Lower Bounds against Weakly Uniform Circuits

Ruiwen Chen and Valentine Kabanets

School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada
ruiwenc@sfu.ca kabanets@cs.sfu.ca

Abstract. A family of Boolean circuits $\{C_n\}_{n\geqslant 0}$ is called $\gamma(n)$-*weakly uniform* if there is a polynomial-time algorithm for deciding the direct-connection language of every C_n, given *advice* of size $\gamma(n)$. This is a relaxation of the usual notion of uniformity, which allows one to interpolate between complete uniformity (when $\gamma(n) = 0$) and complete non-uniformity (when $\gamma(n) > |C_n|$). Weak uniformity is essentially equivalent to *succinctness* introduced by Jansen and Santhanam [12].
Our main result is that PERMANENT is not computable by polynomial-size $n^{o(1)}$-weakly uniform TC^0 circuits. This strengthens the results by Allender [2] (for *uniform* TC^0) and by Jansen and Santhanam [12] (for weakly uniform *arithmetic* circuits of constant depth). Our approach is quite general, and can be used to extend to the "weakly uniform" setting all currently known circuit lower bounds proved for the "uniform" setting. For example, we show that PERMANENT is not computable by polynomial-size $(\log n)^{O(1)}$-weakly uniform threshold circuits of depth $o(\log\log n)$, generalizing the result by Koiran and Perifel [16].

Keywords: advice complexity classes, alternating Turing machines, counting hierarchy, permanent, succinct circuits, threshold circuits, uniform circuit lower bounds, weakly uniform circuits

1 Introduction

Understanding the power and limitation of efficient algorithms is the major goal of complexity theory, with the "P vs. NP" problem being the most famous open question in the area. While proving that no NP-complete problem has a uniform polynomial-time algorithm would suffice for separating P and NP, a considerable amount of effort was put into the more ambitious goal of trying to show that no NP-complete problem can be decided by even a *nonuniform* family of polynomial-size Boolean circuits.

More generally, an important goal in complexity theory has been to prove strong (exponential or super-polynomial) circuit lower bounds for "natural" computational problems that may come from complexity classes larger than NP, e.g., the class NEXP of languages decidable in nondeterministic exponential time. By the counting argument of Shannon [23], a randomly chosen n-variate Boolean function requires circuits of exponential size. However, the best currently known circuit lower bounds for *explicit* problems are only linear for NP problems [17,11], and polynomial for problems in the polynomial-time hierarchy PH [14].

To make progress, researchers introduced various restrictions on the circuit classes. In particular, for Boolean circuits of *constant* depth, with NOT and unbounded fan-in AND and OR gates (AC^0 circuits), exponential lower bounds are known for the PARITY function [8,29,9]. For constant-depth circuits that additionally have (unbounded fan-in) MOD_p gates, one also needs exponential size to compute the MOD_q function, for any distinct primes p and q [20,24]. With little progress for decades, Williams [28] has recently shown that a problem in NEXP is not computable by polynomial-size ACC^0 circuits, which are constant-depth circuits with NOT gates and unbounded fan-in AND, OR and MOD_m gates, for any integer $m > 1$. However, no lower bounds are known for the class TC^0 of constant-depth threshold circuits with unbounded fan-in majority gates.[1]

To make more progress, another restriction has been added: *uniformity* of circuits. Roughly speaking, a circuit family is called uniform if there is an efficient algorithm that can construct any circuit from the family. There are two natural variations of this idea. One can ask for an algorithm that outputs the entire circuit in time polynomial in the circuit size; this notion of uniformity is known as P-uniformity. In the more restricted notion, one asks for an algorithm that describes the local structure of the circuit: given two gate names, such an algorithm determines if one gate is the input to the other gate, as well as determines the types of the gates, in time linear (or polynomial) in the input size (which is logarithmic or polylogarithmic time in the size of the circuit described by the algorithm); such an algorithm is said to decide the *direct-connection language* of the given circuit. This restricted notion is called $\mathsf{DLOGTIME}$- (or $\mathsf{POLYLOGTIME}$-) uniformity [22,5,3]. We will use the notion of $\mathsf{POLYLOGTIME}$-uniformity by default, and, for brevity, will omit the word $\mathsf{POLYLOGTIME}$.

It is easy to show (by diagonalization) that, for any fixed exponential function $s(n) = 2^{n^c}$ for a constant $c \geqslant 1$, there is a language in EXP (deterministic exponential time) that is not computable by a uniform (P-uniform) family of Boolean $s(n)$-size circuits.[2] Similarly, as observed in [2], a PSPACE-complete language requires exponential-size uniform TC^0 circuits. For the smaller complexity class $\#\mathsf{P} \subseteq \mathsf{PSPACE}$, Allender and Gore [3] showed PERMANENT (which is complete for $\#\mathsf{P}$ [26]) is not computable by uniform ACC^0 circuits of subexponential size. Later, Allender [2] proved that PERMANENT cannot be computed by uniform TC^0 circuits of size $s(n)$ for any function s such that, for all k, $s^{(k)}(n) = o(2^n)$ (where $s^{(k)}$ means the function s composed with itself k times). Finally, Koiran and Perifel [16] extended this result to show that PERMANENT is not computed by polynomial-size uniform threshold circuits of depth $o(\log \log n)$.

