse to $\mathsf{C}_=\mathsf{L}$ is not known because this class is not known to be closed under complement.

The class $\mathsf{C}_=\mathsf{NC}^1$ was probably mentioned explicitly for the first time in [6] where it is shown that (a uniform version of) it is contained in Logspace. Various hierarchies over this class including the Boolean, AC^0, and the arithmetic hierarchy were studied in [8].

To the extent of our knowledge, the only previous known occurence of the class $\mathsf{C}_=\mathsf{SAC}^1$ is in the context of deciding the properties of monomials computed by arithmetic circuits[16]. The classes SAC^1 and its arithmetic analogs $\#\mathsf{SAC}^1$, GapSAC^1 have been actively investigated in literature [28,2,15].

1.2 Our Results

In this paper, we extend and generalize the framework used in [8] to prove that the $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ for a NC^1-well-behaved class[1] $\mathcal{K}$ collapses to $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$. As a result, we obtain several results as corollaries:

1. $\mathsf{AC}^0(\mathsf{C}_=\mathsf{NC}^1) = \mathsf{NC}^1(\mathsf{C}_=\mathsf{NC}^1) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{NC}^1)$, improving on [8].
2. $\mathsf{AC}^0(\mathsf{C}_=\mathsf{L}) = \mathsf{NC}^1(\mathsf{C}_=\mathsf{L}) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{L})$, which gives an alternative proof of [1]
3. $\mathsf{AC}^0(\mathsf{C}_=\mathsf{SAC}^1) = \mathsf{NC}^1(\mathsf{C}_=\mathsf{SAC}^1) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{SAC}^1)$.

[1] see Definitions 4, 8

4. $\mathsf{AC}^0(\mathsf{C}_=\mathsf{P}) = \mathsf{NC}^1(\mathsf{C}_=\mathsf{P}) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{P})$, which gives an alternative proof of [18]

We prove that such a collapse can be made DLOGTIME-uniform. Note that we need to prove this strict uniformity of our collapse in order to exploit Venkateswaran's characterization of NP as exactly those languages that are decidable by DLOGTIME-uniform semi-unbounded circuit families of exponential size.

1.3 Proof Idea

The unifying feature of the aforementioned classes($\mathsf{C}_=\mathsf{NC}^1$, $\mathsf{C}_=\mathsf{L}$, $\mathsf{C}_=\mathsf{P}$, $\mathsf{C}_=\mathsf{SAC}^1$) is that roughly speaking, they can be viewed as languages accepted by the $\mathsf{C}_=$-version of an arithmetic circuit family of *polynomial degree*. Thus if we consider arithmetic formulas, arithmetic weakly skew circuits, arithmetic circuits of polynomial size and arithmetic circuits of exponential size respectively, then imposing the polynomial degree bound gives us exactly the classes GapNC^1, GapL, GapSAC^1 and GapP [6,24,26,27]. Applying the $\mathsf{C}_=$ operator to these arithmetic classes yields our candidate exact arithmetic classes. We also tacitly use that the class of circuits are closed under composition with formulas.

The other important point to notice is that the [8] proof which just shows the collapse of the $\mathsf{AC}^0(\mathsf{C}_=\mathsf{NC}^1)$ hierarchy to the third level can be extended to a collapse of $\mathsf{NC}^1(\mathsf{C}_=\mathsf{NC}^1)$. This follows by observing that the level by level collapse outlined in [8] does not blow up the degree to more than a polynomial value even though we need to perform it logarithmically many times. This is because of the Cook-Wilson relativization, which can be interpreted as saying that if the oracle circuits have polynomial degree, so is the final circuit. This idea can then be generalized to any circuit family of polynomial degree.

A simple but crucial observation makes it possible to extend the results from bounded fan-in circuits in [8] to circuits where this restriction does not hold e.g. in the cases of $\mathsf{C}_=\mathsf{SAC}^1$ and $\mathsf{C}_=\mathsf{P}$. This observation concerning Vandermonde Determinants is described in Proposition 1 and used in the proof of Lemma 1.

1.4 Organization of the paper

We present preliminaries in Section 2. In Section 3, we state our main result and list some immediate corollaries of our result. We discuss possible future directions to our work in Section 5.

2 Preliminaries

We mention the standard complexity classes that we will use. For definitions and important results regarding these classes, we refer the reader to a standard text like [29].

An *arithmetic circuit* is a directed acyclic graph(DAG) with nodes labelled by $\{\times, +\} \cup X \cup \{-1\}$, where X is the set of input variables. Note that -1 is the only constant necessary, and we can use it to generate $1 = (-1) \times (-1)$, $0 = (-1) + 1$, and use $-1, 0, 1$ to generate any integer.

Given an arithmetic circuit C, the formal degree of the circuit is defined inductively as follows: Every input variable and the constant -1 have degree[2] 1. If C_1 and C_2 are two subcircuits of degree d_1 and d_2 fed in to a $\times$ gate (respectively $+$ gate), then the degree of the $\times$ gate ($+$ gate) is $d_1 + d_2$ (respectively $\max(d_1, d_2)$).

An $(m \times m)$ Vandermonde matrix V is one where the (i, j)-th entry of V, $V_{ij} = i^j$.

Fact 1 *The absolute value of determinant of V is equal to $\prod_{i<j;i,j\leq m}(j-i) = \prod_{k=1}^{m} k!$*

It is important to note that the determinant of an $(m \times m)$-Vandermonde matrix(with entries that are indeterminates) can be computed by an arithmetic circuit of degree m. But since we start out with just the constant -1, we will have to construct these numbers $1, \ldots, m$ and use them to find the the entries of the matrix, which can be done via repeated squaring and addition with a circuit of size at most $O(m^2)$, depth $O(\log m)$ and degree $O(m^2 \log m)$, for every entry of the matrix. Also, the value of this determinant is at most $\prod_{k=1}^{m} k! \leq (m!)^m \leq m^{m^2}$ which can be computed by an arithmetic circuit of size at most $O(\binom{m}{2} m^2) = O(m^4)$, depth $O(\log m)$ and degree $O(m^4 \log m)$.

