Lifting for Constant-Depth Circuits and Applications to MCSP

Marco Carmosino
Department of Computer Science, Boston University, MA, USA

Kenneth Hoover[1]
Department of Computer Science, University of California, San Diego, CA, USA

Russell Impagliazzo
Department of Computer Science, University of California, San Diego, CA, USA

Valentine Kabanets
School of Computing Science, Simon Fraser University, Burnaby, Canada

Antonina Kolokolova
Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada

Abstract

Lifting arguments show that the complexity of a function in one model is essentially that of a related function (often the composition of the original function with a small function called a *gadget*) in a more powerful model. Lifting has been used to prove strong lower bounds in communication complexity, proof complexity, circuit complexity and many other areas.

We present a lifting construction for constant depth unbounded fan-in circuits. Given a function f, we construct a function g, so that the depth $d+1$ circuit complexity of g, with a certain restriction on bottom fan-in, is controlled by the depth d circuit complexity of f, with the same restriction. The function g is defined as f composed with a parity function. With some quantitative losses, average-case and general depth-d circuit complexity can be reduced to circuit complexity with this bottom fan-in restriction. As a consequence, an algorithm to approximate the depth d (for any $d > 3$) circuit complexity of given (truth tables of) Boolean functions yields an algorithm for approximating the depth 3 circuit complexity of functions, i.e., there are quasi-polynomial time mapping reductions between various gap-versions of AC^0-MCSP. Our lifting results rely on a *blockwise* switching lemma that may be of independent interest.

We also show some barriers on improving the efficiency of our reductions: such improvements would yield either surprisingly efficient algorithms for MCSP or stronger than known AC^0 circuit lower bounds.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Circuit complexity

Keywords and phrases circuit complexity, constant-depth circuits, lifting theorems, Minimum Circuit Size Problem, reductions, Switching Lemma

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.44

Category Track A: Algorithms, Complexity and Games

Funding *Marco Carmosino*: Work supported by NSF Computing Innovation Fellowship.
Kenneth Hoover: Work supported by the Simons Foundation and NSF grant CCF-1909634.
Russell Impagliazzo: Work supported by the Simons Foundation and NSF grant CCF-1909634.
Valentine Kabanets: Work supported in part by an NSERC Discovery grant.
Antonina Kolokolova: Work supported in part by an NSERC Discovery grant.

Acknowledgements We thank the anonymous reviewers for their helpful and insightful comments.

[1] Corresponding author

1 Introduction

Determining the circuit complexity of a given Boolean function (presented by its truth table) is a fundamental algorithmic problem with a long history. One can view the whole area of circuit synthesis as consisting of variants of this problem. Historically, this is an apparently intractable search problem that motivated early thought on the P vs. NP problem; both Karp [21] and Levin [30] were thinking about this problem. It is also a fundamental problem from the point of view of computational complexity, where its complexity in different models has been related to cryptography, learning theory, derandomization, and circuit lower bounds.

Its decision version, known as the Minimum Circuit Size Problem (MCSP), asks to decide for a given truth table of a function f and a parameter s whether there is a circuit with at most s gates computing the function f. The high-level overview is that while any algorithmic improvement for MCSP would have dramatic consequences, thus making MCSP very likely to be extremely difficult, there are also many barriers to showing that MCSP is NP-complete [20, 24, 17, 5, 16, 4, 15, 29].

Intuitively, both algorithms for MCSP and completeness results for MCSP would require a better understanding of circuits, and in particular, would seem to be connected to proving lower bounds for circuit complexity. We could then hope to understand variants of MCSP where we restrict the class of circuits considered. However, MCSP remains mysterious even for the best understood circuit classes, such as constant depth unbounded fan-in circuits, where strong lower bounds are known. One aspect that adds to the mystery is that there is no known general relation between the minimum circuit size problems for various circuit classes, even when one class is stronger than the other, or even when they are almost identical in computational power.

Recently, Ilango [18] made break-through progress by showing NP-completeness for the constant-depth formula size version of MCSP under randomized quasi-polynomial time Turing reductions. He used a *lifting* argument to reduce the depth d formula complexity of a function f to the depth $d+1$ formula complexity of a related function f'. *Lifting arguments* show that the complexity of a function in one model is essentially that of a related function (often the composition of the original function with a small function called a *gadget*) in a more powerful model. Lifting has been used to prove strong lower bounds in communication complexity, proof complexity, circuit complexity and many other areas [26, 12, 25, 10]. As in the work of Ilango, lifting results can also be viewed as a *reduction* from the problem of computing the complexity of functions in the first model to that in the second model.

This intriguing result also raises a number of questions. In particular, what is the strength of lifting arguments as reductions between MCSP variants? Are the restrictions in Ilango's argument (formulas rather than circuits, randomization, quasi-polynomial time, and Turing rather than mapping) essential, or can at least some of these be eliminated? How general is the lifting technique, i.e., could there be natural reductions between these problems that do not use lifting? Our work is a first step towards answering some of these questions.

1.1 Our Results

To answer these questions, we present a lifting construction for constant depth unbounded fan-in circuits that given a function f, constructs a function g, so that the depth $d+1$ complexity of g is controlled by the depth d complexity of f. Our construction is very simple: g is f composed with a suitable sized parity function. Given that there are strong constant depth lower bounds using the properties of the parity function, the form of our construction is quite natural (and can be traced back to the classical Andreev function construction [6]).

However, converting this intuition into a formal lifting theorem is delicate. First, tight lifting for this construction seems to require that we look at circuits with restricted bottom gate fan-in, rather than general circuits. Even with this restriction, we need to get both lower bounds and upper bounds on the complexity of the constructed function g, that are very close together. So we need to take extra care for both parts of the argument. In particular, for the upper bound, it is not sufficient to compose the circuit for f with an optimal depth 2 circuit for the parity; instead, we show that the bottom level of the circuit for f applied to parities has a relatively small depth 2 circuit. For the lower bound, we need a variant of the classical Switching Lemma [13], because we cannot afford to have even a small probability that an input to f is removed by the restriction. Our Switching Lemma variant (Lemma 10) is actually both as easy or easier to prove than the standard one, and implies the standard switching lemma, so could have independent pedagogical value.

Our main lifting theorem (Theorem 18) shows that if a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is computable by a size s depth $d+1$ circuit with bottom gate fan-in at most $\log s$ (we term such circuits to be of depth $d + 1/2$), then the new "lifted" function $g = f \circ \oplus_\ell$ (f composed with parity on $\ell < n$ inputs) will have its depth $((d+1) + 1/2)$ circuit complexity sandwiched between $s^{1/2}$ and s^6.

We apply this lifting theorem to give reductions between approximately computing the constant-depth circuit complexity of a given function at depth d and any higher constant depth d'. Because our lifting theorems control the circuit complexity only approximately, we phrase these reductions as reductions between gap-versions of MCSP; these are promise problems to distinguish between functions of circuit complexity at most s_{yes} and of circuit complexity greater than s_{no}, for some parameters $s_{yes} < s_{no}$.

We first observe that the restriction on circuits we have in our lifting theorem does not change their average-case complexity substantially. This gives a reduction from small depth "tolerant" gap-MCSP to tolerant gap-MCSP for larger depths.

Also, by giving a non-trivial size/bottom fan-in trade-off for constant depth circuits, loosely based on the size-width trade-offs in proof complexity [8], we show that hardness for approximation of MCSP for weakly exponential sizes at one depth can be translated to similar hardness for higher depths.

Since the input to MCSP is the entire truth table of the function, the operation we give yields a quasi-polynomial, rather than polynomial time reduction, between MCSP for smaller depths and larger. Under a suitable assumption about the difficulty of MCSP, we show that any natural (in the sense of [20]) polynomial time reduction would either implicitly contain a stronger lifting theorem or yield a subexponential-time algorithm for small-depth MCSP.

Finally, we note that our lifting results are a step towards proving the NP-completeness of AC^0_d-MCSP for any constant depth $d \geq 3$: one just needs to prove that depth 3 circuit complexity is NP-hard to approximate to within the factors needed by our lifting theorems.

1.2 Related Work

The most closely related to our work is the recent result by Ilango [18] that there is a quasi-polytime randomized Turing reduction from the AC^0_d minimum formula size problem to the AC^0_{d+1} minimum formula size problem. The main theorem used for this reduction is a kind of lifting theorem, where for any f and a sufficiently hard to compute g relative to f, the AC^0_{d+1} formula complexity of $f \wedge g$ is close to the AC^0_d formula complexity of f plus the AC^0_{d+1} formula complexity of g. With this, they were able to show that AC^0 minimum formula size problem is NP-hard under such reductions. Their proof crucially relied on the model being *formulas*, whereas our work gives similar depth-increasing reductions for *circuits*, along with some barriers to improving these reductions.

Khot and Saket [22] proved essentially optimal hardness of approximation for DNF-GapMCSP, under the assumption that $\mathsf{NP} \not\subseteq \mathsf{QuasiP}$, via a connection shown by Feldman [11] between the gap problem and a hypercube covering problem. Under this same assumption, Hirahara, Oliveira, and Santhanam [15] showed that DNF $\circ$ MOD_m-GapMCSP was hard for $O(\log n)$ gaps. Both of these gap problems are known to be easy for $O(n)$ gaps [9, 15]. These results rely on the existing geometric characterization of the classes in question, the former being a union of subcubes, the latter a union of affine subspaces. For general fixed-depth circuits, we no longer have these characterizations; the bottom levels, being CNFs or DNFs, can represent any subset of the hypercube.

Allender and Hirahara [4] show that under modest cryptographic assumptions (the existence of a one-way function), GapMCSP for general circuits and $N^{1-o(1)}$ gaps is NP-intermediate. Their arguments depend heavily on being able to compose functions to large depths. See a recent survey by Allender [1] for more on MCSP-related results.

Remainder of the paper. We give basic definitions and prove our blockwise switching lemma in Section 2. We prove our lifting theorem for the worst-case AC^0 circuit complexity in Section 3, and for the average-case AC^0 circuit complexity in Section 4. We discuss the barriers to improving our lifting reductions in Section 5, and state some open questions in Section 6.