Recently, Jansen and Santhanam [12] have proposed a natural relaxation of uniformity, termed *succinctness*, which allows one to interpolate between nonuniformity and uniformity. According to [12], a family of $s(n)$-size circuits $\{C_n\}$

[1] A plausible explanation of this "barrier" is given by the "natural proofs" framework of [21], who argue it is hard to prove lower bounds against the circuit classes that are powerful enough to implement cryptography.

[2] Unlike the nonuniform setting, where every n-variate Boolean function is computable by a circuit of size about $2^n/n$ [18], *uniform* circuit lower bounds can be $> 2^n$.

is succinct if the direct-connection language of C_n is decided by some circuit of size $s(n)^{o(1)}$. In other words, while there may not be an efficient algorithm for describing the local structure of a given $s(n)$-size circuit C_n, the local structure of C_n can be described by a *non-uniform* circuit of size $s(n)^{o(1)}$. Note that if we allow the non-uniform circuit to be of size $s(n)$, then the family of circuits $\{C_n\}$ would be completely non-uniform. So, intuitively, the restriction to the size $s(n)^{o(1)}$ makes the notion of succinctness close to that of non-uniformity.

The main result of [12] is that PERMANENT does not have succinct polynomial-size *arithmetic* circuits of constant depth, where arithmetic circuits have unbounded fan-in addition and multiplication gates and operate over integers. While relaxing the notion of uniformity, [12] were only able to prove a lower bound for the *weaker* circuit class, as polynomial-size constant-depth arithmetic circuits can be simulated by polynomial-size TC^0 circuits. A natural next step was to prove a super-polynomial lower bound for PERMANENT against succinct TC^0 circuits. This is achieved in the present paper.

1.1 Our main results

We improve upon [12] by showing that PERMANENT does not have succinct polynomial-size TC^0 circuits. In addition to strengthening the main result from [12], we also give a simpler proof. Our argument is quite general and allows us to extend to the "succinct" setting all previously known uniform circuit lower bounds of [3,2,16].

Recall that the direct-connection language for a circuit describes the local structure of the circuit; more precise definitions will be given in the next section. For a function $\alpha : \mathbb{N} \to \mathbb{N}$, we say that a circuit family $\{C_n\}$ of size $s(n)$ is *α-weakly uniform* if the direct-connection language L_{dc} of $\{C_n\}$ is decided by a polynomial-time algorithm that, in addition to the input of L_{dc} of size $m \in O(\log s(n))$, has an advice string of size $\alpha(m)$; the advice string just depends on the input size m. The notion of α-weakly uniform is essentially equivalent to the notion of α-succinct introduced in [12]; see the next section for details.

We will call a circuit family subexp-*weakly uniform* if it is α-weakly uniform for $\alpha(m) \in 2^{o(m)}$. Similarly, we call a circuit family poly-*weakly uniform* if it is α-weakly uniform for $\alpha(m) \in m^{O(1)}$. Observe that for $m = O(\log s)$, we have $2^{o(m)} = s^{o(1)}$ and $m^{O(1)} = \mathsf{poly} \log s$.

Our main results are as below. First, we strengthen the lower bound of [12].

Theorem 1. PERMANENT *is not computable by* subexp-*weakly uniform* poly-*size* TC^0 *circuits.*

Let us call a function $s(n)$ *sub-subexponential* if, for any constant $k > 0$, we have that the k-wise composition $s^{(k)}(n) \leqslant 2^{n^{o(1)}}$. We use $\mathsf{subsubexp}$ to denote the class of all sub-subexponential functions $s(n)$. We extend a result of Allender [2] to the "weakly-uniform" setting.

Theorem 2. PERMANENT *is not computable by* poly-*weakly uniform* $\mathsf{subsubexp}$-*size* TC^0 *circuits.*

Finally, we extend the result of [16].

Theorem 3. PERMANENT *is not computable by* poly*-weakly uniform* poly*-size threshold circuits of depth* $o(\log \log n)$.

1.2 Our techniques

At the high level, we use the method of *indirect diagonalization*:

- assuming PERMANENT is easy and using diagonalization, we first show the existence of a "hard" language in a certain complexity class $\mathcal{C}$ (the counting hierarchy, to be defined below);
- assuming PERMANENT is easy, we show that the above "hard" language is actually "easy" (as the easiness of PERMANENT collapses the counting hierarchy), which is a contradiction.