We start with the usual definitions of AC^0 and NC^1 hierarchies over Boolean complexity classes.

Definition 1. *Let $\mathcal{C}$ be a Boolean complexity class. $\mathsf{AC}^0(\mathcal{C})$ is the class of languages recognized by AC^0 circuits with additional oracle gates(of unbounded fan-in) for $\mathcal{C}$.*

Defining the NC^1 hierarchy naively as above will yield a circuit where there could be $O(\log n)$ many oracle gates on any path, which will stand in contrast to the definition of NC^1 as that of circuits with bounded fan-in gates. The definition due to Cook and Wilson[7,30] gives a reasonable model, which avoids these problems:

Definition 2. *(Cook-Wilson) Let $\mathcal{C}$ be a Boolean complexity class. $\mathsf{NC}^1(\mathcal{C})$ is the class of languages recognized by NC^1 circuits with additional oracle gates for $\mathcal{C}$, where an oracle gate of fan-in k is charged $\log k$ towards the depth of the circuit.*

We note that the *small blob chains* property defined in [8] is essentially modelled by the Cook-Wilson relativization for NC^1 as given in Definition 2. An easy inclusion follows from the definitions above: For any boolean complexity class $\mathcal{C}$, we have $\mathsf{AC}^0(\mathcal{C}) \subseteq \mathsf{NC}^1(\mathcal{C})$. Next, we abstract the classes of base circuits over which we will consider various hierarchies:

Definition 3. *Let $\mathcal{K}$ be a class of arithmetic circuits. Then $\mathsf{C}_=\mathcal{K}$ is the class of languages recognized by the circuits from the class $\mathcal{K}$ with an additional gate at the top that compares the output to zero to produce a Boolean output.*

We will abuse notation to identify $\mathsf{C}_=\mathcal{K}$ with the class of circuits recognizing the class of represented languages.

[2] Note the non-standard convention to account for the degree of a constant. We do this so as to account for the size and degree contribution of the Vandermonde matrices, which we will use extensively in our results.

Definition 4. *Let $\mathcal{K}_1, \mathcal{K}_2$ be classes of circuits. Then $\mathcal{K}_1 \circ \mathcal{K}_2$ is the class consisting of circuits with a circuit from $\mathcal{K}_1$ at the top, all of whose inputs are circuits from $\mathcal{K}_2$. We say that the class $\mathcal{K}_2$ is closed under composition with $\mathcal{K}_1$ if $\mathcal{K}_2 \circ \mathcal{K}_1 \subseteq \mathcal{K}_2$.*

Definition 5. *(Cook-Wilson for Arithmetic Classes) Let $\mathcal{K}$ be an arithmetic complexity class of polynomial formal degree. A special case when $\mathcal{K}$ is composed with itself $O(\log n)$ many times is denoted by,*

$$\mathcal{K}^{(l)} = \overbrace{\mathcal{K} \circ \mathcal{K} \ldots \circ \mathcal{K}}^{O(\log n) \text{ times}}$$

where along any path in the $\mathcal{K}^{(l)}$ circuit, the product of the degrees of $\mathcal{K}$ circuits is bounded by a polynomial in the number of input variables to the $\mathcal{K}^{(l)}$ circuit.

Throughout the paper, we use a very specific type of AC^0 circuit, namely, an AC^0_3 circuit which is essentially a boolean circuit of depth 3 consisting of an $\vee$ gate at the root, followed by a layer of $\wedge$ gates, which are fed by $\mathsf{C}_=\mathcal{K}$ or $\mathsf{coC}_=\mathcal{K}$ oracle gates. We will refer to such circuits as $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$ circuits.

Definition 6 (Toda[24]). *A gate in a circuit is said to be weakly skew if for any multiplication gate α with children β and γ,one of the two sub-circuits C_β or C_γ is only connected to the rest of the circuit by the wire going to α. A weakly skew circuit is one where all the multiplication gates are weakly skew.*

A simple consequence of this definition is that every formula is a weakly skew circuit.

3 Exact Arithmetic Hierarchies

Definition 7. *Some examples of natural arithmetic circuit classes are as follows:*

- *$\mathcal{K}_{form}$ are formulas of polynomial size*
- *$\mathcal{K}_{wskew}$ are weakly skew circuits of polynomial size*
- *$\mathcal{K}_{poly}$ are circuits of polynomial degree and polynomial size*
- *$\mathcal{K}_{exp}$ are circuits of polynomial degree and exponential ($= 2^{n^{O(1)}}$) size*

Throughout this paper, we will be interested in uniform versions of the classes above. Unless mentioned otherwise, all the arithmetic circuit families we mention are DLOGTIME-uniform.

Proposition 1. *We prove the following for the aforementioned circuit classes:*

1. $\mathsf{C}_=\mathcal{K}_{form} = \mathsf{C}_=\mathsf{NC}^1$
2. $\mathsf{C}_=\mathcal{K}_{wskew} = \mathsf{C}_=\mathsf{L}$
3. $\mathsf{C}_=\mathcal{K}_{poly} = \mathsf{C}_=\mathsf{SAC}^1$
4. $\mathsf{C}_=\mathcal{K}_{exp} = \mathsf{C}_=\mathsf{P}$

Proof. Some of the above follow from well-known results:

1. Follows from the simulation of formulas by arithmetic straight-line programs due to Ben-Or and Cleve [5].
2. Follows from Toda's characterization of determinant using weakly skew circuits[24].
3. Follows from Venkateswaran's uniform circuit characterization of SAC^1 as consisting of languages recognized by circuit families having polynomial degree and polynomial size[27,26].
4. Follows from Venkateswaran's uniform circuit characterization of NP as consisting of languages recognized by circuit families having polynomial degree and exponential size.[27]

□

Notice that skew circuits which yield the usual definition of GapL are not closed under composition with formulas (because formulas are not necessarily skew circuits). Thus we have to use the alternative(and equivalent) definition of GapL in terms of weakly skew circuits and in this case all formulas are indeed weakly skew. One of the key requirements in our collapse is the ability to compose two different circuit families.