2 Preliminaries

2.1 General

▶ **Definition 1** (Boolean function composition). *The composition of boolean functions $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^m \to \{0,1\}$ is the function $f \circ g : \{0,1\}^{nm} \to \{0,1\}$ obtained by dividing the input z into n consecutive blocks $z_1, \dots, z_n$ of length m, applying g to each block, and then computing $f(g(z_1), g(z_2), \dots, g(z_n))$.*

▶ **Definition 2** (Time $t(n)$ many-one reductions). *We say that $A \leq_m^t B$ if there is a time $t = t(n)$-computable function $g\colon \{0,1\}^n \to \{0,1\}^{n'}$ such that $x \in A \iff f(x) \in B$. We denote the polynomial $t(n) = n^{O(1)}$ by* poly*, and quasipolynomial $t(n) = n^{(\log n)^{O(1)}}$ by* qpoly.

2.2 Complexity Measures

We denote the set of n-input Boolean functions by $\mathcal{F}_n$, and the set of all Boolean functions by $\mathcal{F}$. A *complexity measure* is a function $\Lambda : \mathcal{F} \to \mathbb{N}$ that maps Boolean functions to the natural numbers, quantifying some aspect of their complexity relative to a concrete computational model. The most basic such quantity is number of gates in a circuit C, denoted by $|C|$. We assume every circuit has at least one gate for each input bit.

▶ **Definition 3** (Constant-Depth Alternating Circuits: AC^0_d). *The* depth-d alternating circuit complexity *of f, denoted $\mathsf{AC}^0_d(f)$, is the minimum number of gates in any unbounded-fanin circuit computing f where gates are from the set $\{\wedge, \vee, \neg\}$, layers alternate, and the depth is at most d.*

▶ **Definition 4** (Bounded Bottom-Fanin Alternating Circuits: $\mathsf{AC}^0_{d+1/2}$). *The* bounded bottom fan-in depth-d alternating circuit complexity *of f, denoted $\mathsf{AC}^0_{d+1/2}(f)$, is the minimum number s of gates in any depth-$d+1$ circuit computing f with bottom fan-in at most* $\log s$. *Equivalently, it is the minimum over all depth-$d+1$ circuits C computing f of* $\max\{|C|, 2^{w_C}\}$, *where w_C denotes the bottom fan-in of C.*

We measure average-case complexity by expanding what counts as "computing" f.

▶ **Definition 5** (Tolerance Operator: $\widehat{\Lambda}$). *Let Λ be a complexity measure. Denote by* $\mathrm{dist}(f, \epsilon)$ *the Hamming ball of radius $\epsilon \cdot 2^n$ around the truth-table of f. The* ϵ-relaxation *of Λ is* $\widehat{\Lambda}[\epsilon] = \min_{f' \in \mathrm{dist}(f,\epsilon)} \Lambda(f')$.

2.3 Meta-Complexity Problems

Every meta-complexity problem is defined relative to a complexity measure.

▶ **Definition 6** (MCSP). *For a complexity measure Λ, $\Lambda\text{-}\mathtt{MCSP} = \{(f, s) \mid \Lambda(f) \leq s\}$.*

▶ **Definition 7** (GapMCSP). *For a complexity measure Λ, $\Lambda\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}]$ is the following promise problem, where f_n denotes an n-variate Boolean function:*

$$\mathcal{Y} = \{f_n \mid \Lambda(f_n) \leq s_{yes}\} \quad \textit{and} \quad \mathcal{N} = \{f_n \mid \Lambda(f_n) > s_{no}\}$$

▶ **Definition 8** (Tolerant GapMCSP). *For a complexity measure Λ, $\widehat{\Lambda}[\epsilon_1, \epsilon_2]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}]$ is the following promise problem:*

$$\mathcal{Y} = \{f_n \mid (\widehat{\Lambda}[\epsilon_1])(f_n) \leq s_{yes}\} \quad \textit{and} \quad \mathcal{N} = \{f_n \mid (\widehat{\Lambda}[\epsilon_2])(f_n) > s_{no}\}$$

2.4 Blockwise Switching Lemma

We will need a strengthening of Håstad's Switching Lemma [13] for the case of *structured* random restrictions that leave exactly one variable unset in every block of variables.

▶ **Definition 9** (Blockwise Restrictions, $\mathcal{B}_n^\ell$). *A binary string of length $n \cdot \ell$ can naturally be divided into n consecutive "blocks" of ℓ bits each. Variables $\{y_{i,j} : i \in [n],\ j \in [\ell]\}$ index into these strings. Denote by $\mathcal{B}_n^\ell$ the set of all restrictions ρ that place **exactly one** $\star$ in each block of an n-block, ℓ-block-size string. Formally, we have $\rho : [n] \times [k] \to \{0, 1, \star\}$ and $\forall i \in [n]\ \exists! j \in [k]$ such that $\rho(i, j) = \star$.*

▶ **Lemma 10** (Blockwise Switching Lemma). *Let φ be a k-CNF on $n \cdot \ell$ variables. For any $s \geq 0$, $\Pr_{\rho \sim \mathcal{B}_n^\ell}[\varphi \restriction_\rho$ cannot be expressed as an 2^s-term s-DNF$] \leq \left(\frac{8k}{\ell}\right)^s$.*

▶ Remark 11. While the proof of Lemma 10 is actually slightly simpler than that of the standard Switching Lemma [27, 7], this Blockwise Switching Lemma implies the standard Switching Lemma (as stated in [7]). A uniformly random subset of pn out of n variables can be chosen as follows: Randomly uniformly permute the n variables, then partition them into pn consecutive disjoint blocks of size $1/p$ each, and, finally, randomly uniformly choose exactly one variable from each of the pn blocks. For each fixed permutation of n variables, Lemma 10 applies with $\ell = 1/p$. We get that the probability that a given k-CNF fails to simplify to an s-DNF when hit with a random restriction that leaves exactly pn variables unset is upper-bounded by $(8pk)^s$.

▶ Remark 12. Our blockwise restrictions are different from Håstad's blockwise random restrictions used in the context of the AC^0 depth hierarchy theorem [13] (later improved to the average-case depth hierarchy theorem in [14]). Håstad's blockwise restrictions were designed to preserve the structure of Sipser's function; ours will allow us to recover a circuit for f from a higher-depth circuit for the composition $f \circ \oplus_\ell$ of f with parities over disjoint blocks of ℓ variables, for any Boolean function f. Because of this, the two random restriction distributions are very different, and the Switching Lemmas that result are quantitatively quite different.

We prove the Switching Lemma (Lemma 10) below, via a modification of the "compression"-based proof of the Switching Lemma due to Razborov [27, 7].

Canonical Decision Trees & Notation

We write assignments to a set of variables $\{x_i | i \in [n]\}$ as functions $\alpha : [n] \to \{0,1\}$. A k-CNF $\varphi(x_1, \ldots, x_n)$ is a conjunction of m clauses, where each clause is a disjunction over at most k literals. A **Decision Tree** is a binary tree where nodes are labeled by variables $x_1, \ldots, x_n$, and leaves and edges are labeled by constants $\{0,1\}$.

To evaluate a Decision Tree on an assignment α, begin at the root, labeled by some x_i. Move down the edge labeled by $\alpha(i)$. Repeat until you arrive at a leaf and report the constant labeling that leaf as the value of the tree.

Given a k-CNF $\varphi(x_1, \ldots, x_n) = C_1 \wedge \cdots \wedge C_m$ we can create a ***Canonical* Decision Tree**. Fix a lexical ordering on variables and use it to sort and de-duplicate clauses; let $i \in [m]$ index the clauses of φ in this sorted order. We define $\mathrm{CDT}(\varphi)$ recursively:

> Transform $C_1 \in \varphi$ to a depth $\leq k$ tree T querying all variables of C_1 in lexical order. For each branch b of T, follow b to induce a partial assignment α_b; set $\varphi_b \leftarrow$ Simplify φ/α_b. If φ_b is empty, terminate b with leaf labeled 1; if φ_b is falsified, terminate b with leaf labeled 0; otherwise, if φ_b is undetermined, extend b with $\mathrm{CDT}(\varphi_b)$.

A restriction is a *partial assignment*: a map $\rho : [n] \to \{0, 1, \star\}$. The result of applying a restriction to a Boolean function f is written $f \restriction_\rho$ where we substitute each occurrence of x_i by $\rho(i)$ for every $\rho(i) \neq \star$. We will need to define restrictions that *extend* other restrictions. Let $\mathcal{E}_D(\rho)$ denote the set of restrictions that are identical to ρ, except for replacing D star locations with constants. Let $\mathcal{E}(\rho)$ denote the set of restrictions that replace *all* $\star$-locations of ρ with constants. We will be concerned with the blockwise restrictions of Definition 9.

Coding and Decoding Large-Depth Restrictions

Suppose all we know about ρ is that it produces a large-depth canonical decision tree when applied to φ. We can witness this with some "long" path σ through the tree. Our code will consist of a restriction $\tilde{\rho}_c$ that extends ρ and a short bitstring "hint" that allows us to implicitly navigate "down" a long path of the CDT and guess ρ by un-setting variables of ρ_c.

Algorithm 1 ENC.

```
Let $\sigma$ be a long path ($\geq$ depth $D$) through $T$;
foreach clause along $\sigma$, $C_i^\sigma$ do
    foreach variable $\eta_{ij}$ appearing in $C_i^\sigma$ do
        record the following hint: begin
            $\eta_{ij}$ as an index into $C_i^\sigma$ ($\log k$ bits);
            Assignment to $\eta_{ij}$ along $\sigma$ (single bit);
            Is this the last variable queried in $C_i^\sigma$? (single bit);
    Record $\tau_i$ as an assignment to $\eta_i$ that falsifies $C_i^\sigma$;
$\rho_c = \rho \circ \tau_1 \circ \cdots \circ \tau_D$
return $\tilde{\rho}_c \leftarrow \rho_c$ completed to a full assignment uniformly at random, all hints
```

▷ Claim 13 (Decoding from ENC output). Suppose $\rho \in \mathcal{B}_n^\ell$ fails to simplify a particular k-CNF φ, so that $\mathrm{CDT}(\varphi \restriction_\rho) \geq D$. Then, $\Pr_{\tilde{\rho}_c \sim \mathtt{ENC}(\rho)} [\mathtt{DEC}(\tilde{\rho}_c) = \rho] \geq \left(\frac{1}{\ell}\right)^{(n-D)}$.