In more detail, we first extend the well-known correspondence between uniform TC^0 and alternating polylog-time Turing machines (that use majority states) to the weakly uniform setting, by considering alternating Turing machines with *advice*. To construct the desired "hard" language, we use diagonalization against such machines with advice. The assumed easiness of PERMANENT is used to argue two things about the constructed "hard" language L_{hard}:

1. L_{hard} is in fact "hard" for a much more powerful class $\mathcal{A}$ of algorithms;
2. L_{hard} is decided by a "simple" algorithm A.

The contradiction ensues since algorithm A turns out to be from the class $\mathcal{A}$.

1.3 Relation to the previous work

A similar indirect-diagonalization strategy was used (explicitly or implicitly) in all previous papers showing uniform or weakly uniform circuit lower bounds for PERMANENT [3,2,16,12]. Our approach is most closely related to that of [2,16]. The main difference is that we work in the weakly uniform setting, which means that we need to handle a certain amount of non-uniform advice. To that end, we have adapted the method of indirect diagonalization, making it modular (as outlined above) and sufficiently general to work also in the setting with advice. Due to this generality of our proof argument, we are able to extend the aforementioned lower bounds from the uniform setting to the weakly uniform setting.

The approach adopted by [12] goes via the well-known connection between derandomization and circuit lower bounds (cf. [10,13,1]). Since the authors of [12] work with the algebraic problem of Polynomial Identity Testing (given an arithmetic circuit computing some polynomial over integers, decide if the polynomial is identically zero), their final lower bounds are also in the algebraic setting: for weakly uniform arithmetic constant-depth circuits. By making the diagonalization arguments in [12] more explicit (along the lines of [2]), we are able to get the lower bound for weakly uniform Boolean (TC^0) circuits, thereby both strengthening the results and simplifying the proofs from [12].

2 Preliminaries

We refer to [4] for the basic complexity notions.

2.1 Weakly uniform circuit families

Following [22,3], we define the *direct connection language* of a circuit family $\{C_n\}$ as $L_{dc} = \{(n, g, h)\colon g = h$ and g is a gate in C_n, or $g \neq h$ and h is an input to $g\}$, where n is in binary representation, and g and h are binary strings encoding the gate types and names. The *type* of a gate could be constant 0 or 1, Boolean logic gate NOT, AND, or OR, majority gate MAJ, modulo gate MOD_m for some integer m, or input $x_1, x_2, \ldots, x_n$. For a circuit family of size $s(n)$, we need $c_0 \log s(n)$ bits to encode (n, g, h), where c_0 is a small constant at most 4.

A circuit family $\{C_n\}$ is *uniform* [5,3] if its direct connection language is decidable in time polynomial in its input length $|(n, g, h)|$; this was referred to as $\mathsf{POLYLOGTIME}$-uniformity in [3].

We say a function $f(n)$ is *constructible* if there is a deterministic TM that computes $f(n)$ in binary in time $O(f(n))$, when given n in binary as the input[3].

Following [12], for a constructible function $\alpha : \mathbb{N} \to \mathbb{N}$, we say that a circuit family $\{C_n\}$ of size $s(n)$ is *α-succinct* if its direct connection language L_{dc} is in $\mathsf{SIZE}(\alpha)$; i.e., L_{dc} has (non-uniform) Boolean circuits of size $\alpha(m)$, where $m = c_0 \log s(n)$ is the input size for L_{dc}. Trivially, for $\alpha(m) \geqslant 2^m$, every circuit family is α-succinct. The notion becomes nontrivial when $\alpha(m) \ll 2^m/m$. We will use $\alpha(m) = 2^{o(m)}$ (slightly succinct) and $\alpha(m) = m^{O(1)}$ (highly succinct).

We recall the definition of Turing machines with advice from [15]. Given functions $t\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ and $\alpha\colon \mathbb{N} \to \mathbb{N}$, we say that a language L is in $\mathsf{DTIME}(t)/\alpha$, if there is a deterministic Turing machine M and a sequence of advice strings $\{a_n\}$ of length $\alpha(n)$ such that, for any $x \in \{0,1\}^n$, machine M on inputs (x, a_n) decides whether $x \in L$ in time $t(n, \alpha(n))$. If the function $t(n, m)$ is upper-bounded by a polynomial in $n + m$, we say that $L \in \mathsf{P}/\alpha$.

Definition 1. *A circuit family $\{C_n\}$ of size $s(n)$ is* α-weakly uniform *if its direct connection language is decided in P/α; recall that the input size for the direct-connection language describing C_n is $m = c_0 \log s(n)$, and so the size of the advice string needed in this case is $\alpha(c_0 \log s(n))$.*

The two notions are closely related.