Definition 8. *Given an arithmetic class $\mathcal{K}$, consider the following:*

(i) $\mathcal{K}_{form} \subseteq \mathcal{K}$
(ii) $\mathcal{K} \subseteq \mathcal{K}_{exp}$.
(iii) $\mathcal{K} \circ \mathcal{K} \subseteq \mathcal{K}$
(iv) $\mathcal{K}^{(l)} \subseteq \mathcal{K}$.

We call $\mathcal{K}$, AC^0-well-behaved if it satisfies $(i), (ii), (iii)$ and NC^1-well-behaved if it satisfies $(i), (ii), (iv)$.

It is easy to see that every NC^1-well-behaved class of circuits $\mathcal{K}$ is also AC^0-well-behaved. Theorem 1 below already entails a collapse of $\mathsf{AC}^0(\mathsf{C}_=\mathcal{K})$ by the observation that NC^1-well-behaved implies AC^0-well-behaved. We show that if one is interested in the collapse of $\mathsf{AC}^0(\mathsf{C}_=\mathcal{K})$ then it is sufficient for $\mathcal{K}$ to be AC^0-well-behaved, which is a weaker notion.

Our main goal in this paper is to study the AC^0 and NC^1 reducibilities to $\mathsf{C}_=\mathcal{K}$. $\mathsf{AC}^0(\mathsf{C}_=\mathcal{K})$(respectively $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$) is the class consisting of languages which can be decided by constant depth(respectively $O(\log n)$-depth), polynomial size circuits consisting of $\wedge$, $\vee$, $\neg$ and $\mathsf{C}_=\mathcal{K}$ gates. Our main theorem is the following:

Theorem 1. *For every NC^1-well-behaved class $\mathcal{K}$ the exact hierarchy $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ collapses to $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$.*

Proof. First we recall some terminology from [8]. Let C be a circuit from $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$. Then a blob consists of the subcircuit rooted at some equality gate g in the circuit where the equality gates in the sub-circuit are replaced by new formal variables. In other words it consists of a single oracle gate. A blob chain consists of a root to leaf path in the NC^1 circuit and includes all the equality gates (aka oracle gates) along the path.

We need a generalization of a lemma from [8] which shows that a circuit of two blob layers i.e. a blob with its input blobs can be replaced by a shallow Boolean circuit (AC^0_3) with a single layer of blobs below it. In this "flattening" process the degree and the size of the circuit increase by an extent we explicate in the lemma.

In [8] this lemma was proved in the context of $\mathsf{C}_{=}\mathsf{NC}^1$ but can be generalized to any $\mathsf{C}_{=}\mathcal{K}$ since the only property of GapNC^1 used in its proof was that it was a class of arithmetic circuits closed under composition with logarithmic depth polynomial size formulas; and this is true for all well-behaved $\mathcal{K}$ by definition.

Lemma 1. *(Rephrased from [8]) Let $C_0(\boldsymbol{x}), C_1(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y})$ be $m+1$ circuits from $\mathsf{C}_{=}\mathcal{K}$. Let $f : \boldsymbol{y} \mapsto C_0(\boldsymbol{C}(\boldsymbol{y}))$ be the function computed by feeding in the circuits C_i for $i > 0$ as inputs of C_0. Then, f is also computed by the following $\mathsf{AC}^0_3(\mathsf{C}_{=}\mathcal{K})$ circuit:*

$$\bigvee_{i=1}^{m} z_i(\boldsymbol{y}) = \bigvee_{i=1}^{m} [S_{i,0}(\boldsymbol{C'}(\boldsymbol{y})) \wedge S_{i,1}(\boldsymbol{C'}(\boldsymbol{y})) \wedge B_i(\boldsymbol{C'}(\boldsymbol{y}))]$$

Notice that, we have used abbreviations, $\boldsymbol{x} = (x_1, \ldots, x_m), \boldsymbol{y} = (y_1, \ldots, y_n)$, and similarly $\boldsymbol{C}, \boldsymbol{C'}$ to simplify notation. More importantly, C'_i are functions in $\mathcal{K}$, $S_{i,0}$ is a $\mathsf{C}_{=}\mathcal{K}$ circuit, $S_{i,1}$ is a $\mathsf{C}_{\neq}\mathcal{K}$ circuit, and B_i from $\mathsf{C}_{\neq}\mathcal{K}$. Further, the size and the degree of the entire circuit is bounded by $O(ms_0) + O(m^8 s)$ and $O(m^9 d + d_0 d)$ respectively. Here s_0 is the size of C_0, s is the sum of sizes of the C_i's (for $i > 0$), d_0 is the degree of C_0 and d is the sum of degrees of C_i's for $i > 0$.

Emulating [8] we can use Lemma 1 to collapse two levels of blobs into one with an AC^0_3 circuit at the top. The AC^0_3 circuit can be converted to an element of $\mathcal{K}_{form}$ (the proof of the lemma shows that the AC^0_3 circuit is an *unambiguous* formula and straightaway arithmetization will preserve the value). Assuming we start with a $\mathsf{AC}^0_k(\mathsf{C}_{=}\mathcal{K})$ circuit, then the closure of $\mathcal{K}$ under composition with $\mathcal{K}_{form}$ allows us to convert the circuit to the form $\mathsf{AC}^0_{k-1}(\mathsf{C}_{=}\mathcal{K})$ i.e. with depth one lesser than earlier. Since we need to repeat this operation k times the size remains bounded by $n^{O(k)}$ times the size of the original circuit and the degree by the degree of the original circuit raised to a power which is $O(k)$. Notice that this seems to work only for AC^0 relativizations because the naïve upper bounds on the size and degree become quasipolynomial when we repeat the collapse logarithmically many times as necessitated by NC^1 relativizations. We show in Lemma 3 that this not the case.

Proof. (of Lemma 1)

Our proof is a quantitative version of the one used in [8]. There, given an $\mathsf{AC}^0(\mathsf{C}_{=}\mathcal{K})$ circuit, one applies Lemma 1 $O(k)$ times, where k is the depth of the $\mathsf{AC}^0(\mathsf{C}_{=}\mathcal{K})$ circuit, to get the required $\mathsf{AC}^0_3(\mathsf{C}_{=}\mathcal{K})$ circuit.