Proof of Claim 13. Fix $\rho \in \mathcal{B}_n^\ell$ and suppose $T = \mathrm{CDT}(\varphi \restriction_\rho)$ has depth $\geq D$. Let σ be a witnessing path of length at least D through T. We'll require some notation; denote by C_i^σ the ith clause traversed along the path σ, in the sense that the recursive CDT construction

Algorithm 2 DEC.

```
Initialize: $\rho_1 \leftarrow \rho_c = \rho \circ \tau_1 \circ \cdots \circ \tau_D$ and $i \leftarrow 1$
for $i = 1$ to $D$ do
    Simplify $\varphi_i \lceil_{\rho_i}$ ;
    Find first falsified clause of $\varphi_i = C_i^\sigma$;
    Read hint to find $\tau_i$ and $\sigma_i$ (stop-bit tells you when to stop).;
    $\rho_{i+1} \leftarrow \rho_i$ with $\tau_i$ replaced by $\sigma_i$ (so $\rho_{i+1} = \rho \circ \sigma_1 \circ \sigma_{i-1} \circ \sigma_i \circ \tau_{i+1} \ldots \tau_D$)
return $\rho_D$ with $\sigma_1 \circ \ldots \sigma_D$ unset, and $\star$'s guessed uniformly at random for all other
  blocks
```

worked on clause C to produce that section of the decision tree. Note, this may well be smaller than m, due to simplifications applied during construction of the CDT. Further, let σ_i be the section of σ that traverses C_i^σ and let η_i be the variables queried along σ_i. We can think of σ_i as a sequence of assignments to these variables.

Now, consider the operation of DEC on $\tilde{\rho}_c \sim$ ENC(ρ). First observe that no clauses of φ were falsified by ρ alone, by our assumption that CDT($\varphi \lceil_\rho) \geq D$ – a falsified clause would give a depth-1 decision tree with a single 0 leaf. Therefore, any falsified clause is due to variables set by some τ_i or a randomly set variable.

Because the CDT is constructed in lexical-clause-order and ENC follows this order, the *first* falsified clause of $\varphi \lceil_{\tilde{\rho}_c}$ must be C_1^σ. We wish to recover which variables τ_1 set; the trick is that now we know they must reside in a uniquely identified clause of at most k variables. So, we spend $\log k$ bits of the hint per variable to name *which* variables of C_1^σ were along σ and thus set in τ_1.

Iterating this argument, we see that lexical ordering of the canonical decision tree ensures recovery of D $\star$ locations of ρ. So, after running the main loop of DEC on $\tilde{\rho}_c$ we have a candidate that matches ρ exactly in D blocks. For each remaining block, DEC will simply guess at random which variable in the block was a $\star$ in ρ. Each block has ℓ bits, so we have a $(1/\ell)$ chance of guessing correctly – that is, in agreement with the original location of the $\star$ in ρ. The number of blocks that must be guessed (instead of recovered using deterministic decoding, DEC) is $(n - D)$. Every guess must be correct to successfully decode ρ. This gives the claimed probability of decoding. ◁

Given instead a *random* completion ρ_r of blockwise restriction ρ and a *random* hint h_r, can any algorithm decode ρ? We can upper bound this probability.

▷ Claim 14 (Decoding from random information). For any algorithm $\mathcal{A}$, for every blockwise restriction ρ, $\Pr_{\rho_r \sim \mathcal{E}(\rho)}[\mathcal{A}(\rho_r, h_r) = \rho] \leq \left(\frac{1}{\ell}\right)^n$.

Proof of Claim 14. The hint h_r is clearly useless, because it is a random string. Furthermore, the random variables ρ_r and ρ are conditionally independent, given that ρ_r is a randomly sampled completion of ρ. This means that observing ρ_r provides *no information* regarding the $\star$-locations of ρ. Therefore, no algorithm can do better than to randomly guess which location, in each block, was a star, for every block of the received ρ_r. There are n blocks of ℓ bits each and every guess must be correct, for the overall probability $(1/\ell)^n$. ◁

Completing the proof

We are now ready to prove Lemma 10, re-stated below in a more general form.

▶ **Lemma 15** (Blockwise Switching Lemma). *Let φ be a k-CNF. Pick ρ from $\mathcal{B}_n^\ell$ uniformly at random. Then* $\Pr[\mathrm{CDT}(\varphi\restriction_\rho) \geq D] \leq \left(\frac{8k}{\ell}\right)^D$.

Proof. We can lower-bound the probability of decoding from random completion ρ_r: if we are lucky enough that the randomly sampled completion agrees with ρ_c in the "special" blocks set by ENC, then we can significantly narrow down the number of blocks whose $\star$ must be guessed at random! That is, the non-trivial probability of recovery for DEC can be exploited. Formally,

$$\begin{aligned}\left(\frac{1}{\ell}\right)^n &\geq \Pr[\mathtt{DEC}(\rho_r, h_r) = \rho] && \text{(by Claim 14)}\\ &\geq \Pr[\mathrm{CDT}(\varphi\restriction_\rho) \geq D] \times \Pr[h_r = h] \times \Pr[\rho_r \text{ extends } \rho_c] \times \Pr[\mathtt{DEC} \text{ decodes} \star\text{'s}]\end{aligned}$$

Taking each event in turn:

1. $|h| = D(\log(k)+2)$ so there are $(4k)^D$ possible strings. Flipping h_r uniformly at random, $\Pr[h_r = h] = (4k)^{-D}$.
2. To extend ρ_c, the randomly chosen ρ_r must agree in D locations. One of these settings is correct, so $\Pr[\rho_r \text{ extends } \rho_c] = 2^{-D}$.
3. Given a correct hint and randomly completed ρ_c, the probability of DEC recovering ρ is $\left(\frac{1}{\ell}\right)^{(n-D)}$ by Claim 13.

Plugging in, we get

$$\left(\frac{1}{\ell}\right)^n \geq \Pr[\mathrm{CDT}(\varphi\restriction_\rho) \geq D] \times (4k)^{-D} \times 2^{-D} \times \left(\frac{1}{\ell}\right)^{(n-D)}.$$

The proof of the lemma follows. ◀

3 Constant-Depth GapMCSP Reductions

The focus of this section is "hardness lifting" for circuits of depth $(d+1)$ to circuits of depth $(d+2)$, and its applications to GapMCSP for the respective classes. Theorem 18 shows how to lift hardness for bounded-fan-in AC^0 circuits from depth d to depth $d+1$ (also bounded fan-in). Here, a function of a not much larger size yet higher depth is constructed by replacing input variables of the original function by disjoint relatively small parities. This theorem is then applied to reduce GapMCSP for AC^0_d circuits to GapMCSP for AC^0_{d+1} circuits.

The reduction proceeds in three steps, with the middle step potentially repeated multiple times for a larger depth increase. The first step converts unbounded bottom fan-in circuits of depth $(d+1)$ to bounded (by log of the circuit size) fan-in circuits of the same depth, at the cost of increasing the size from s to $2^{O(\sqrt{n \log n \log s})}$; see Corollary 17. This rebalancing only needs to be done once.

The second step, which relies on the hardness lifting theorem, is the quasi-polynomial time reduction from GapMCSP for bounded bottom fan-in circuits of depth $(d+1)$, to GapMCSP for bounded bottom fan-in circuits of depth $(d+2)$. The quasi-polynomial running time of this reduction comes from the blow-up in the size of the output truth table of the new function. Then, we show that for this setting GapMCSP for depth $d+1$ circuits reduces to GapMCSP for depth $d+2$ circuits (both bounded bottom fan-in), with a small loss in the gap size. See Theorem 19 for the exact statement.

The last step is a polytime reduction from GapMCSP for bounded bottom fan-in circuits of depth $(d+2)$, to GapMCSP for unbounded bottom fan-in circuits of the same depth; see Theorem 21.

3.1 Depth $d+1$ to $d+1/2$

▶ **Lemma 16** (Fanin vs Size Tradeoff). *For any $d \geq 3$, let C be any depth-d size-s circuit over n inputs. Then, for any $w \geq 1$, there is an equivalent depth-d circuit C' with bottom fan-in at most w, and the size at most $s^{(4n\log n)/w}$.*

Proof. Assume WLOG that all the bottom gates of C are disjunctions. We will recursively define a decision tree T such that each leaf ℓ is associated with a restriction ρ_ℓ resulting in $C\restriction_{\rho_\ell}$ having bottom fan-in at most w.

Initially, T consists of a single leaf node corresponding to the empty restriction. While there is a leaf v in T corresponding to a restriction ρ such that $C\restriction_\rho$ has some nonempty set S of bottom gates of fan-in greater than w, do the following. Let $t = |S| \leq s$. Let z be the literal that occurs in the most gates of S. Since there are more than tw literal occurrences among the gates in S and there are $2n$ literals, z must appear in more than $(tw)/(2n)$ bottom gates. Branch on z, with the left child v_1 of v corresponding to $z=1$, and the right child v_0 to $z=0$. Note that the restriction corresponding to v_1 satisfies all bottom gates containing z, and the restriction corresponding to v_0 reduces their fan-in by 1.

Every left branching we take in the decision tree results in t shrinking by more than a factor $\left(1-\frac{w}{2n}\right)$. So after k left branchings, there are fewer than $t\left(1-\frac{w}{2n}\right)^k$ large fan-in gates left. Setting $k=(2n/w)\ln s$, we have that after k left branchings there are no large fan-in gates left. If $k \geq n/2$ (i.e., $w \leq 4\ln s$), then we can use the trivial upper bound 2^n on the size of T; note that, in this case, $2^n \leq 2^{4n\cdot(\ln s)/w}$, as required.

Otherwise, for $k < n/2$, we can upper-bound the size of T as follows. Since each branch of T is of length at most n, and it may contain at most k left branchings, we get that the size of T is at most

$$\sum_{r=0}^{k}\binom{n}{r} \leq k\cdot\binom{n}{k} \leq k\cdot\left(\frac{ne}{k}\right)^k \leq k\cdot\left(\frac{we}{2\ln s}\right)^k \leq 2^{(1.5)k\cdot\log(w/\ln s)} \leq s^{\frac{3n\log n}{w}},$$

where for the last inequality we used the definition of k and the bound $(w/\ln s) \leq w \leq n$.