Lemma 1. *In the notation above, $\alpha(m)$-succinctness implies $\alpha(m) \log \alpha(m)$-weak uniformity, and conversely, $\alpha(m)$-weak uniformity implies $(\alpha(m)+m)^{O(1)}$-succinctness.*

Proof (sketch). A Boolean circuit of size s can be represented by a binary string of size $O(s \log s)$; and a Turing machine running in time t can be simulated by a circuit family of size $O(t \log t)$. □

[3] We note that $f(n)$ is constructible in our sense if and only if $2^{f(n)}$ is constructible according to Allender's definition in [2].

The notion of weak uniformity (succinctness) interpolates between full uniformity on one end and full non-uniformity on the other end. For example, 0-weak uniformity is the same as uniformity. On the other hand, α-weak uniformity for $\alpha(m) \geqslant 2^m$ is the same as non-uniformity. For that reason, we will assume that the function α in "α-weakly uniform" is such that $0 \leqslant \alpha(m) \leqslant 2^m$.

Definition 2. *We say a circuit family $\{C_n\}$ is* subexp-weakly uniform *if it is α-weakly uniform for $\alpha(m) \in 2^{o(m)}$; similarly, we say $\{C_n\}$ is* poly-weakly uniform *if it is α-weakly uniform for $\alpha(m) \in m^{O(1)}$.*

2.2 Weak uniformity vs. alternating Turing machines with advice

Following [7,19,3], a *threshold Turing machine* is an alternating TM (ATM) with majority (MAJ) states; a configuration in *majority state* may have an unbounded number of successors, and it is accepting iff more than half of its successors are accepting. We denote by $\mathsf{Th}_{d(n)}\mathsf{TIME}(t(n))$ the class of languages accepted by threshold TMs having at most $d(n)$ alternations and running in time $O(t(n))$.

The *counting hierarchy* [27,25] is defined as $\mathsf{CH} = \cup_{d \geqslant 0}\mathsf{CH}_d$, where $\mathsf{CH}_0 = \mathsf{P}$ and $\mathsf{CH}_{d+1} = \mathsf{PP}^{\mathsf{CH}_d}$. The counting hierarchy can be equivalently defined via threshold Turing machines: $\mathsf{CH}_d = \mathsf{Th}_d\mathsf{TIME}(n^{O(1)})$.

It is well-known that uniform $\mathsf{AC}^0(2^{\mathsf{poly}(n)})$ corresponds to the polynomial-time hierarchy PH [8]. Similarly, the correspondence exists between uniform $\mathsf{TC}^0(2^{\mathsf{poly}(n)})$ and the counting hierarchy CH [19,5,2]. For constructible $t(n)$ such that $t(n) = \Omega(\log n)$, we have $\cup_{d \geqslant 0}\mathsf{Th}_d\mathsf{TIME}(\mathsf{poly}(t(n)))$ is precisely the class of languages decided by uniform $\mathsf{TC}^0(2^{\mathsf{poly}(t(n))})$.

The following lemma gives the correspondence between weakly uniform threshold circuits and threshold TMs with advice. The proof follows from [3], and is left to the full version [6].

Lemma 2. *Let L be any language decided by a family of α-weakly uniform $d(n)$-depth threshold circuits of size $s(n)$. Then L is decidable by a threshold Turing machine with $d'(n) = 3d(n) + 2$ alternations, taking advice of length $\alpha(m)$ for $m = c_0 \log s(n)$, and running in time $t(n) = d'(n) \cdot \mathsf{poly}(m + \alpha(m))$.*

3 Indirect diagonalization

Here we establish the components needed for our indirect diagonalization, as outlined in Section 1.2. First, in Section 3.1, we give a diagonalization argument against alternating Turing machines with advice, getting a language in the counting hierarchy CH that is "hard" against weakly uniform TC^0 circuits of certain size. Then, in Section 3.2, using the assumption that a canonical P-complete problem has small weakly uniform TC^0 circuits, we conclude that the "hard" language given by our diagonalization step is actually hard for a stronger class of algorithms: weakly uniform Boolean circuits of some size s' *without any depth restriction*. Finally, in Section 3.3, using the assumption that PERMANENT

has small weakly uniform TC^0 circuits, we show that CH collapses, and our assumed hard language is in fact decidable by weakly uniform s'-size Boolean circuits, which is a contradiction. (Our actual argument is more general: we consider threshold circuits of not necessarily constant depth $d(n)$, and non-constant levels of the counting hierarchy.)

3.1 Diagonalization against ATMs with advice

Lemma 3. *For any constructible functions $\alpha, d, t, T : \mathbb{N} \to \mathbb{N}$ such that $\alpha(n) \in o(n)$ and $t(n) \log t(n) = o(T(n))$, there is a language $D \in \mathsf{Th}_{d(n)}\mathsf{TIME}(T(n))$ which is not decided by threshold Turing machines with $d(n)$ alternations running in time $t(n)$ and taking advice of length $\alpha(n)$.*

Proof. Define the language D consisting of those inputs x of length n that have the form $x = (M, y)$ (using some pairing function) such that the threshold TM M with advice y, where $|y| = \alpha(n)$, rejects input (M, y) in time $t(n)$ using at most $d(n)$ alternations. Language D is decided in $\mathsf{Th}_{d(n)}\mathsf{TIME}(T(n))$ by simulating M and flipping the result[4].