Notice that the AC^0_3 circuit defined by:

$$\bigvee_{i=1}^{m} z_i(\boldsymbol{y}) = \bigvee_{i=1}^{m} [S_{i,0}(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y})) \wedge S_{i,1}(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y})) \wedge B_i(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))]$$

implements the predicate that asserts that there is a subset of circuits from $C_1(\boldsymbol{y}), \ldots C_m(\boldsymbol{y})$ that all evaluate to 1 on $\boldsymbol{y}$. To this end, one implements the following three clauses $S_{i,0}$, $S_{i,1}$ and B_i which evaluate to 1 simultaneously if and only if there is a subset of circuits $\{C_j(\boldsymbol{y})\}$ of size i, for $i, j \in [m]$ such that on input $\boldsymbol{y}$ all these i circuits evaluate to a 1 and the circuit $B_i(C_1(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y}))$ also evaluates to a 1. We use Tzamaret's construction[25] to implement the functions $S_{i,0}$ and $S_{i,1}$:

Let $Y = \{C'_1(\boldsymbol{y}), C'_2(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y})\}$. Consider the polynomial $R_i(Y) = \sum_{S \subseteq Y; |S|=i} \prod_{j \in S} C'_j$ where C'_j come from the set Y consisting of m variables (We assume $R_0(Y) = 1$). Notice that the product of polynomials, $\prod_{i=1}^{l}(x_i + z) = \sum_{j=0}^{m} R_i(Y) z^j$. We can recover the polynomials $R_i(Y)$, given values to the variables in Y by evaluating the polynomials in z at $(m+1)$ distinct values of z, via interpolation.

$$V \times \begin{bmatrix} R_0(Y) \\ R_1(Y) \\ \vdots \\ R_m(Y) \end{bmatrix} = \begin{bmatrix} \prod_{i=1}^{m} C'_i(\boldsymbol{y}) \\ \prod_{i=1}^{m}(C'_i(\boldsymbol{y}) + 1) \\ \vdots \\ \prod_{i=1}^{m}(C'_i(\boldsymbol{y}) + m) \end{bmatrix}$$

where V is a $(m+1) \times (m+1)$ Vandermonde matrix(and hence invertible) where for $0 \leq i, j \leq m, V_{ij} = i^j$. Note that V can be precomputed, since it is a constant. Now we can compute the symmetric polynomial R_i as follows:

$$R_i(C'_1, \ldots, C'_m) = \sum_{j=0}^{m} V^{-1}_{ij} \prod_{k=1}^{m}(C'_k(\boldsymbol{y}) + j)$$

Notice that, here $S_{i,0}$(respectively $S_{i,1}$) are $C_{=}\mathcal{K}$(respectively $C_{\neq}\mathcal{K}$) functions respectively where the underlying $\mathcal{K}$ functions are respectively $R_i(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))$ and $R_{i+1}(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))$. The third clause $B_i(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y})$ is actually emulating our original circuit whereas the other two clauses are essentially used to eliminate the $=$ tests. Also, the AC^0_3 circuit is actually a formula which makes the $\mathsf{AC}^0_3(\mathsf{C}_{=}\mathcal{K})$ circuit unambiguous and hence one can use the fact that $\mathcal{K}_{form} \subseteq \mathcal{K}$ to infer that $\mathcal{K} \circ \mathcal{K}_{form} \subseteq \mathcal{K}$. We further claim that the predicate $\bigvee_{i=1}^{m} z_i(\boldsymbol{y})$ above is true for precisely one value of $i \in [m]$.

Claim. For all $\boldsymbol{y} \in \{0,1\}^n$,

$$C_0(C_1(\boldsymbol{y}), C_2(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y})) \neq 0 \iff \exists i \in [m] : z_i(\boldsymbol{y})$$

Proof. For a given $\boldsymbol{y}$, let $r_{\boldsymbol{y}}$ denote the number of circuits C'_i that evaluate to a non-zero value at $\boldsymbol{y}$. Then,

1. $S_{i,0}(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))$ is 0 when $i > r_{\boldsymbol{y}}$ because the symmetric polynomial on m variables, where each monomial has exactly i variables, is always 0 when $i > r_{\boldsymbol{y}}$, since at least one of the variables in the product is 0.
2. $S_{i,1}(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))$ is 0 exactly when $i < r_{\boldsymbol{y}}$, because of the same reason as above – the monomials are made up from only the non-zero variables, and hence the polynomial is always non-zero.
3. Hence, $z_i(\boldsymbol{y})$ is false whenever $i \neq r_{\boldsymbol{y}}$.

□

So, there is exactly one choice of i, equal to the number of C_is which are non-zero, where the first two clauses of the predicate evaluate to true for all strings $\boldsymbol{y}$. What happens to the third clause B_i at this particular i? One wants this clause to evaluate to a 0 whenever $C_0(C_1(\boldsymbol{y}), C_2(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y}))$ is 0 and otherwise evaluate to 1.

Let $G(\boldsymbol{y}) = \prod_{C'_j(\boldsymbol{y})\neq 0} C'_j(\boldsymbol{y})$. If $\forall i \in [m], C'_i(\boldsymbol{y}) = 0$, define $G(\boldsymbol{y}) = 1$. Hence, $G(\boldsymbol{y})$ is always non-zero by our definition. Plug in $G(\boldsymbol{y})$ instead of 1 and $-G(\boldsymbol{y})$ instead of -1 in $C_0, \ldots, C_m$, and weigh each of the C_is by $G(\boldsymbol{y})$ to eliminate the $=$ gate. This results in a circuit of the form $C_0(C'_1, \ldots, C'_m)$ where $C'_1, \ldots, C'_m$ are functions from the class $\mathcal{K}$. Since the initial $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ was of polynomial degree, the predicate implemented blows up the output of the circuit by a number which is at most polynomially many bits long. Since $B_i(C'_1(\boldsymbol{y}), \ldots, C'_m(\boldsymbol{y}))$ is exactly the function $C_0(C_1(\boldsymbol{y}), C_2(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y}))$, the claim follows. □

One is left with the task of constructing functions in class $\mathcal{K}$ which can implement the functions $S_{i,0}$ and $S_{i,1}$.