Suppose without loss of generality that the top gate of C is a disjunction, i.e., $C = \bigvee_i \bigwedge_j g_{i,j}$. We can rewrite $C(x)$ as $\bigvee_{\text{leaves } \ell\in T}\left(\phi(x,\rho_\ell)\wedge C\restriction_{\rho_\ell}(x)\right)$, where, for a fixed restriction ρ_ℓ, the formula $\phi(x,\rho_\ell)$ indicates whether x is consistent with ρ_ℓ (i.e., whether x ends up at leaf ℓ of our decision tree T). It is easy to see that $\phi(x,\rho_\ell)$ can be written as a conjunction of at most n literals.

As written, the circuit above is a depth-$(d+2)$ size at most $(1+|T|+|T|\cdot s)$ circuit with fan-in at most w. By distributivity, we can rewrite each $\phi(x,\rho_\ell)\wedge C\restriction_{\rho_\ell}(x)$ as $\bigvee_i\left(\phi(x,\rho_\ell)\wedge\bigwedge_j\left(g_{i,j}\restriction_{\rho_\ell}(x)\right)\right)$. Plugging this into (3.1), we obtain a depth-d circuit C' with fan-in at most w, computing the same function as C, and the size of C' is at most $|T|\cdot s \leq s^{(4n\log n)/w}$, as required. ◀

▶ **Corollary 17** (Depth $(d+1)\to(d+1/2)$). *For any $d \geq 2$, n, and s_{yes}, s_{no} such that $\log s_{yes} \leq \frac{\log^2(s_{no}/4)}{n\log n}$, we have*

$$\mathsf{AC}^0_{d+1}\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \leq_m^{\mathsf{poly}} \mathsf{AC}^0_{d+1/2}\text{-}\mathtt{GapMCSP}_n[2^{4\cdot\sqrt{n(\log n)(\log s_{yes})}}, s_{no}]$$

with the identity functions as a reduction.

Proof. The "NO→NO" case is immediate: if $f\colon\{0,1\}^n\to\{0,1\}$ doesn't have size-s_{no} circuits with no restriction on the bottom fan-in, then f doesn't have size-s_{no} circuits with restricted bottom fan-in.

For the "YES→YES" case, we apply Lemma 16 to a depth-$(d+1)$ size-s_{yes} circuit for f, with $w = \sqrt{n(\log n)(\log s_{yes})}$. This results in a circuit for f of size at most $2^{4\cdot\sqrt{n(\log n)(\log s_{yes})}}$, with bottom fan-in at most $\sqrt{n(\log n)(\log s_{yes})}$. ◀

3.2 Depth $d + 1/2$ to $(d + 1) + 1/2$

▶ **Theorem 18** (Hardness lifting). *Let f have $\mathsf{AC}^0_{d+1/2}$ circuit complexity s. Fix $s_0 > 0$. Then there is a function f' on $n' = n \cdot 16 \log s_0$ inputs with $\mathsf{AC}^0_{(d+1)+1/2}$ circuit complexity s' where $s' \le 2s2^{\sqrt{16 \log s \log s_0}}\sqrt{16 \log s \log s_0}$. Moreover, if $s_0 < \sqrt{s/3}$, then $s_0 \le s'$.*

Proof. The construction is as follows: given the truth table of $f \colon \{0,1\}^n \to \{0,1\}$, output the truth table of $f' = f \circ \oplus_\ell$ for $\ell = 16 \log s_0$. This takes time $2^{n\ell} = N^{16 \log s_0} \le N^{O(\log N)}$, quasi-polynomial in N since $s_0 \le N$. We argue the correctness next.

Bounding s' from below. Note that the parameter ℓ must be sufficiently larger than $\log s_0$ so that we can apply the Blockwise Switching Lemma to a depth-$(d+2)$ size-s_0 circuit with bottom fan-in $\log s_0$ that presumably computes $f \circ \oplus_\ell$ to obtain a depth-$(d+1)$ size-s circuit with bottom fan-in $\log s$ that computes f. We prove that if f' has a $\mathsf{AC}^0_{(d+1)+1/2}$ of size s_0, then f has a $\mathsf{AC}^0_{d+1/2}$ circuit of size $s \le 3(s_0)^2$.

Suppose $f \circ \oplus_\ell$ has a depth-$(d+2)$ circuit C' of size s_0 and bottom fan-in at most $\log s_0$. We shall hit C' with a blockwise random restriction ρ, where the blocks are the inputs to each $\oplus_\ell$. Since exactly one bit is left unset in each block, $C' \restriction_\rho$ computes f with some of the input bits potentially negated. For $C' \restriction_\rho$ to simplify to a depth-$(d+1)$ circuit with bottom fan-in at most $k \le \log(3s_0^2) \le \log s$, we need to argue that there exists a blockwise restriction ρ which makes every depth-2 bottom circuit of C' into a decision tree of depth at most k. By the Blockwise Switching Lemma (Lemma 10), this is implied if $s_0 \left(\frac{8 \log s_0}{\ell}\right)^k < 1$, which is equivalent to $2^{\log s_0 - k} < 1$, for our choice of $\ell = 16 \log s_0$. Thus, setting $k = \log s_0 + 1$ satisfies this inequality. Moreover, each bottom CNF or DNF of C' is turned into a DNF or CNF with 2^k clauses. So the size of $C' \restriction_\rho$ is at most $s_0 + s_0 \cdot 2^k \le 3(s_0)^2 \le s$, as required.

Bounding s' from above. Next we need to show that if f has a small depth-$(d+1/2)$ circuit, then $f \circ \oplus_\ell$ has a small depth-$(d+1+1/2)$ circuit. Note that computing the ℓ-bit parities by naive depth-2 circuits of size 2^ℓ is prohibitively expensive, as this would make the size of the new circuit for $f \circ \oplus_\ell$ at least $(s_0)^{16} > s'$, for our choice of $\ell = 16 \log s_0$ (which was dictated by the "NO→NO" case analysis above). Instead we will compute each $\oplus_\ell$ by a depth-3 circuit, as a parity of parities, adapting the standard construction of optimal size-$(\ell 2^{\sqrt{\ell}})$ depth-3 circuits. To get a final circuit for $f \circ \oplus_\ell$ to be of depth $d+1+1/2$, we will need to carefully balance the parameters of our partition of ℓ bits into ℓ_1 blocks of size ℓ_2 each, for ℓ_1 and ℓ_2 such that $\ell = \ell_1 \cdot \ell_2$.

Suppose f has a depth-$(d+1)$ circuit C of size s and bottom fan-in at most $\log s$, with all negations at the leaves; this at most doubles the size. Without loss of generality, assume that the bottom layer of gates consists of disjunctions with fan-in $\log s$. To obtain a circuit for $f \circ \oplus_\ell$, we will compose $\oplus_\ell$ with each of the bottom CNFs of C. Consider a particular CNF $h_i = \bigwedge_{j=1}^k g_{i,j}$ at the bottom of C, where $k \le s$ and each $g_{i,j}$ is a disjunction of at most $\log s$ literals.

For ℓ_1 to be chosen later, let $\ell_2 = \ell/\ell_1$. Using the trivial 2^{ℓ_1}-size CNF for computing $\oplus_{\ell_1}$, we can compute each $g_{i,j} \circ \oplus_{\ell_1}$ by an OR-AND-OR circuit, where the top OR gate has fan-in $\log s$ and the AND gates each have fan-in 2^{ℓ_1}. By distributivity, we can rewrite $g_{i,j} \circ \oplus_{\ell_1}$ as a CNF with $2^{\ell_1 \log s}$ clauses, each of width at most $\ell_1 \log s$.

Since C is a layered circuit, we can merge this CNF into h_i to obtain a depth-2 circuit computing $h_i \circ \oplus_{\ell_1}$. Finally, composing this with the DNF for $\oplus_{\ell_2}$, we get a depth-3 circuit with bottom fan-in ℓ_2 computing $h_i \circ \oplus_\ell$. Replacing each h_i in C with circuits constructed in this way, we obtain a depth-$(d+2)$ circuit for $f \circ \oplus_\ell$ with bottom fan-in ℓ_2. The subcircuit for computing each $g_{i,j} \circ \oplus_\ell$ is of size at most $\sigma = 1 + 2^{\ell_1 \log s} + 2^{\ell_2} \cdot \ell_1 \log s$. So the total size of the circuit for $f \circ \oplus_\ell$ is at most $s + s \cdot \sigma = s(\sigma+1)$. If we set $\ell_2 = \sqrt{\ell \log s}$ and $\ell_1 = \sqrt{\frac{\ell}{\log s}}$, then the total size is at most

$$s\left(2 + 2^{\sqrt{\ell \log s}} + 2^{\sqrt{\ell \log s}} \cdot \sqrt{\ell \log s}\right) \le (2s) \cdot 2^{\sqrt{\ell \log s}} \cdot \sqrt{\ell \log s} \le s'.$$

Since the bottom fan-in is at most $\sqrt{\ell \log s} \le \log s'$, this concludes the proof. ◀

▶ **Theorem 19** (Depth $(d + 1/2) \to ((d+1) + 1/2)$ GapMCSP). *For any $d \ge 1$, n, s_{yes}, s'_{yes}, s'_{no}, and s_{no} such that $s_{yes} < s_{no}$, $s'_{yes} < s'_{no}$, $s_{no} \ge 3(s'_{no})^2$ and*

$$s'_{yes} \ge 2(s_{yes}) 2^{\sqrt{16 (\log s_{yes})(\log s'_{no})}} \sqrt{16 (\log s_{yes})(\log s'_{no})},$$

we have $\mathsf{AC}^0_{d+1/2}\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \le_m^{\mathsf{qpoly}} \mathsf{AC}^0_{(d+1)+1/2}\text{-}\mathtt{GapMCSP}_{n'}[s'_{yes}, s'_{no}]$, *where $n' = 16 n \log s'_{no} \le O(n^2)$.*

Proof. We use the construction in Theorem 18 as the reduction function, with $s_0 = s'_{no}$. For the YES $\to$ YES side, if $AC^0_{d+1/2}(f) \le s_{yes}$, then

$$\mathsf{AC}^0_{d+1+1/2}(f') \le 2(s_{yes}) 2^{\sqrt{16 (\log s_{yes})(\log s'_{no})}} \sqrt{16 (\log s_{yes})(\log s'_{no})}$$

as desired. For the NO $\to$ NO side, if $AC^0_{d+1/2}(f) > s_{no}$, then $\mathsf{AC}^0_{d+1+1/2}(f') \ge s_0 = s'_{no}$. ◀

▶ Remark 20. If we apply this to succinct $\mathtt{MCSP}$, we actually get a polytime reduction instead; constructing the naïve $f \circ \oplus_\ell$ circuit given a circuit for f takes polytime, it just makes the truth table too large.