For contradiction, suppose that D is decided by some threshold Turing machine M_0 with $d(n)$ alternations taking advice $\{a_n\}$ of size $\alpha(n)$. Consider the input (M_0, a_n) with $|M_0| = n - \alpha(n)$; we assume that each TM has infinitely many equivalent descriptions (by padding), and so for large enough n, there must exist such a description of size $n - \alpha(n)$. By the definition of D, we have (M_0, a_n) is in D iff M_0 with advice a_n rejects it; but this contradicts the assumption that M_0 with advice $\{a_n\}$ decides D. □

3.2 If P is easy

Let L_0 be a P-complete language under uniform projections (functions computable by uniform Boolean circuits with NOT gates only). For example, the standard P-complete set $\{(M, x, 1^t) : M \text{ accepts } x \text{ in time } t\}$ works.

Lemma 4. *Suppose L_0 is decided by a family of α-weakly uniform $d(n)$-depth threshold circuits of size $s(n)$. Then, for any constructible function $t(n) \geqslant n$ and $0 \leqslant \beta(m) \leqslant 2^m$, every language L in β-weakly uniform $\mathsf{SIZE}(t(n))$ is decided by $\mu(n)$-weakly uniform $d(\mathsf{poly}(t(n)))$-depth threshold circuits of size $s'(n) = s(\mathsf{poly}(t(n)))$ on n inputs, where $\mu(n) = \alpha(c_0 \log s'(n)) + \beta(c_0 \log t(n))$.*

Proof. Let U be an advice-taking algorithm deciding the direct-connection language for the $t(n)$-size circuits for L. For any string y of length $\beta(m)$ for

[4] $\mathsf{Th}_{d(n)}\mathsf{TIME}(T(n))$ is closed under complement since the negation of MAJ is MAJ of negated inputs when MAJ has an odd number of inputs; the latter is easy to achieve by replacing $MAJ(x_1, \ldots, x_k)$ with $MAJ(x_1, x_1, \ldots, x_k, x_k, 0)$. Allender [2] uses a lazy diagonalization argument [30] for nondeterministic TMs. However, that argument seems incapable of handling the amount of advice we need. Fortunately, the basic diagonalization argument we use here is sufficient for our purposes.

$m = c_0 \log t(n)$, we can run U with the advice y to construct some circuit C^y of size $t(n)$ on n inputs. We can construct the circuit C^y in time at most $\mathsf{poly}(t(n))$, and then evaluate it in time $\mathsf{poly}(t(n))$ on any given input of size n.

Consider the language $L' = \{(x, y, 1^{t(n)}) \mid |x| = n,\ |y| = \beta(m),\ C^y(x) = 1\}$. By the above, we have $L' \in \mathsf{P}$. Hence, by assumption, L' is decided by an α-weakly uniform $d(l)$-depth threshold circuits of size $s(l)$, where $l = |(x, y, 1^{t(n)})| \leqslant \mathsf{poly}(t(n))$. To get a circuit for L, we simply use as y the advice of size $\beta(m)$ needed for the direct-connection language of the $t(n)$-size circuits for L. Overall, we need $\alpha(c_0 \log s(l)) + \beta(m)$ amount of advice to decide L by weakly uniform $d(\mathsf{poly}(t(n)))$-depth threshold circuits of size $s(\mathsf{poly}(t(n)))$. □

3.3 If Permanent is easy

Since PERMANENT is hard for the first level of the counting hierarchy CH, assuming that PERMANENT is "easy" implies the collapse of CH (see, e.g., [2]). It was observed in [16] that it is also possible to collapse super-constant levels of CH, under the same assumption. Below we argue the collapse of super-constant levels of CH by assuming that PERMANENT has "small" weakly uniform circuits.

We use the notation $f \circ g$ to denote the composition of the functions f and g, and the notation $f^{(i)}$ is used to denote the composition of f with itself for i times; we use the convention that $f^{(0)}$ is the identity function.

Lemma 5. *Suppose that* PERMANENT *is in γ-weakly uniform $\mathsf{SIZE}(s(n))$, for some $\gamma(m) \leqslant 2^{o(m)}$. For every $d(n) \leqslant n^{o(1)}$, every language A in $\mathsf{Th}_{d(n)}\mathsf{TIME}(\mathsf{poly})$ is also in $(2d(n)\cdot\gamma)$-weakly uniform $\mathsf{SIZE}((s \circ q)^{(d(n)+1)}(n))$, for some polynomial q dependent on A.*

Proof. The language A is computable by a uniform threshold circuit family $\{C_n\}$ of depth $d(n)$ and size $\mathsf{poly}(n)$. Let M be a polynomial-time TM deciding the direct-connection language of $\{C_n\}$. More precisely, we identify the gates of the circuit with the configurations of the given threshold TM for A; the output gate is the initial configuration; leaf (input) gates are halting configurations; deciding if one gate is an input to the other gate is deciding if one configuration follows from the other according to our threshold TM, and so can be done in polynomial time (dependent on A); finally, given a halting configuration, we can decide if it is accepting or rejecting also in polynomial time (dependent on A).