Claim. For all $i \in [m]$, there exist $\mathcal{K}$ functions N_i and D_i such that, $S_{i,0}(C_1(\boldsymbol{y}), C_2(\boldsymbol{y}), \ldots, C_m(\boldsymbol{y})) = N^i_m(\boldsymbol{y})/D^i_m(\boldsymbol{y})$.

Proof. The symmetric functions are constructed using the expression:

$$R_i(C'_1, \ldots, C'_m) = \sum_{j=0}^{m} V^{-1}_{ij} \prod_{k=1}^{m} (C'_k(\boldsymbol{y}) + j)$$

The implementation of $S_{i,0}$ and $S_{i,1}$ via Tzamaret's technique requires constructing and inverting a $(m \times m)$-Vandermonde matrix, which requires size $O(m^4)$ from the analysis in Section 2. Since V is an integer matrix, each of the entries of V^{-1} can be written as ratio of integers(the denominator being $det(V) = \prod_{i<j}(j-i)$). So, to compute $R_i(C'_1, \ldots, C'_m)$, we have $D_i(\boldsymbol{y}) = det(V)$ and $N_i(\boldsymbol{y}) = \sum_{j=0}^{m} V^{-1}_{ij} \prod_{i=1}^{m}(C_i(\boldsymbol{y}) + j)$. We can multiply the whole equation by $det(V)$ and this is equivalent to computing $det(V)N_i(\boldsymbol{y}) = \sum_{j=0}^{m} det(V)V^{-1}_{ij} \prod_{i=1}^{m}(C_i(\boldsymbol{y}) + j)$. □

Naïvely, it would seem like the determinants of the Vandermonde matrices get multiplied, at each level, leading to an explosion in the degree. But, notice that by Fact 1, at every level, the denomianator will be just $max_{i\in[m]} det(V_i)$, where V_i,the Vandermonde matrix of dimension $(i \times i)$, is the denominator that percolates up as a result of previous collapses. This will remove the need to carry around the symmetric function as a numerator-denominator pair. Since we replace 1 with $G(\boldsymbol{y})$ and -1 with $-G(\boldsymbol{y})$ in our circuit B_i, in order to bound the size and the degree, one has to make sure that the size and degree of the expression $det(V)N_i(\boldsymbol{y})$ is bounded by a polynomial in the number of input variables. The size of the circuit resulting from the collapse is $O(ms_0 + m^8 s)$: $S_{i,0}$ and $S_{i,1}$ contribute $O(m^8 s)$ each for every value of i, and considering the fact that the boolean OR is over m values of i, their total contribution to the size is $O(m^9 s)$. The clause B_i contributes $O(ms_0 + ms)$: the circuit at the top, C_0 contributes s_0 for each AND gate(totally $O(ms_0)$, and its inputs are of size s, adding up to $O(ms)$ size. Hence the total size of the circuit resulting from a single collapse is $O(m^8 s + ms_0)$. In case of the degree, one really has to bound $\max_{i\in[m]}$ degree$[z_i(\boldsymbol{y})]$. The degree of the circuit is at most $O(m^8 \log^2 md)$(this is in order to account for V^{-1} which is written as a ratio of determinant of cofactor matrices and $det(V)$). Hence the degree of the whole circuit obtained as a result of a single collapse is at most (degree of $S_{i,0}$ + degree of $S_{i,1}$ +

degree of $B_i) \leq O(m^8 \log^2 md) + O(m^8 \log^2 md) + O(d_0 d) \leq O(m^8 \log^2 md + d_0 d)$, where d_0 is the degree of C_0 and d is the sum of degrees of C_is. Thus we get the size and degree bounded as claimed in Lemma 1. □

Now we show how to do a careful analysis of the collapse in case of NC^1 relativizations which allows us to conclude that the size and degree do not become too large. The key is to use basic counting along with Cook-Wilson condition on NC^1-relativization which allows us to conclude the following:

Proposition 2. *Consider an $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ circuit C with size, height and number of inputs $s(C), h(C)$ and $n(C)$ respectively. Then,*

- *the number of distinct maximal blob chains is upper bounded by $s(C)$.*
- *the product of the fan-ins of the oracle gates along a blob-chain is upper bounded by $(n(C))^{c_1}$ for some constant c_1 depending on the circuit family;(a consequence of the Cook-Wilson property)*

Next we bound the total size of the flattened circuit. Lemma 2 shows that, for the $\mathcal{K}^{(l)}$ circuits obtained as a result of the collapse, indeed $\mathcal{K}^{(l)} = \mathcal{K}$.

Lemma 2. *The size of the flattened circuit is upper bounded by: $n^c s^2$, where s is the sum of sizes of the circuits corresponding to oracle gates and c is a constant depending on the family of circuits (and n is the number of inputs to the original circuit).*

Proof. We use the recurrence:

$$S(g) = 3f_g^9 \sum_{h:\text{parent}(h)=g} S(h) + s_g f_g \tag{1}$$

$$\leq 3f_g^9 \sum_{h:\text{parent}(h)=g} S(h) + s_g^2 \text{ (Since } f_g \leq s_g) \tag{2}$$

Here $S(g)$ is the total size of the sub-circuit below gate g after flattening, s_g is the original size of the $\mathcal{K}$-circuit for just the gate g and f_g is the fan-in of g. Let us further denote by h_g the height of g. Then the recurrence has the solution:

$$S(g) = 3^{h_g} \sum_{g':g' \text{ is a descendant of } g} \left(\left(\prod_{h \in B(g,g')} f_h^9 \right) s_{g'}^2 \right)$$

where $B(g, g')$ denotes the blob-chain between g and g'.