3.3 Depth $d + 1/2$ to $d + 1$

▶ **Theorem 21** (Depth $(d + 1/2) \to (d+1)$). *For any $d \ge 1$, $n, s_{yes}, s_{no}, s'_{yes}, s'_{no}$, such that $s_{yes} < s_{no}$, $s'_{yes} < s'_{no}$, $s_{no} \ge (s'_{no})^5$ and $s'_{yes} \ge 2(s_{yes})^3$, we have*

$$\mathsf{AC}^0_{d+1/2}\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \le_m^{\mathsf{poly}} \mathsf{AC}^0_{d+1}\text{-}\mathtt{GapMCSP}_{2n}[s'_{yes}, s'_{no}]$$

Proof. The reduction is as follows: given the truth table of $f \colon \{0,1\}^n \to \{0,1\}$, output the truth table of $g = f \circ \oplus_2$. The size of the input for g is $2n$. The runtime of the reduction is $\mathsf{poly}(N)$. Next we argue the correctness of this reduction.

NO $\to$ NO. Suppose $f \circ \oplus_2 \colon \{0,1\}^{2n} \to \{0,1\}$ is computable by a size-s'_{no} circuit C' of depth $d+1$. Without loss of generality, we may assume that the bottom gates of C' are ANDs. We will hit C' with a random blockwise restriction ρ. Consider a particular bottom AND-gate of fan-in t, for some $1 \le t \le n$. Since each block in a blockwise restriction is of size two, there must be at least $t/2$ variables from distinct blocks that feed into this AND gate. Each one of these variables will be chosen as a non-star variable by ρ with probability $1/2$, and then independently set to 0 with probability $1/2$. This would simplify the AND gate to the constant 0, with probability $1/4$. This happens independently for each of these

$t/2$ variables. Thus the probability that the AND gate of fan-in at least t survives a random restriction is at most $(3/4)^{t/2}$. By the union bound, the probability that any such AND gate survives is at most $s'_{no} \cdot (3/4)^{t/2}$, which is less than 1 for $t = 5(\log s'_{no})$. Thus there exists a blockwise restriction ρ which simplifies C' to a depth-$(d+1)$ circuit computing f, with size at most $s'_{no} \leq s_{no}$ and bottom fan-in at most $5(\log s'_{no}) \leq \log s_{no}$.

YES $\to$ YES. Suppose $f\colon \{0,1\}^n \to \{0,1\}$ is computable by a size-s_{yes} circuit C of depth $d+1$, with bottom fan-in at most $\log s_{yes}$. WLOG, assume the bottom gates of C are ANDs. Note that we can express the XOR and the negated XOR of two variables as the following 2-CNFs:

$$y \oplus z = (\bar{y} \vee \bar{z}) \wedge (y \vee z) \qquad \text{and} \qquad \neg(y \oplus z) = (\bar{y} \vee z) \wedge (y \vee \bar{z}).$$

Replacing the input literals of C by these circuits for (possibly negated) $\oplus_2$, and merging the bottom AND gate of C with the top AND gate of these parity circuits, we get a depth-$(d+2)$ circuit C' for $f \circ \oplus_2$, with 2-CNFs on $t = (2\log s_{yes})$ clauses as the bottom depth-2 sub-circuits. By distributivity, we can rewrite each 2-CNF on t clauses as a t-DNF on 2^t terms. Then merge the OR gates of these DNFs with the OR gates at the preceding level in C', obtaining an equivalent depth-$(d+1)$ circuit C'' for $f \circ \oplus_2$, of size at most $s_{yes} + s_{yes} \cdot (s_{yes})^2 \leq 2(s_{yes})^3 \leq s'_{yes}$ (and bottom fan-in at most $(2\log s_{yes}) \leq \log s'_{yes}$). ◀

3.4 Combining the steps: Depth $d+1$ to $d+c$ for any constant $c > 1$

The reduction in Theorem 19 can be repeated multiple times, resulting in the overall reduction lifting hardness to constantly many levels. The following theorem shows how the parameters evolve over all steps of the reduction.

▶ **Theorem 22** (Depth $(d+1) \to (d+c)$)**.** *For any $d \geq 2$, $c > 1$, $n \geq n_0(\alpha, \delta, c)$, and $0 < \alpha < \delta < 1$ where $1 + \alpha < 2\delta$, we have*

$$\mathsf{AC}^0_{d+1}\text{-}\mathtt{GapMCSP}_n[2^{n^\alpha}, 2^{n^\delta}] \leq_m^{\mathsf{qpoly}} \mathsf{AC}^0_{(d+c)}\text{-}\mathtt{GapMCSP}_{n'}[2^{(n')^\beta}, 2^{(n')^\gamma}],$$

where $n' = n^{(c-1)\delta+1}$, $\gamma \approx \frac{1}{c-1}$, $\beta \approx \frac{1}{c-1} - \frac{1}{(c-1)2^{c-1}} \cdot (1 - \frac{1+\alpha}{2\delta})$.

Proof. As outlined at the beginning of the section, we will create this reduction via composing the reductions in Corollary 17 and Theorems 19 and 21. Let $a = \frac{1+\alpha+\frac{\log\log n+4}{\log n}}{2}$.

Step 1. $\mathsf{AC}^0_{d+1}\text{-}\mathtt{GapMCSP}_n[2^{n^\alpha}, 2^{n^\delta}] \leq_m^{\mathsf{poly}} \mathsf{AC}^0_{d+1/2}\text{-}\mathtt{GapMCSP}_n[2^{n^a}, 2^{n^\delta}]$
This follows immediately from Corollary 17 with $s_{yes} = 2^{n^\alpha}$ and $s_{no} = 2^{n^\delta}$.

Step 2.
$\mathsf{AC}^0_{d+1/2}\text{-}\mathtt{GapMCSP}_n[2^{n^a}, 2^{n^\delta}] \leq_m^{\mathsf{qpoly}} \mathsf{AC}^0_{(d+c-1)+1/2}\text{-}\mathtt{GapMCSP}_{n^{(c-1)\delta+1}/2}[\exp_2\left(5n^{\frac{a+(2^{c-1}-1)\delta}{2^{c-1}}}\right),$
$\exp_2(\frac{n^\delta}{2^{c-1}} - \frac{2^{c-1}-1}{2^{c-1}}\log 3)]$
We will show each of n, s_{yes}, and s_{no} map to the corresponding values after $c-1$ applications of the reduction in Theorem 19. Define $n^{(i)}$, $s^{(i)}_{yes}$, and $s^{(i)}_{no}$ to be each value after applying i iterations of the reduction, with $n^{(0)}$, $s^{(0)}_{yes}$, and $s^{(0)}_{no}$ set to the initial values.

We will first show that $s^{(i)}_{no} = 2^{\frac{n^\delta}{2^i} - \frac{2^i-1}{2^i}\log 3}$; this is true for $i = 0$, and for larger i we have $s^{(i+1)}_{no} = \sqrt{\frac{s^{(i)}_{no}}{3}} = 2^{\frac{n^\delta}{2^{i+1}} - \frac{2^i-1}{2^{i+1}}\log 3 - \frac{\log 3}{2}} = 2^{\frac{n^\delta}{2^{i+1}} - \frac{2^{i+1}-1}{2^{i+1}}\log 3}$.

Next, we show that $n^{(i)} = 16^i n \prod_{j=1}^{i} [\frac{n^\delta}{2^j} - \frac{2^j-1}{2^j}\log 3]$; via padding, we can increase the number of variables to $n^{(c-1)\delta+1}/2$ at the end. Again, this is true for $i = 0$. For larger i,

$$n^{(i+1)} = 16n^{(i)} \log s_{no}^{(i+1)} = 16^{i+1} n \prod_{j=1}^{i+1} \left[\frac{n^\delta}{2^j} - \frac{2^j-1}{2^j}\log 3\right].$$

Finally, for $s_{yes}^{(i)}$, we show that after i iterations of the Theorem 19 reduction, $s_{yes} = 2^{n^a}$, $s_{no} = 2^{n^\delta}$ would be mapped to at most $s'_{yes} = \exp_2(5n^{\frac{a+(2^i-1)\delta}{2^i}})$. For $i > 0$, assuming $s_{yes}^{(i)} \le \exp_2(5n^{\frac{a+(2^i-1)\delta}{2^i}})$, we have $s_{yes}^{(i+1)}$ is at most

$$\exp_2\left(1 + 5n^{\frac{a+(2^i-1)\delta}{2^i}} + \sqrt{\frac{80}{2^i} n^{\frac{a+(2^i-1)\delta}{2^i}} (n^\delta - \Theta(2^i))} + O(\log n)\right) \le \exp_2\left(5n^{\frac{a+(2^{i+1}-1)\delta}{2^{i+1}}}\right),$$

fixing n_0 sufficiently large. For $i = 0$, note that $n^a < 5n^{\frac{a+(2^0-1)\delta}{2^0}}$.

Step 3.
$\mathsf{AC}^0_{(d+c-1)+1/2}\text{-}\mathtt{GapMCSP}_{n^{(c-1)\delta+1}/2}[\exp_2\left(5n^{\frac{a+(2^{c-1}-1)\delta}{2^{c-1}}}\right), \exp_2(\frac{n^\delta}{2^{c-1}} - \frac{2^{c-1}-1}{2^{c-1}}\log 3)] \le_m^{\mathsf{poly}}$
$\mathsf{AC}^0_{d+c}\text{-}\mathtt{GapMCSP}_{n^{(c-1)\delta+1}}[2^{n^\beta}, 2^{n^\gamma}]$
This follows immediately from Theorem 21, setting

$$s_{yes} = \exp_2\left(5n^{\frac{a+(2^{c-1}-1)\delta}{2^{c-1}}}\right) \text{ and } s_{no} = \exp_2\left(\frac{n^\delta}{2^{c-1}} - \frac{2^{c-1}-1}{2^{c-1}}\log 3\right). \quad ◀$$

4 Constant-Depth Tolerant GapMCSP Reductions

We will show an analogous "hardness lifting" reduction from the GapMCSP problem for average-case circuits of depth d to depth $d+1$.