Consider an arbitrary n. Let $d = d(n)$. For a gate g of C, we denote by C_g the subcircuit of C that determines the value of the gate g. We say that g is at depth i, for $1 \leqslant i \leqslant d$, if the circuit C_g is of depth i. Note that each gate at depth $i \geqslant 1$ is a majority gate.

For every $0 \leqslant i \leqslant d$, let B_i be a circuit that, given $x \in \{0,1\}^n$ and a gate g at depth i, outputs the value $C_g(x)$.

Claim. There are polynomials q and q' dependent on A such that, for each $0 \leqslant i \leqslant d$, there are $2i\gamma$-weakly uniform circuits B_i of size $(s \circ q)^{(i)} \circ q'$.

Proof. We argue by induction on i. For $i = 0$, to compute $B_0(x, g)$, we need to decide if the halting configuration g of our threshold TM for A on input x is accepting or not; by definition, this can be done by the TM M in deterministic polynomial time. Hence, B_0 can be decided by a completely uniform circuit of size at most $q'(n)$ for some polynomial q' dependent on the running time of M.

Assume we have the claim for i. Let s' be the size of the γ'-weakly uniform circuit B_i, where $s' \leqslant (s \circ q)^{(i)} \circ q'$ and $\gamma' \leqslant 2i\gamma$. Consider the following TM N:

> "On input $z = (x, g, U, y, 1^{s'/2})$, where $|x| = n$, g is a gate of C, $|U| = \gamma(c_0 \log s')$, $|y| = \gamma'(c_0 \log s')$, interpret U as a Turing machine that takes advice y to decide the direct-connection language of some circuit D of size s' on inputs of length $|(x, g)|$. Construct the circuit D using U and y, where to evaluate U on a given input we simulate U for at most s' steps. Enter the MAJ state. Nondeterministically guess a gate h of C and a bit $b \in \{0, 1\}$. If h is not an input gate for g, then accept if $b = 1$ and reject if $b = 0$; otherwise, accept if $D(x, h) = 1$ and reject if $D(x, h) = 0$."

We will be interested in the case where U is a polynomial-time TM. For any such U, the running time on any input is bounded by $\mathsf{poly}(c_0 \log s' + \gamma'(c_0 \log s'))$, which is less than s' by our assumptions that $\gamma(m) \leqslant 2^{o(m)}$ and $d \leqslant (s')^{o(1)}$. Thus, to evaluate U on a particular input, it suffices to simulate U for at most s' steps, which is independent of what the actual polynomial time bound of U is. It follows that we can construct the circuit D (given U and y) in time $p(s')$, where p is a polynomial that does not depend on U. Also, to decide if h is an input gate to g, we use the polynomial-time TM M. We conclude that N is a PP machine which runs in some polynomial time (dependent on A). Since PERMANENT is PP-hard [26,31], we have a uniform reduction mapping z (an input to N) to an instance of PERMANENT of size $q(|z|)$, for some polynomial q (dependent on A).

By our assumption on the easiness of PERMANENT, we get that the language of N is decided by γ-weakly uniform circuits C_N of size at most $s'' = s(q(s'))$. If we plug in for U and y the actual TM description and the advice needed to decide the direct-connection language of B_i, we get from C_N the circuit B_{i+1}. Note that the direct-connection language of this circuit B_{i+1} is decided in polynomial time (using the algorithm for direct-connection language of C_N) given the advice needed for C_N plus the advice needed to describe U and y. The total advice size is at most $\gamma(c_0 \log s'') + \gamma(c_0 \log s'') + \gamma'(c_0 \log s') \leqslant 2(i + 1)\gamma(c_0 \log s'')$. □

Finally, we take the circuit B_d and use it to evaluate $A(x)$ by computing the value $B_d(x, g)$ where g is the output gate of C, which can be efficiently constructed (since this is just the initial configuration of our threshold TM for A on input x). By fixing g to be the output gate of C, we get the circuit for A which is $2d\gamma$-weakly uniform of size at most $(s \circ q)^{(d)}(r(n))$, where the polynomial r depends on the language A. Upper-bounding r by $(s \circ q)$ yields the result. □

4 Proofs of the main results

Here we use the technical tools from the previous section in order to prove our main results. Recall that L_0 is the P-complete language defined earlier.

4.1 Proof of Theorem 1

First, assuming L_0 is easy, we construct a hard language in CH.