Let g_0 be the root of the circuit. From Proposition 2, the products of the fan-ins in the sum above is upper bounded by n^{c_1}, where $n = n(C)$ is the number of inputs of the circuit. Also, h_{g_0} is bounded by $c_2' \log n$ for some constant c_2', since the reduction is an NC^1 reduction. Hence, $S(g_0) \leq n^c \sum_g {s_g}^2$. □

Finally we bound the degree of the flattened circuit, which is crucial to proving our collapse:

Lemma 3. *The degree of the flattened circuit is bounded by $n^{c'}$ where c' is a constant depending on the original circuit family.*

Proof. The degree of a well-behaved circuit on t inputs is at most $t^{c'_1}$ for some constant c'_1 depending on the circuit family (since a circuit family is a subclass of $\mathcal{K}_{exp}$ circuits which have a polynomial degree. Now, we have the recurrence:

$$D(g) = f_g^9 d_g \sum_{h:\text{parent}(h)=g} D(h)$$

where we define $D(g)$ to be the degree of the flattened circuit at g and d_g its original degree as a circuit in $\mathcal{K}$. This has the solution:

$$D(g) = \sum_{B:B \text{ is a maximal blob-chain rooted at } g} \left(\prod_{h:h\in B} (f_h^9 f_h^{c'_1}) \right)$$

Thus, $D(g_0) \leq s(C) n^{(c'_1+1)c_1} \leq n^{c'}$, using the bounds from Proposition 2 for the first inequality and observing that the fan-in of any oracle gate and the size of the NC^1 circuit is bounded by n. □

Hence we have proved that for each equality gate in the final circuit, the degree remains polynomial via composition with formulas which means the circuit rooted at the equality gate is in $\mathsf{C}_=\mathcal{K}$, and the overall circuit in $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$. This completes the proof of the theorem.

We now document some consequences of Theorem 1:

Corollary 1. $\mathsf{NC}^1(\mathsf{C}_=\mathsf{NC}^1) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{NC}^1)$

Proof. Formulas are AC^0-well-behaved since they are clearly closed under composition with formulas and have polynomial degree. The fact that they are NC^1-well-behaved follows from Definition 5, namely the arithmetic Cook-Wilson property. □

Corollary 2. $\mathsf{NC}^1(\mathsf{C}_=\mathsf{L}) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{L})$

Proof. Weakly skew circuits are closed under composition with $\mathcal{K}_{form}$ since $\mathcal{K}_{form}$ circuits are weakly skew(by [24]). Also, the polynomials computable by weakly skew circuits of polynomial size are also computable by determinants of polynomial sized matrices by [24]. Hence the former have polynomial degrees. Thus weakly skew circuits are AC^0-well-behaved. To prove that weakly skew circuits are NC^1-well-behaved, it is sufficient to note that, weakly skew circuits composed with themselves $O(\log n)$-many times, remain weakly skew due to the arithmetic Cook-Wilson in Definition 5. □

Corollary 3. $\mathsf{NC}^1(\mathsf{C}_=\mathsf{SAC}^1) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{SAC}^1)$

Proof. Formulas of polynomial size have polynomial degree, thus polynomial degree circuits of polynomial size are closed under composition with them completing the proof that they are AC^0-well-behaved. Similarly, polynomial degree circuits composed with themselves $O(\log n)$-many times still have polynomial formal degree, and hence are NC^1-well-behaved. □

Corollary 4. $\mathsf{NC}^1(\mathsf{C}_=\mathsf{P}) = \mathsf{AC}^0_3(\mathsf{C}_=\mathsf{P})$

Proof. This also follows exactly from the same reasoning as the one for $\mathsf{C}_=\mathsf{SAC}^1$ circuits. □

4 Uniformity

We have proved that $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ collapses to $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$. In this section we show that such an $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$ family is DLOGTIME-uniform, when the original $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ circuit family is DLOGTIME-uniform.

Following [29], we call our AC^0_3 circuit uniform, if the following language can be decided in $O(\log n)$ time by a deterministic turing machine (random access to its tape) : $L_{DC} = \{\langle y, g, p, b\rangle\}$, where

- $|y| = n$
- g is the number of a gate v in C_n
- $p \in \{0,1\}^*$ such that if $p = \epsilon$, then b encodes the type of the gate from the basis of the circuit family and if p is the binary representation of k, then b is the number of the k-th predecessor gate to v wrt to a fixed ordering on the gates in the circuit.

We encode every gate g in the given circuit $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ C by a concatenation of two labels (g_e, g_i). g_e is an external encoding which assigns labels lexicographically to every $\mathsf{C}_=\mathcal{K}$ gate in C. g_i is an internal encoding which assigns labels to the $+, \times, =$ gates which constitute the $\mathsf{C}_=\mathcal{K}$ gates. More precisely, given $\langle y, g, p, b\rangle$, we have $y = 1^n$, $g = (g_e, g_i)$. g_e is a $O(\log n)$ length string (since the underlying circuit is a NC^1 circuit the total number of bits needed to uniquely address a gate in the circuit is $O(\log n) + \max_\pi \sum_{g\in\pi} \log \text{fanin}(g) = O(\log n)$, where the maximum is taken over all paths π in the circuit) by the definition of oracle circuits in the Cook-Wilson model). g_i is of length $O(\log n)$(since each of the $\mathsf{C}_=\mathcal{K}$ gates is poly-sized)[3].

We will analyze a typical scenario in our collapse - Given a circuit of the form $C = C_1 \circ C_2 \circ C_3$ (see Figure 1A), where C_1, C_2 and C_3 are $\mathsf{C}_=\mathcal{K}$ circuits, we collapse C_2 and C_3 to obtain a circuit $C' = C_1 \circ C'_2 \circ C'_3$ (see Figure 1B). Here C'_2 is a circuit of the form $\vee \circ \wedge$ and C'_3 is a layer of $\mathsf{C}_=\mathcal{K}$ circuits. We prove that the collapse of C to C' can be made DLOGTIME-uniform and then show that under the Cook-Wilson property, such collapses can be combined to make the collapse of $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$, DLOGTIME-uniform.

Claim. For every NC^1-well-behaved class $\mathcal{K}$, DLOGTIME-uniform $\mathsf{C}_=\mathcal{K} \circ \mathsf{C}_=\mathcal{K} \circ \mathsf{C}_=\mathcal{K}$ = DLOGTIME-uniform $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$.