In this average case setting, instead of applying the machinery of Lemma 16, we can instead make use of the observation that bottom gates of large fan-in are almost always equal to their bias; see Theorem 23. Thus we get smaller gaps on the output side of the reduction, at a small cost to the tolerance parameter.

4.1 Tolerant depth $d+1$ to $d+1/2$ and reverse

▶ **Theorem 23** (Tolerant depth $(d+1) \to (d+1/2)$). *For any $0 \le \epsilon_1, \epsilon_2 < 1/2$, $d \ge 1$, $n \ge 1$, and $s_{yes} < s_{no}$, we have*

$$\widehat{\mathsf{AC}}^0_{d+1}[\epsilon_1, \epsilon_2]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \le_m^{\mathsf{poly}} \widehat{\mathsf{AC}}^0_{d+1/2}[\epsilon_1 + 1/n, \epsilon_2]\text{-}\mathtt{GapMCSP}_n[(s_{yes})^2, s_{no}]$$

with the identity functions as a reduction.

Proof. The "NO"→"NO" case is obvious. For the "YES"→"YES" case, suppose C is a depth $d+1$ circuit of size s_{yes} that disagrees with f on at-most an ϵ_1-fraction of inputs. For each bottom gate of C with fan-in larger than $2\log|C|$, replace the gate with a 1 if it is an OR, or a 0 if it is an AND. Call this new circuit with the replaced gates C'. For a uniformly-random sampled input, any of the replaced gates would disagree with this bit with probability at most $|C|^{-2}$, and so the probability C' disagrees with C on a uniformly-random input is at most $1/|C|$, via a union bound. Since $|C| \ge n$, this is at most $1/n$, and so C' disagrees with f on at most an $(\epsilon_1 + 1/n)$-fraction of inputs. Note that $|C'| \le |C| \le (s_{yes})^2$ and the bottom fan-in of C' is at most $2\log s_{yes} \le \log(s_{yes})^2$, as required. ◀

▶ **Theorem 24** (Tolerant depth $(d + 1/2) \to (d + 1)$). *For any $0 \le \epsilon_1, \epsilon_2 < 1/2$, $d \ge 1$, $n \ge 1$, $s_{yes}, s_{no}, s'_{yes}, s'_{no}$ such that $s_{yes} < s_{no}$, we have, via the the identity functions as a reduction,*

$$\widehat{\mathsf{AC}}^0_{d+1/2}[\epsilon_1, \epsilon_2 + 1/n]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \le_m^{\mathsf{poly}} \widehat{\mathsf{AC}}^0_{d+1}[\epsilon_1, \epsilon_2]\text{-}\mathtt{GapMCSP}_n[s_{yes}, \sqrt{s_{no}}].$$

Proof. The "YES"→"YES" case is obvious. For the "NO"→"NO" case, let C' be depth-$(d{+}1)$ circuit of size at most $s'_{no} = \sqrt{s_{no}}$ that ϵ_2-approximates f. As in the proof of Theorem 23 above, we replace by constants all bottom gates of C' that have fan-in larger than $2\log|C'|$, getting a new circuit C that computes f on all but at most $\epsilon_2 + (1/n)$ fraction of inputs. The size of C is at most $s'_{no} \le s_{no}$, and the bottom fan-in is at most $2\log s_{no}^{1/2} = \log s_{no}$, as required. ◀

4.2 Tolerant depth $d + 1/2$ to $(d + 1) + 1/2$

▶ **Theorem 25** (Tolerant depth $(d + 1/2) \to ((d + 1) + 1/2)$). *For any $d \ge 1$, $n \ge 1$, $0 \le \epsilon_1, \epsilon_2 < 1/2$, s_{yes}, s'_{yes}, s'_{no}, and s_{no} such that $s_{yes} < s_{no}$, $s'_{yes} < s'_{no}$, $s_{no} \ge 3(s'_{no})^2(\epsilon_2 n + 1)$ and $s'_{yes} \ge 2(s_{yes})2^{\sqrt{16(\log s_{yes})(\log s'_{no})}}\sqrt{16(\log s_{yes})(\log s'_{no})}$, we have*

$$\widehat{\mathsf{AC}}^0_{d+1/2}[\epsilon_1, \epsilon_2 + 1/n]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \le_m^{\mathsf{qpoly}} \widehat{\mathsf{AC}}^0_{(d+1)+1/2}[\epsilon_1, \epsilon_2]\text{-}\mathtt{GapMCSP}_{n'}[s'_{yes}, s'_{no}],$$

where $n' = 16n\log s'_{no} \le O(n^2)$.

Proof. We shall use the same reduction as in Theorem 19, outputting $f \circ \oplus_\ell$ on input f, where $\ell = 16\log s'_{no}$.

NO → NO. Let C' be a depth-$(d+2)$ circuit of size s'_{no} and bottom fan-in at most $\log s'_{no}$ that ϵ_2-approximates $f \circ \oplus_\ell$. We shall hit C' with a blockwise random restriction, as before. Here, we simultaneously require that $C' \upharpoonright \rho$ simplifies to a depth-$(d{+}1)$ circuit with bounded bottom fan-in, and that its truth table is $(\epsilon_2 + 1/n)$-close to (some fixed shift of) f.

For any $x \in \{0,1\}^n$ and a blockwise restriction ρ, we denote by $\langle x, \rho\rangle$ the $(n \cdot \ell)$-tuple of bits obtained by placing x in the star positions of ρ. Clearly, picking x and ρ uniformly at random results in $\langle x, \rho\rangle$ being the uniform distribution on $\{0,1\}^{n\cdot\ell}$. By our assumption on C', we have $\mathsf{Exp}_{x,\rho}[C'(\langle x,\rho\rangle) \ne (f \circ \oplus_\ell)(\langle x,\rho\rangle)] \le \epsilon_2$. By Markov's Inequality,

$$\Pr_\rho\left[\mathsf{Exp}_x\left[C'(\langle x,\rho\rangle) \ne (f \circ \oplus_\ell)(\langle x,\rho\rangle)\right] > \epsilon_2 + \frac{1}{n}\right] < \frac{\epsilon_2}{\epsilon_2 + (1/n)}.$$

Hence, with probability at least $(\epsilon_2 \cdot n + 1)^{-1}$, for a randomly chosen blockwise restriction ρ

$$\begin{aligned}\mathsf{Exp}_x\left[C'(\langle x,\rho\rangle) \ne (f \circ \oplus_\ell)(\langle x,\rho\rangle)\right] &= \mathsf{Exp}_x\left[C' \upharpoonright_\rho (x) \ne (f \circ \oplus_\ell) \upharpoonright_\rho (x)\right] \\ &= \mathsf{Exp}_x\left[C' \upharpoonright_\rho (x) \ne f(x \oplus b^\rho)\right] \le \epsilon_2 + \frac{1}{n},\end{aligned}$$

for $b^\rho = b_1 \ldots b_n \in \{0,1\}^n$ such that b_i is the parity of assigned values in the ith block of ρ.

So, if $C' \upharpoonright_\rho$ fails to simplify with probability less than $(\epsilon_2 \cdot n{+}1)^{-1}$, then we are guaranteed there is some ρ such that $C' \upharpoonright_\rho (x)$ agrees with $f(x \oplus b^\rho)$, a shift of f, on all but at most $(\epsilon_2 + (1/n))$-fraction of inputs $x \in \{0,1\}^n$, and is a depth-$(d + 1)$ circuit with bounded bottom fan-in.

By the Blockwise Switching Lemma (Lemma 10), the probability that $C' \upharpoonright \rho$ fails to simplify to depth $(d + 1)$ circuit with bottom fan-in at most k is at most $s'_{no}\left(\frac{8\log s'_{no}}{\ell}\right)^k = 2^{\log s'_{no} - k}$, which is less than $(\epsilon_2 \cdot n + 1)^{-1}$ if we choose $k = \log(2s'_{no}(1 + \epsilon_2 n))$.

Thus, there must exist a blockwise restriction ρ such that $C' \upharpoonright_\rho$ is simplified and agrees with $f(x \oplus b^\rho)$ on all but at most $(\epsilon_2 + (1/n))$ fraction of inputs. We have that $C' \upharpoonright_\rho$ is of size at most $s'_{no}(1+2^k) \le s'_{no}(1+2s'_{no}(1+\epsilon_2 n)) \le 3(s'_{no})^2(1+\epsilon_2 n) \le s_{no}$. Also, the bottom fan-in is at most $k \le \log s_{no}$ for our choice of k. Then the circuit $C(x) = C' \upharpoonright_\rho (x \oplus b^\rho)$ agrees with $f(x)$ on all but at most $(\epsilon_2 + (1/n))$ fraction of inputs, and C has depth $(d+1)$, size at most s_{no}, and bottom fan-in at most $\log s_{no}$, as required.

YES $\rightarrow$ YES. Suppose f is ϵ_1-approximated by a depth-$(d+1)$ circuit C with size s_{yes} and bottom fan-in $\log s_{yes}$. Let g be the Boolean function computed by C. Using the same techniques as in the "YES$\rightarrow$YES" case analysis in the proof of Theorem 19, we construct a depth-$(d+1)$ circuit C' computing $g \circ \oplus_\ell$, with size at most s'_{yes} and bottom fan-in at most $\log s'_{yes}$.

We will argue that C' computes $f \circ \oplus_\ell$ on all but at most ϵ_1 fraction of inputs. Indeed, since the parity of a uniformly random string of bits is a uniformly random bit, we get that

$$\Pr_{z \in \{0,1\}^{n\ell}}[(f \circ \oplus_\ell)(z) = (g \circ \oplus_\ell)(z)] = \Pr_{x \in \{0,1\}^n}[f(x) = g(x)],$$

which is at most ϵ_1 by our assumption. This concludes the proof. ◀

4.3 Combining the steps: Tolerant depth $d+1$ to $d+2$

Using the above reductions, we can obtain a reduction from tolerant depth $d+1$ gap-MCSP to tolerant depth $d+2$ gap-MCSP. Extending this to depth $d+c$ can be done via repeatedly composing this reduction with itself.