Lemma 6. *Suppose L_0 is in* subexp*-weakly uniform TC^0 of depth d. Then, for a constant d' dependent on d, there is a language $L_{diag} \in \mathsf{CH}_{d'}$ which is not in* subexp*-weakly uniform* $\mathsf{SIZE}(\mathsf{poly})$.

Proof. Let $\alpha(m) \in 2^{o(m)}$ be such that L_0 is in α-weakly uniform TC^0 of depth d. Consider an arbitrary language L in β-weakly uniform $\mathsf{SIZE}(\mathsf{poly})$, for an arbitrary $\beta(m) \in 2^{o(m)}$. By Lemma 4, L has $\mu(n)$-weakly uniform threshold circuits of depth d and polynomial size, where $\mu(n) = \alpha(O(\log n)) + \beta(O(\log n)) \leqslant n^{o(1)}$. By Lemma 2, we have that L is decided by a threshold Turing machine with $d' = O(d)$ alternations, taking advice of length $\mu(n) \leqslant n^{o(1)} \leqslant n/\log^2 n$, and running in time $d' \cdot \mathsf{poly}(O(\log n) + n^{o(1)}) \leqslant n^{o(1)} \leqslant n/\log^2 n$. We conclude that *every* language in subexp-weakly uniform $\mathsf{SIZE}(\mathsf{poly})$ is also decided by some threshold TM in time $n/\log^2 n$, using d' alternations and advice of size $n/\log^2 n$.

Using Lemma 3, define L_{diag} to be the language in $\mathsf{Th}_{d'}\mathsf{TIME}(n)$ which is not decidable by any threshold Turing machine in time $n/\log^2 n$, using d' alternations and advice of size $n/\log^2 n$. It follows that L_{diag} is different from every language in subexp-weakly uniform $\mathsf{SIZE}(\mathsf{poly})$. □

Next, assuming PERMANENT is easy, we have that every language in CH is easy. The proof is immediate by Lemma 5.

Lemma 7. *If* PERMANENT *is in* subexp*-weakly uniform* $\mathsf{SIZE}(\mathsf{poly})$*, then every language in* CH *is in* subexp*-weakly uniform* $\mathsf{SIZE}(\mathsf{poly})$.

We now show that L_0 and PERMANENT cannot both be easy. The proof is immediate by Lemmas 6 and 7.

Theorem 4. *At least one of the following must be false:*

1. *L_0 is in* subexp*-weakly uniform TC^0;*
2. PERMANENT *is in* subexp*-weakly uniform* $\mathsf{SIZE}(\mathsf{poly})$.

To unify the two items in Theorem 4, we use the next lemma and its corollary.

Lemma 8 ([26,3]). *For every language $L \in \mathsf{P}$, there are uniform AC^0-computable function M (mapping a binary string to a poly-size Boolean matrix) and Boolean function f such that, for every x, we have $x \in L$ iff $f(\text{PERMANENT}(M(x)) = 1$.*

This lemma immediately yields the following.

Corollary 1. *If* PERMANENT *has α-weakly uniform $d(n)$-depth threshold circuits of size $s(n)$, then L_0 has α-weakly uniform $(d(n^{O(1)}) + O(1))$-depth threshold circuits of size $s(n^{O(1)})$.*

Now we prove Theorem 1, which we re-state below.

Theorem 5. PERMANENT *is not in* subexp*-weakly uniform* TC^0.

Proof. Otherwise by Corollary 1, both claims in Theorem 4 would hold, which is impossible. □

4.2 Proofs of Theorem 2 and Theorem 3

The following two lemmas are similar to Lemma 6, and are used to prove Theorems 2 and 3; the proofs can be found in the full version [6].

Lemma 9. *Suppose L_0 is in* poly*-weakly uniform* $\mathsf{TC}^0(\mathsf{subsubexp})$ *of depth d. Then, for a constant $d' = O(d)$, there is a language $L_{diag} \in \mathsf{CH}_{d'}$ which is not in* poly*-weakly uniform* $\mathsf{SIZE}(\mathsf{subsubexp})$.

Lemma 10. *Suppose L_0 is computable by* poly*-weakly uniform polynomial-size threshold circuits of depth $o(\log\log n)$. Then there exists a language $L_{diag} \in \mathsf{Th}_{\log\log n}\mathsf{TIME}(n)$ which is not in* poly*-weakly uniform* $\mathsf{SIZE}(n^{\mathsf{poly}(\log n)})$.

5 Other lower bounds

Using similar indirect diagonalization, we are also able to show the following; the proofs are left to the full version [6] of this paper.

Theorem 6. PERMANENT *is not in* poly*-weakly uniform* $\mathsf{ACC}^0(2^{n^{o(1)}})$.

Theorem 7. EXP *is not in* poly*-weakly uniform* $\mathsf{SIZE}(2^{n^{o(1)}})$.

