Electronic Colloquium on Computational Complexity, Report No. 21 (2019)

$\mathsf{AC}^0[p]$ Lower Bounds and NP-Hardness for Variants of MCSP

Rahul Ilango
Rutgers University, Piscataway, USA
rahul.ilango@rutgers.edu

Abstract

The Minimum Circuit Size Problem (MCSP) asks whether a (given) Boolean function has a circuit of at most a (given) size. Despite over a half-century of study, we know relatively little about the computational complexity of MCSP. We do know that questions about the complexity of MCSP have significant ramifications on longstanding open problems. In a recent development, Golovnev *et al.* [11] improve the status of unconditional lower bounds for MCSP, showing that MCSP $\notin \mathsf{AC}^0[p]$ for any prime p. While their results generalize to most "typical" circuit classes, it fails to generalize to the circuit minimization problem for depth-d formulas, denoted (AC^0_d)-MCSP. In particular, their result relies on a Lipchitz hypothesis that is unknown (and possibly false) in the case of (AC^0_d)-MCSP. Despite this, we show that (AC^0_d)-MCSP $\notin \mathsf{AC}^0[p]$ by proving even the failure of the Lipchitzness for AC^0_d formulas implies that MAJORITY $\leq_{tt}^{\mathsf{AC}^0}$ (AC^0_d)-MCSP. Somewhat remarkably, our proof (in the case of non-Lipchitzness) uses completely different techniques than [11]. To our knowledge, this is the first MCSP reduction that uses modular properties of a function's circuit complexity.

We also define MOCSP, an oracle version of MCSP that takes as input a Boolean function f, a size threshold s, and oracle Boolean functions $f_1, \ldots, f_t$, and determines whether there is an oracle circuit of size at most s that computes f when given access to $f_1, \ldots, f_t$. We prove that MOCSP is NP-complete under non-uniform AC^0 many-one reductions as well as (uniform) ZPP truth table reductions. We also observe that improving this ZPP reduction to a deterministic polynomial-time reduction requires showing EXP $\neq$ ZPP (using theorems of Hitchcock and Pavan [17] and Murray and Williams [22]). Optimistically, these MOCSP results could be a first step towards NP-hardness results for MCSP. At the very least, we believe MOCSP clarifies the barriers towards proving hardness for MCSP and provides a useful "testing ground" for questions about MCSP.

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

Keywords and phrases Minimum Circuit Size Problem, reductions, NP-completeness, circuit lower bounds

Acknowledgements I would like to give a special thanks to Eric Allender for innumerable suggestions and perspectives during all stages of this work. To name just a single example, one of his suggestions led me to improve a PARITY-reduction to the presented MAJORITY-reduction for (AC^0_d)-MCSP. I would also like to thank Abhishek Bhrushundi and Aditi Dudeja for their help in results about constant depth formulas that lead to this paper. I am grateful to Ryan Williams for asking interesting questions and helping to improve the exposition of the paper. Finally, I thank Harry Buhrman, Lance Fortnow, Aditya Potukuchi, and Michael Saks for answering my questions and engaging in many useful discussions.

1 Introduction

The Minimum Circuit Size Problem (MCSP) takes as input a Boolean function f (represented by its truth table) and a size parameter s and asks if there is a circuit of size at most s computing f. Study of this problem began in the 1950s by complexity theorists in the Soviet Union [30], where MCSP was of such great interest that Levin is said to have delayed

ISSN 1433-8092

publishing his initial NP-completeness results in hope of showing that MCSP is NP-complete.[1] Interest in MCSP was revitalized when Kabanets and Cai [19] connected the problem with the natural proofs framework of Razborov and Rudich [27]. Since then, MCSP has been the subject of intense research. We begin by reviewing some of this work.

1.1 Known lower bounds, hardness, and non-hardness for MCSP

It is easy to see that MCSP is in NP (the circuit of size at most s can be used as a witness), but, despite work by numerous researchers, the exact complexity of MCSP remains unknown.