▶ **Corollary 26.** *For any $d \ge 1, 0 \le \epsilon_1, \epsilon_2 < 1/2, s_{yes}, s_{no}, s'_{yes}, s'_{no}$ where $s_{no} \ge (2\epsilon n+1){s'_{no}}^4$ and $s'_{yes} \ge 2(s_{yes})^2 \cdot 2^{\sqrt{16\log({s_{yes}}^2)\log({s'_{no}}^2)}}\sqrt{16\log({s_{yes}}^2)\log({s'_{no}}^2)}$, we have*

$$\widehat{\mathsf{AC}}^0_{d+1}[\epsilon_1, \epsilon_2 + \frac{2}{n}]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}] \;\le_m^{\mathsf{qpoly}}\; \widehat{\mathsf{AC}}^0_{d+2}[\epsilon_1 + \frac{1}{n}, \epsilon_2]\text{-}\mathtt{GapMCSP}_{32n\log s'_{no}}[s'_{yes}, s'_{no}].$$

Proof. We obtain the desired reduction by composing the reductions from Theorems 23, 25, and 24. Using $\langle \epsilon_1, \epsilon_2, s_{yes}, s_{no}\rangle_{d,n}$ as a shorthand for $\widehat{\mathsf{AC}}^0_d[\epsilon_1, \epsilon_2]\text{-}\mathtt{GapMCSP}_n[s_{yes}, s_{no}]$, the reductions operate as follows:

$$\begin{aligned}
\langle \epsilon_1, \epsilon_2 + \frac{2}{n}, s_{yes}, s_{no}\rangle_{d+1,n} &\le_m^{\mathsf{poly}} \langle \epsilon_1 + \frac{1}{n}, \epsilon_2 + \frac{2}{n}, (s_{yes})^2, s_{no}\rangle_{d+1/2,n} && \text{Theorem 23}\\
&\le_m^{\mathsf{qpoly}} \langle \epsilon_1 + \frac{1}{n}, \epsilon_2 + \frac{1}{n}, s'_{yes}, (s'_{no})^2\rangle_{d+1+1/2,32n\log s'_{no}} && \text{Theorem 25}\\
&\le_m^{\mathsf{poly}} \langle \epsilon_1 + \frac{1}{n}, \epsilon_2, s'_{yes}, s'_{no}\rangle_{d+2,32n\log s'_{no}} && \text{Theorem 24}
\end{aligned}$$ ◀

5 Barriers to More Efficient Natural Reductions

Our reductions are deterministic, many-one, and "simple" in the original size parameter. However, they require quasi-polynomial time. Here, we give evidence that improving such "nice" reductions to run in polynomial time for the *exact* `MCSP` is difficult: such reductions would immediately give breakthrough circuit lower bounds or non-trivial `MCSP` algorithms, and either outcome seems like dramatic progress.[2] To begin, observe that every reduction we present is qpoly-Natural in the following sense.

[2] Similar arguments apply to the gap-versions of the problem that we study above, but we argue about the exact version here to facilitate exposition.

▶ **Definition 27** (Natural Reductions between Parametric Problems). *Let A and B be* parametric problems, *that is, inputs are of the form: $\{\langle x, s\rangle : x \in \{0,1\}^n,\ s \in \mathbb{N}\}$. We call a* parametric reduction $R = \langle R_I, R_P\rangle$ *where R_I outputs instances and R_P outputs parameters,* $t(\cdot)$-natural *if it is:*

- ***Parametric Many-one:*** $\langle x, s\rangle \in A \iff \langle R_I(x,s),\ R_P(x,s)\rangle \in B$
- ***Parameter-Value Uniform:*** *$R_P(x,s)$ depends* **only** *on the size of the input and value of the parameter; we will treat R_P as a function from $\mathbb{N} \times \mathbb{N}$ in this case.*
- ***$t(\cdot)$-Efficient:*** *The combined runtime of R_I and R_P is bounded by $t(|x|, s)$.*

A natural reduction R from Λ-`MCSP` to Γ-`MCSP` is many-one, so a Λ-`MCSP` algorithm follows by brute-force search through Γ-circuits, and Λ-to-Γ lifting follows by mapping a Λ-hard function h through R. This gives the next two lemmas. Kabanets and Cai used the same reasoning to prove that NP-hardness of `MCSP` under poly-time natural reductions would imply breakthrough circuit lower bounds (Theorem 15 of [20]). Removing NP-hardness from the picture, we instead obtain the following:

▶ **Lemma 28** (Black-Box `MCSP` Algorithms from Natural `MCSP`-Redux). *If there is a* poly*-Natural Reduction from Λ-`MCSP` to Γ-`MCSP`, then there is a fixed constant $k \in \mathbb{N}$ such that $\Lambda\text{-MCSP}_n$* $\in \mathsf{TIME}[\mathsf{poly}(nk) \times \Gamma\text{-count}(R_P(2^n, s))]$

Proof. Fix a reasonable encoding of Γ-circuits that admits efficient evaluation. Then write Γ-count(s) for the total number of circuits so encoded that witness Γ-measure at most s. On input (f, s) to $\Lambda\text{-MCSP}_n$ we first run (f, s) through the natural reduction R to obtain (f', s'). Just as above, because R is poly-time, there is a fixed k such that $t(n) = 2^{kn}$. This means $|f'| \le 2^{kn}$, so we obtain an instance of Γ-`MCSP` with new size parameter $s' = R_P(2^n, s)$ on at most kn input variables.

Then, because R is parametric many-one, a (yes, no)-instance of $\Lambda\text{-MCSP}_n$ becomes a (yes, no)-instance of $\Gamma\text{-MCSP}_{kn}$ (respectively). So, we can solve the resulting instance of Γ-MCSP by brute-force search over the set of all s'-measure-witnessing Γ-circuits, and answer accordingly. We must evaluate a s'-size Γ-circuit on $\le kn$ bits at most Γ-count(s') times. This takes $\mathsf{poly}(nk) \cdot \Gamma\text{-count}(s')$ time in total. ◀

Lifting begins with pre-existing lower bounds for Λ, which we formalize below. Many concrete circuit lower bounds are *far more* explicit, but this weak notion will suffice for lifting via natural and efficient inter-`MCSP` reductions.

▶ **Definition 29** (Explicit Complexity Lower Bounds). *Let $H = \{h_n\}_{n\in\mathbb{N}}$ be a sequence of Boolean functions in* E*, and let $s_\Lambda : \mathbb{N} \to \mathbb{N}$ be a function in* FP*. We call the pair $\langle H, s_\Lambda\rangle$ an* explicit Λ-complexity lower bound *if $\forall n\ \Lambda(h_n) > s_\Lambda(n)$.*

▶ **Lemma 30** (Black-Box Lifting from Natural MCSP-Redux). *Let $\langle H, s\rangle$ be a Λ-complexity lower bound. If there is a* poly*-Natural Reduction R from Λ-`MCSP` to Γ-`MCSP`, then there exists a constant k and sequence of m-input Boolean functions H' such that $\langle H', R_P(2^{m/k}, s(m/k))\rangle$ is an explicit Γ-complexity lower bound.*

Proof. Fix an explicit Λ-complexity lower bound $\langle H, s\rangle$ and poly-natural reduction $R = \langle R_I, R_P\rangle$ from Λ-`MCSP` to Γ-`MCSP`. Now run the reduction: let H' be the sequence $h'_n = R_I(h_n, s(n))$ and let $s'(n) = R_P(h_n, s(n))$. We know $(h_n, s(n)) \notin \Lambda$-`MCSP` by the hardness assumption about H. Then, because R is parametric many-one, $(h'_n, s'(n)) \notin \Gamma$-`MCSP` and thus $\Gamma(h'_n) > s'(n)$. To make this explicit, we bound the runtime of answering queries according to h' on inputs x of m bits. This amounts to re-indexing the sequence H' to ensure that a Γ-hard function is defined everywhere and computable in E.

First, because R is poly-time, there is a fixed k such that $t(n) = 2^{kn}$. This means $|h'_n| \leq 2^{kn}$, so we send each input length n through the reduction to a new input length of at most kn. We evaluate h at m/k input bits and pad to fill in the gaps. Propagating this padded sequence of functions through the parameter-map R_P, we obtained the claimed Γ-complexity lower bound. ◀

5.1 Efficient Natural Reductions Between AC^0_d-, AC^0_{d+1}-MCSP: Win/Win

Notice how both applications of poly-Natural reductions depend quantitatively on R_P, the size parameter of the reduction. For lifting, we want $R_P(\cdot)$ *large enough* to improve the best known Γ-complexity lower bound by starting with a stronger lower bound for Λ. For solving Λ-MCSP by brute-force on Γ-MCSP, we want $R_P(\cdot)$ *small enough* such that searching all relevant Γ-circuits is faster than trivial brute-force over all relevant Λ-circuits. This observation suggests a case analysis of the function R_P, to obtain either a non-trivial MCSP algorithm or improved circuit lower bounds. For poly-Natural reductions from AC^0_d-MCSP to AC^0_{d+1}-MCSP, such a win/win argument succeeds. Informally, we have the following:

▶ **Theorem 31** (poly-Natural MCSP Reduction Win/Win). *Suppose there is a* poly*-Natural reduction from* AC^0_d-MCSP *to* $\mathsf{AC}^0_{d'}$-MCSP, *for* $d' > d$. *Then, either:*
- *There is a surprisingly fast algorithm for* AC^0_d-MCSP, *or*
- *There are breakthrough explicit circuit lower bounds against* $\mathsf{AC}^0_{d'}[2^{\Omega(n^{1/d})}]$ *for* $d < d'$ *!*

We spend the remainder of this section formalizing and proving variations on the above.

5.2 Quantitative Consequences of a Hardness Hypothesis for MCSP

We first formulate an appropriate hypothesis about the hardness of MCSP.

▶ **Definition 32** (Weak Exponential Time Hypothesis (WETH) for Λ-MCSP). *There exists an* $\epsilon > 0$ *such that for all "nice" size functions* $s(n)$, $\Lambda\text{-MCSP}_n[s(n)] \notin \mathsf{TIME}[2^{s(n)^\epsilon}]$.