Proof. Assuming the DLOGTIME-uniformity and the aforementioned labelling scheme of C, we will now see how to decide the connection language of C' when we have access to the connection language of C. First we explain how we label C' from C: The

[3] Note that from [27] $\mathsf{C}_=\mathsf{P}$ is exactly characterized by DLOGTIME-uniform semi-unbounded circuits of $2^{n^{O(1)}}$ size and logarithmic depth and in this case the labels of the gate themselves will be strings of length $n^{O(1)}$

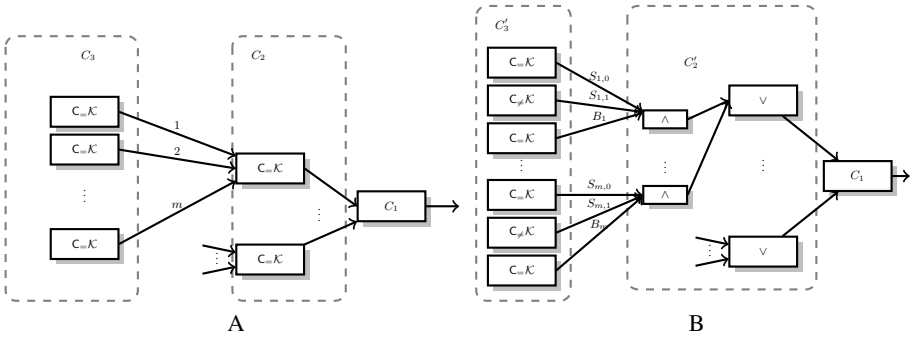

Fig. 1: (A) A circuit of the form $C_1 \circ C_2 \circ C_3 = \mathsf{C}_{=}\mathcal{K} \circ \mathsf{C}_{=}\mathcal{K} \circ \mathsf{C}_{=}\mathcal{K}$ (B) The collapsed circuit obtained from the circuit $C_1 \circ C_2 \circ C_3$ above of the form $C_1 \circ C_2' \circ C_3' = \mathsf{C}_{=}\mathcal{K} \circ \vee \circ \wedge \circ C_{=}\mathcal{K}$.

labels of C_1 carry over from C to C'. For the $\vee \circ \wedge$ circuit C_2' we give new set of external and internal labels similar to C. This takes $O(\log n)$ many bits. There are two kinds of circuits in C_3': circuits computing the symmetric functions, which are built via interpolation by symmetric polynomials, using the Vandermonde matrix and circuits of the form $\mathsf{C}_{=}\mathcal{K} \circ \mathcal{K}$ (see Lemma 1). From Fact 1, it is clear that the entries of this matrix can be computed by circuits of polynomial size, which are labelled using $O(\log n)$ bits. The circuit of the form $\mathsf{C}_{=}\mathcal{K} \circ \mathcal{K}$ in C_3' is essentially C_2 to which the inputs are symmetric functions of C_3. Even here we assign a concatenation of external and internal encoding of the circuits from C_2 and C_3 respectively, which are $O(\log n)$ bits long.

In each of C_1, C_2', C_3', we identify three kinds of gates which we call $\alpha_0, \alpha_{con}, \alpha_{mid}$. α_0 gates are either the output gate or the input gates of C'. α_{con} are gates which connect C_i to C_{i+1} for $i \in \{1, 2\}$. For example, the root gate of C_1, the $\vee$ gate in C_2' and the $=$ gate in C_3' are α_{con} type gates - their parents and children are not of type α_{con}. α_{mid} gates are those whose parents and children are both α_{mid} or α_{con}. The internal gates of C_1 and C_3 are gates of type α_{mid}. Note that all the gates in C' belong to one of these three categories.

So now we are ready to describe the label of a gate in C'. It is of the form $\langle y, g, p, b \rangle$ as above where g is now the concatenation of the external labels of the path to the nearest $=$ ancestor to the gate and the internal label of the gate in C. For example, the label of a gate in C_3' would consist of the concatenation of external labels of C_1, C_2', and the internal label of the gate in C_3'. If p is ϵ, then b is the type of the gate, which could be $\wedge, \vee, \mathsf{C}_{=}\mathcal{K}$. Else the binary number encoded by the string p points to the position of the gate according to our fixed ordering. Note that at the end of one collapse, the label lengths increase by atmost $O(\log n)$ bits(namely the bits required to internally label the $\vee \circ \wedge$ system arising out of the collapse. With the above labelling convention, now there are two tasks to be accomplished:

1. Given a label of a gate g in C', we have to verify if it is indeed a valid label.
2. Given labels of gates g, h is g a parent/child of h?

The validity of a label is easily checked: One has to check if the concatenation of labels leads to a valid external and internal encoding by querying the DLOGTIME machine for the original circuit C and check if it is a α_0, α_{mid} or α_{con} gate. For example, given two gates, deciding if one is the parent/child of the other can be done by checking if the label of one of them is a prefix of the other, and if yes, verifying if the gate types of the parent and child is valid according to the relation between αs specified above. □

Now we prove that such a labelling convention leads to easily checking the validity of a label and connections in the collapse from Theorem 1.

Claim. For every NC^1-well-behaved class $\mathcal{K}$, DLOGTIME-uniform $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K}) =$ DLOGTIME-uniform $\mathsf{AC}^0_3(\mathsf{C}_=\mathcal{K})$.

Proof. To label the final $\mathsf{AC}^0(\mathsf{C}_=\mathcal{K})$ circuit, we concatenate the labels of the intermediate collapses. This might seem like it requires labels of length $O(\log^2 n)$ since the circuit we started out with had labels of length $O(\log n)$ and the $\mathsf{NC}^1(\mathsf{C}_=\mathcal{K})$ circuit requires $O(\log n)$ collapses to reduce it to an $\mathsf{AC}^0(\mathsf{C}_=\mathcal{K})$ circuit. But just as in the proof of Theorem 1, the Cook-Wilson property ensures that the length of the labels is $O(\log n)$. This is because, even though we do the collapse as many as $O(\log n)$ times, the increase in label lengths is only due to the $\vee \circ \wedge$ block. Recall that the number of $\wedge$ gates created in 1 is exactly equal to the fan-in of the oracle gate above, and the Cook-Wilson relativization model bounds the product of fan-ins along any path to a polynomial in the number of inputs. This also bounds the number of new gates created, and hence they can be labelled by $O(\log n)$-many bits. □

5 Conclusion and Open Problems

We provide sufficient conditions on arithmetic circuits under which the NC^1 hierarchy over the corresponding exact arithmetic class collapses. Natural extensions can include proving similar results for more powerful reducibilities like Boolean Formula reductions or even Logarithmic Boolean Formula reductions (which do not satisfy the "small-blob-chains" property or equivalently do not follow Cook-Wilson relativization). An appropriately defined notion of SAC^1-reductions is another intriguing possibility.