Lower bounds and hardness results. We believe MCSP is not easy to compute. Kabanets and Cai [19] show that $\mathsf{MCSP} \notin \mathsf{P}$ conditioned on a widely-believed cryptographic hypothesis, and Allender and Das [2] show that MCSP is hard for SZK under BPP-Turing reductions.

Unconditionally, we know lower bounds against MCSP for restricted classes of circuits. Hirahara and Santhanam [15] show that MCSP requires nearly quadratic sized DeMorgan formulas, and Allender *et al.* [1] prove that $\mathsf{MCSP} \notin \mathsf{AC}^0$. In a recent paper, Golovnev *et al.* [11] improve the latter result, showing that MCSP requires exponential-sized $\mathsf{AC}^0[p]$ circuits by proving $\mathsf{MAJORITY} \in (\mathsf{AC}^0)^{\mathsf{MCSP}}$. The MAJORITY hardness result of [11] generalizes to the circuit minimization problem for many circuit classes, however, the techniques fail in the case of constant depth formulas.

Under weak reductions, we know MCSP is hard for some subclasses of P. Oliveira and Santhanam [25] prove that MCSP is hard for DET under TC^0 truth table reductions, and Golovnev *et al.* [11] use the results of [25] to show that $\mathsf{NC}^1 \subseteq (\mathsf{AC}^0)^{\mathsf{MCSP}}$. Surprisingly, we know stronger results for the "program" variant of MCSP, MKTP. Allender and Hirahara [3] show that MKTP is hard for DET under NC^0 many-one reductions, and Hirahara and Santhanam [15] show average-case lower bounds for MKTP against $\mathsf{AC}^0[p]$.

The most natural question is whether MCSP is NP-complete. As of yet, we have not managed to uncover even strong supporting evidence for, or against, MCSP being NP-complete. We do know that the circuit minimization problem is NP-complete for some restricted classes of circuits: DNF circuits by Masek [20] and $\mathsf{OR} \circ \mathsf{AND} \circ \mathsf{MOD}_m$ circuits by Hirahara, Oliveira, and Santhanam [14]. Impagliazzo, Kabanets, and Volkovich [18] show that if there exist Indistinguishability Obfuscators against randomized polynomial-time algorithms, then $\mathsf{MCSP} \in \mathsf{ZPP} \iff \mathsf{NP} = \mathsf{ZPP}$.

Known non-hardness results. The unconditional non-hardness results for MCSP rule out NP-hardness under certain types of reductions. For example, Hirahara and Watanabe [16] show that "oracle-independent reductions" cannot show that MCSP is hard for either a class larger than P under polynomial-time Turing reductions or a class larger than $\mathsf{AM} \cap \mathsf{coAM}$ under BPP reductions with one query to MCSP. Moreover, while most NP-complete problems are complete under rather weak reductions such as $\mathsf{TIME}[n^{o(1)}]$ or AC^0 many-one reductions, Murray and Williams [22] prove that MCSP is not NP-hard under $\mathsf{TIME}[n^{.49}]$ reductions, and Allender, Ilango, and Vafa [5] show that a super-linear approximations of MCSP cannot be NP-hard under even non-uniform AC^0 many-one reductions.

Conditioned on a widely-believed cryptographic hypothesis, Allender and Hirahara [3] show that a very weak approximation of MCSP is NP-intermediate.

[1] [6] cites a personal communication from Levin regarding this story.

1.2 Implications of lower bounds and hardness for MCSP

While we have not managed to establish the complexity of MCSP, a series of works, beginning with Kabanets and Cai [19], connect the computational complexity of MCSP and its variants to longstanding open questions in the field.