Theorem 8. PSPACE *is not computable by* poly*-weakly uniform Boolean formulas of size* $O(2^{n^{o(1)}})$.

6 Conclusion

We have shown how to use indirect diagonalization to prove lower bounds against weakly uniform circuit classes. In particular, we have proved that PERMANENT cannot be computed by polynomial-size TC^0 circuits that are only slightly uniform (whose direct-connection language can be efficiently computed using sublinear amount of advice). We have also extended to the weakly uniform setting other circuit lower bounds that were previously known for the uniform case.

One obvious open problem is to improve the TC^0 circuit lower bound for PERMANENT to be exponential, which is not known even for the uniform case. Another problem is to get super-polynomial uniform TC^0 lower bounds for a language from a complexity class below #P (e.g., PH). Strongly exponential lower bounds even against uniform AC^0 would be very interesting. One natural problem is to prove a better lower bound against *uniform* AC^0 (say for PERMANENT) than the known non-uniform AC^0 lower bound for PARITY.

References

1. M. Agrawal. Proving lower bounds via pseudo-random generators. In *Proc. of the 25th Conf. on Foun. of Software Tech. and Theoretical Comp. Sci.*, p 92–105, 2005.

2. E. Allender. The permanent requires large uniform threshold circuits. *Chicago Journal of Theoretical Computer Science*, 1999.
3. E. Allender and V. Gore. A uniform circuit lower bound for the permanent. *SIAM Journal on Computing*, 23(5):1026–1049, 1994.
4. S. Arora and B. Barak. *Complexity theory: a modern approach*. CUP, NY, 2009.
5. D.A.M. Barrington, N. Immerman, and H. Straubing. On uniformity within NC^1. *JCSS*, 41:274–306, 1990.
6. R. Chen, and V. Kabanets. Lower bounds against weakly uniform circuits. In *ECCC*, 19:7, 2012.
7. A. Chandra, D. Kozen, and L. Stockmeyer. Alternation. *JACM*, 28(1):114, 1981.
8. M. Furst, J.B. Saxe, and M. Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, April 1984.
9. J. Håstad. Almost optimal lower bounds for small depth circuits. In *STOC*, 1986.
10. J. Heintz and C.-P. Schnorr. Testing polynomials which are easy to compute. *L'Enseignement Mathématique*, 30:237–254, 1982.
11. K. Iwama and H. Morizumi. An explicit lower bound of $5n - o(n)$ for boolean circuits. In *Proc. of the 27th Inte. Symp. on MFCS*, p 353–364. 2002.
12. M. Jansen and R. Santhanam. Permanent does not have succinct polynomial size arithmetic circuits of constant depth. In *Proc. 38th ICALP, I*, p 724–735, 2011.
13. V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1–2):1–46, 2004.
14. R. Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55:40–56, 1982.
15. R.M. Karp and R.J. Lipton. Turing machines that take advice. *L'Enseignement Mathématique*, 28(3-4):191–209, 1982.
16. P. Koiran and S. Perifel. A superpolynomial lower bound on the size of uniform non-constant-depth threshold circuits for the permanent. In *CCC*, 2009.
17. O. Lachish and R. Raz. Explicit lower bound of $4.5n - o(n)$ for boolean circuits. In *Proc. of the Thirty-Third ACM Symp. on Theory of Computing*, p 399–408, 2001.
18. O.B. Lupanov. On the synthesis of switching circuits. *Doklady Akademii Nauk SSSR*, 119(1):23–26, 1958. English translation in *Soviet Mathematics Doklady*.
19. I. Parberry and G. Schnitger. Parallel computation with threshold functions. In *Proc. of the First IEEE Conf. on Structure in Complexity Theory*, p 272–290, 1986.
20. A.A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes*, 41:333–338, 1987.
21. A.A. Razborov and S. Rudich. Natural proofs. *JCSS*, 55:24–35, 1997.
22. W.L. Ruzzo. On uniform circuit complexity. *JCSS*, 22(3):365–383, 1981.
23. C.E. Shannon. The synthesis of two-terminal switching circuits. *Bell System Technical Journal*, 28(1):59–98, 1949.
24. R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proc. of the Nineteenth ACM STOC*, p 77–82, 1987.
25. J. Torán. Complexity classes defined by counting quantifiers. *JACM*, 38:752, 1991.
26. L. Valiant. The complexity of computing the permanent. *TCS*, 8:189–201, 1979.
27. K.W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23:325–356, 1986.
28. R. Williams. Non-uniform ACC circuit lower bounds. In *CCC*, 2011.
29. A.C. Yao. Separating the polynomial-time hierarchy by oracles. In *FOCS*, 1985.
30. S. Zak. A Turing machine hierarchy. *TCS*, 26:327-333, 1983.
31. V. Zanko. #P-Completeness via Many-One Reductions. *IJFCS*, 1:77, 1991.