It may also be an interesting idea to give similar uniform proofs for the NC^1 hierarchies over the better known classes $\mathsf{PNC}^1, \mathsf{PL}, \mathsf{PSAC}^1, \mathsf{PP}$ where we already know collapses for the second[17] and the last [4] hierarchies and also of the $\mathsf{AC}^0(\mathsf{PNC}^1)$-hierarchy [8].

Acknowledgements

We thank Eric Allender, Vikraman Arvind, Sourav Chakraborty, Meena Mahajan, Prajakta Nimbhorkar and B.V.Raghavendra Rao for helpful discussions. The second author would like to thank the organisers of the Dagstuhl seminar on "Algebraic and Combinatorial Methods in Computational Complexity" where many of the previously mentioned discussions took place. We thank the anonymous referees for helpful comments.

References

1. Allender, E., Beals, R., Ogihara, M.: The complexity of matrix rank and feasible systems of linear equations. Computational Complexity 8(2), 99–126 (1999)
2. Allender, E., Jiao, J., Mahajan, M., Vinay, V.: Non-commutative arithmetic circuits: depth reduction and size lower bounds. Theoretical Computer Science 209(1), 47–86 (1998)
3. Beigel, R., Reingold, N., Spielman, D.A.: PP is closed under intersection. Journal of Computer and System Sciences 50(2), 191–202 (1995)
4. Beigel, R., Fu, B.: Circuits over pp and pl. J. Comput. Syst. Sci. 60(2), 422–441 (2000)
5. Ben-Or, M., Cleve, R.: Computing algebraic formulas using a constant number of registers. SIAM Journal on Computing 21, 54–58 (1992)
6. Caussinus, H., McKenzie, P., Thérien, D., Vollmer, H.: Nondeterministic NC^1 computation. Journal of Computer and System Sciences 57, 200–212 (1998), preliminary version in Proceedings of the 11th IEEE Conference on Computational Complexity, 1996, 12–21
7. Cook, S.: A taxonomy of problems with fast parallel algorithms. Information and Control 64, 2–22 (1985)
8. Datta, S., Mahajan, M., Rao, B.V.R., Thomas, M., Vollmer, H.: Counting classes and the fine structure between NC^1 and L. In: Proceedings of the 35th international conference on Mathematical foundations of computer science. pp. 306–317. MFCS'10 (2010)
9. Fenner, S.A., Fortnow, L.J., Kurtz, S.A.: Gap-definable counting classes. Journal of Computer and System Sciences 48(1), 116–148 (Feb 1994)
10. Gill, J.: Computational complexity of probabilistic turing machines. SIAM Journal on Computing 6(4), 675–695 (1977)
11. Gottlob, G.: Collapsing oracle-tape hierarchies. In: IEEE Conference on Computational Complexity. pp. 33–42 (1996)
12. Hemachandra, L.: The strong exponential hierarchy collapses. In: Structure in Complexity Theory Conference. IEEE Computer Society (1987)
13. Hoang, T.M., Thierauf, T.: The complexity of the characteristic and the minimal polynomial. Theor. Comput. Sci. 295, 205–222 (2003)
14. Immerman, N.: Nondeterministic space is closed under complementation. SIAM Journal on Computing 17(5), 935–938 (Oct 1988)
15. Limaye, N., Mahajan, M., Rao, B.: Arithmetizing classes around nc 1 and l. STACS 2007 pp. 477–488 (2007)
16. Mengel, S.: Conjunctive Queries, Arithmetic Circuits and Counting Complexity. Ph.D. thesis, Universität Paderborn (2012)
17. Ogihara, M.: The PL hierarchy collapses. SIAM J. Comput. 27(5), 1430–1437 (1998)
18. Ogihara, M.: Equivalence of NC^k and AC^{k-1} closures of NP and Other Classes. Inf. Comput. 120(1), 55–58 (1995)
19. Ogiwara, M.: Generalized theorems on relationships among reducibility notions to certain complexity classes. Mathematical Systems Theory 27(3), 189–200 (1994)
20. Santha, M., Tan, S.: Verifying the determinant in parallel. Computational Complexity 7(2), 128–151 (1998)
21. Schöning, U., Wagner, K.W.: Collapsing oracle hierarchies, census functions and logarithmically many queries. In: STACS. pp. 91–97 (1988)
22. Simon, J.: On some central problems in computational complexity (1975)
23. Szelepcsényi, R.: The method of forced enumeration for nondeterministic automata. Acta Informatica 26(3), 279–284 (1988)
24. Toda, S.: Classes of arithmetic circuits capturing the complexity of computing the determinant. IEICE Transactions on Information and Systems E75-D, 116–124 (1992)

25. Tzamaret, I.: Studies in Algebraic and Propositional Proof Complexity. Ph.D. thesis, Tel Aviv University (2008)
26. Venkateswaran, H.: Properties that characterize LogCFL. Journal of Computer and System Sciences 42, 380–404 (1991)
27. Venkateswaran, H.: Circuit definitions of nondeterministic complexity classes. SIAM J. on Computing 21, 655–670 (1992)
28. Vinay, V.: Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In: Proceedings of 6th Structure in Complexity Theory Conference. pp. 270–284 (1991)
29. Vollmer, H.: Introduction to Circuit Complexity: A Uniform Approach. Springer-Verlag New York Inc. (1999)
30. Wilson, C.B.: Relativized circuit complexity. J. Comput. Syst. Sci. 31(2), 169–181 (1985)

ECCC ISSN 1433-8092

http://eccc.hpi-web.de