For the general MCSP (when Λ is the class of unrestricted Boolean circuits), it can be shown that the WETH for MCSP is implied by the cryptographic conjecture that exponentially-strong one-way functions exist (using the ideas of [28, 20, 2]). One can also show that if WETH for general MCSP is false, then $\mathsf{NEXP} \not\subset \mathsf{P/poly}$ (using the ideas of [19]). For every $d \geq 2$, the WETH for AC^0_d-MCSP is also reasonable to assume, although we don't seem to have any strong evidence to support it yet (see [3] for some cryptographic hardness of AC^0_d-MCSP for large d).

Under this hypothesis, we establish barriers to giving poly-Natural reductions from AC^0_d-MCSP to AC^0_{d+c}-MCSP. We begin by recalling the best-known AC^0_d circuit lower bounds.

▶ **Theorem 33** (Håstad [13]). *Any depth* $(d+1)$ *alternating circuit computing* $\oplus_n$ *requires* $2^{\Omega(n^{1/d})}$ *gates. Furthermore, this bound is clearly* explicit *as in Definition 29.*

▶ **Theorem 34.** *Suppose there is a* poly*-Natural reduction from* AC^0_d-MCSP *to* $\mathsf{AC}^0_{d'}$-MCSP, *for* $d' > d$. *Then, either:*
- *The WETH for* AC^0_d-MCSP *is* ***false***, *or*
- *There is an explicit circuit lower bound with* $s(n) = 2^{\Omega(n^{1/(d-1)})}$ *against* $\mathsf{AC}^0_{d'}$.

Proof. Assume such a poly-Natural reduction $R = \langle R_I, R_P \rangle$ exists, with run-time 2^{kn}. We reason by cases on bounds for R_P.

Suppose R_P is small. That is, $\forall\epsilon. R_P(2^n, s(n)) < s(n)^\epsilon$. Substituting into the black-box MCSP algorithm above, we have that $\forall\epsilon.\mathsf{AC}^0_d\text{-}\mathtt{MCSP}_n \in \mathsf{TIME}[\mathsf{poly}(nk) \times \mathsf{AC}^0_{d'}\text{-}\,\mathrm{count}(s(n)^\epsilon)] \in \mathsf{TIME}[2^{s(n)^{2\epsilon}}]$, where the first inclusion is by Lemma 28, and second by counting $\mathsf{AC}^0_{d'}$ circuits. This contradicts the MCSP-WETH for AC^0_d.

Suppose R_P is large. That is, $\exists\epsilon. R_P(2^n, s(n)) > s(n)^\epsilon$. Lifting $\oplus$ through R we have that there is an explicit sequence of Boolean functions H on m-bit inputs such that we have the following explicit $\mathsf{AC}^0_{d'}$-complexity bounds: $R_P(2^{m/k}, s(m/k)) > s(m/k)^\epsilon > 2^{\Omega(m^{1/(d-1)})}$. Here, the lower bound is by Lemma 30, the first inequality by size assumption about R_P, and the last by application of Håstad's bound. ◀

When $d' > d$, the lower-bound case above would be a breakthrough in circuit complexity.

▶ **Corollary 35** (Breakthrough Circuit Lower Bounds for Alternating Constant-Depth). *Suppose the WETH for* AC^0_d-$\mathtt{MCSP}$ *holds, for every* $d \geq 2$*. Then, if* $\forall d > d_0$ *we have a* poly*-Natural reduction* R_d *from* AC^0_d-$\mathtt{MCSP}$ *to* $\mathsf{AC}^0_{(d+1)}$-$\mathtt{MCSP}$*, then there is a fixed constant* α *such that, for each depth* $d > d_0$*, there is a Boolean function* $f^d \in E$ *such that any depth-*d *alternating circuit computing* f^d_n *requires* $2^{\Omega_d(n^\alpha)}$ *gates.*

Proof. Fix any constant $d > d_0$. We first compose R_d with itself sufficiently many times to obtain a many-one reduction R'_d all the way from $\mathsf{AC}^0_{d_0}$-$\mathtt{MCSP}$ to AC^0_d-$\mathtt{MCSP}$. Observe that R'_d remains poly-Natural, because all the polynomial resource bounds are closed under a constant number of compositions – though the leading constant exponent of runtime for R'_d certainly increases proportional to the gap between d and d_0; this is precisely what is hidden by Ω_d in the bound. To conclude, we apply black box lifting (Lemma 30) to the composed poly-Natural reduction R'_d, with Håstad's lower bound for $\oplus$ at depth d_0, getting $\alpha = 1/d_0$ in the theorem. ◀

Combining with a simulation of shallow formulas by constant-depth circuits, we get

▶ **Lemma 36** (Folklore). *Any sequence* f_n *of Boolean functions on* n *inputs computable by formulas of depth* $c\log(n)$ *is computable by depth-*d *alternating* circuits *of size* $2^d \times 2^{n^{(c/d)}}$.

▶ **Theorem 37** (Breakthrough Circuit Lower Bounds for Formulas). *Suppose the WETH for* AC^0_d-$\mathtt{MCSP}$ *holds, for every* $d \geq 2$*. Then, if* $\forall d > d_0$ *we have a* poly*-Natural reduction* R_d *from* AC^0_d-$\mathtt{MCSP}$ *to* $\mathsf{AC}^0_{(d+1)}$-$\mathtt{MCSP}$*, for every fixed* k *there exists* f^k *a sequence of Boolean functions in* E*, such that* f^k *does not have size-*n^k *formulas.*

Proof. Fix constant k, and let $c \in \mathbb{N}$ be the leading constant that results from re-balancing an arbitrary n^k-size formula to log-depth. Any function computed by such a formula will have – for every d – AC^0_d circuits of size $\approx 2^{n^{c/d}}$ by Lemma 36. Therefore, if we choose d such that $1/d_0 > c/d$, the size bound that results from lifting $\oplus$ through iterated composition of R_d exceeds the constant-depth simulation-size of any n^k-size formula. The rest of this argument is identical to the proof of Corollary 35 above. ◀

6 Open Questions

One obvious question is whether one can show the NP-completeness of AC^0_d-MCSP for any constant depth $d \geq 3$ by proving that depth 3 circuit complexity is NP-hard to approximate to within the factors needed by our lifting theorems? Note that while we have hardness of approximation result for DNFs [23, 3], the (almost tight) approximation gap there is not strong enough for our lifting theorems to apply. For depth-3 circuits, on the other hand, there are no known hardness of approximation results.

Another natural question is to tighten the gap (the approximation factor) of our lifting theorems. Finally, can one provide more evidence supporting the Weak Exponential Time Hypothesis for AC^0_d-MCSP?

References

1 Eric Allender. The new complexity landscape around circuit minimization. In Alberto Leporati, Carlos Martín-Vide, Dana Shapira, and Claudio Zandron, editors, *Language and Automata Theory and Applications*, pages 3–16, Cham, 2020. Springer International Publishing.

2 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

3 Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing Disjunctive Normal Form Formulas and AC^0 Circuits Given a Truth Table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.

4 Eric Allender and Shuichi Hirahara. New insights on the (non-)hardness of circuit minimization and related problems. *ACM Transactions on Computation Theory (ToCT)*, 11(4):1–27, 2019.

5 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. In *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, pages 21–33, 2015. doi:10.4230/LIPIcs.STACS.2015.21.

6 Alexander E. Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of π-schemes. *Moscow Univ. Math. Bull.*, 42:63—-66, 1987.

7 Paul Beame. A switching lemma primer, 1994.

8 Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

9 Vàclav Chvátal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4(3):233–235, 1979. doi:10.1287/moor.4.3.233.

10 Susanna F. de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 24–30. IEEE, 2020.

11 Vitaly Feldman. Hardness of approximate two-level logic minimization and PAC learning with membership queries. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, page 363–372, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1132516.1132569.

12 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM Journal on Computing*, 47(6):2435–2450, 2018.

13 Johan Håstad. Almost optimal lower bounds for small depth circuits. In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20. ACM, 1986. doi:10.1145/12130.12132.

14 Johan Håstad, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. An average-case depth hierarchy theorem for boolean circuits. *J. ACM*, 64(5):35:1–35:27, 2017. doi:10.1145/3095799.

15 Shuichi Hirahara, Igor C. Oliveira, and Rahul Santhanam. NP-hardness of minimum circuit size problem for OR-AND-MOD circuits. In *33rd Computational Complexity Conference (CCC 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

16 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *31st Conference on Computational Complexity, CCC*, pages 18:1–18:20, 2016. doi:10.4230/LIPIcs.CCC.2016.18.

17 John M. Hitchcock and A. Pavan. On the NP-completeness of the minimum circuit size problem. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, pages 236–245, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.236.

18 Rahul Ilango. Constant depth formula and partial function versions of MCSP are hard. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 424–433. IEEE, 2020.

19 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *J. of Computer and System Sciences*, 65(4):672–694, 2002.

20 Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.

21 Richard M. Karp, F. E. McFarlin, J. P. Roth, and J. R. Wilts. A computer program for the synthesis of combinational switching circuits. In *2nd Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1961)*, pages 182–194, 1961. doi:10.1109/FOCS.1961.1.

22 Subhash Khot and Rishi Saket. Hardness of minimizing and learning DNF expressions. In *2008 IEEE 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 231–240, Los Alamitos, CA, USA, October 2008. IEEE Computer Society. doi:10.1109/FOCS.2008.37.

23 William J. Masek. Some NP-complete set covering problems, 1979.

24 Cody D. Murray and Ryan R. Williams. On the (non) NP-hardness of computing circuit complexity. In *30th Conference on Computational Complexity, CCC*, pages 365–380, 2015. doi:10.4230/LIPIcs.CCC.2015.365.

25 Toniann Pitassi and Robert Robere. Lifting nullstellensatz to monotone span programs over any field. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1207–1219, 2018.

26 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. In *Proceedings 38th Annual Symposium on Foundations of Computer Science*, pages 234–243. IEEE, 1997.

27 Alexander A. Razborov. Bounded arithmetic and lower bounds in boolean complexity. In Peter Clote and Jeffrey B. Remmel, editors, *Feasible Mathematics II*, pages 344–386, Boston, MA, 1995. Birkhäuser Boston.

28 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. of Computer and System Sciences*, 55(1):24–35, 1997.

29 Michael Saks and Rahul Santhanam. Circuit lower bounds from NP-hardness of MCSP under Turing reductions. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

30 Boris A. Trakhtenbrot. A Survey of Russian Approaches to Perebor (Brute-Force Searches) Algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984. doi:10.1109/MAHC.1984.10036.