Separations of complexity classes. Several works ([17], [22], [2], [19]) show that MCSP being NP-hard, under various notions of reducibility, implies unknown class separations. For example, Hitchcock and Pavan [17] and Murray and Williams [22] show that if MCSP is NP-hard under polynomial-time truth-table reductions, then $\mathsf{ZPP} \neq \mathsf{EXP}$, a major open problem.[2]

Worst-case versus average-case complexity for NP. Using tools developed by Nisan and Wigderson [23] and Carmosino, Impagliazzo, Kabanets, and Kolokova [9], Hirahara [13] gives a "worst-case to average-case" reduction for NP conditioned on a certain approximation to MCSP being NP-hard. Thus, if one could show this approximation to MCSP is NP-hard, the worst-case and average-case complexity of NP would be equivalent.

Circuit Lower Bounds. Recent work by Oliveira, Pich, and Santhanam ([26] and [24])[3] explores a phenomenon they term "hardness magnification," whereby even weak circuit lower bounds on certain computational problems imply strong lower bounds on other problems. For example, [26] shows that if MCSP cannot be solved on average with no error by linear-size formulas, then NP does not have polynomial-size formulas. [24] shows that if a certain approximation to MCSP cannot be computed by circuits of size $n^{1+\epsilon}$, then NP does not have polynomial-sized circuits.

1.3 Our Contributions

In this work, we focus on hardness results for variants of MCSP, in particular establishing an $\mathsf{AC}^0[p]$ lower bound and an NP-hardness result.

MAJORITY-hardness for (AC^0_d)-MCSP

As mentioned previously, Golovnev *et al.* [11] proves that $\mathsf{MAJORITY} \in (\mathsf{AC}^0)^{\mathsf{MCSP}}$. Using similar techniques, they also show that, for restricted classes of circuits $\mathcal{C}$ such as formulas and constant depth circuits, the $\mathcal{C}$-circuit minimization problem, denoted $(\mathcal{C})$-MCSP, is hard for MAJORITY under AC^0 reductions. For these MAJORITY reductions to work, [11] requires that the size of the minimum $\mathcal{C}$-circuit on truth tables of length n is roughly $(n^{.49})$-Lipchitz.

This Lipchitzness hypothesis is unknown (and perhaps even false) in the class of depth-d *formulas*, which we denote AC^0_d.[4] Despite this, we prove MAJORITY-hardness for (AC^0_d)-MCSP by giving a MAJORITY reduction that works in the case that Lipchitzness fails. Applying the lower bounds of Razborov [27] and Smolensky [29] then gives an $\mathsf{AC}^0[p]$ lower bound for (AC^0_d)-MCSP.

---

2 [22] only shows the result under many-one reductions, but their techniques easily generalize to the truth table case. [17] explicitly proves the truth table result using a different approach than [22].

3 Pich is an author on [24] but not [26]

4 We will always use the notation AC^0_d to refer to depth-d *formulas* and never depth-d *circuits*.

▶ **Theorem 1.1.** *Let $d \geq 2$. Then* $\mathsf{MAJORITY} \leq_{tt}^{\mathsf{AC}^0} (\mathsf{AC}_d^0)\text{-}\mathsf{MCSP}$. *Consequently,* $(\mathsf{AC}_d^0)\text{-}\mathsf{MCSP} \notin \mathsf{AC}^0[p]$ *for any prime p.*

Remarkably, the techniques used for this MAJORITY reduction (in the case of non-Lipchitzness) are entirely different than the ones used by [11] for general MCSP. Indeed, the non-Lipchitz case reduction we present is of a very different flavor than, to our knowledge, all known MCSP hardness results. As far as the author knows, it is the only MCSP hardness result that does not easily generalize to an approximation of MCSP. This is because the key step in the reduction is determining, exactly, a Boolean function's circuit complexity modulo a certain prime.

▶ **Open Question 1.2.** *Can one extend Theorem 1.1 to an approximation of* MCSP*?*

We also remark that our notion of size for AC_d^0 formulas is critical for Theorem 1.1. We define the size of an AC_d^0 formula to be the number of input leaves. While this is the standard definition of formula size, we make heavy use of elementary direct product theorems known, specifically, for this notion of formula size. It is not clear to us how to generalize Theorem 1.1 to the case when the size of an AC_d^0 formula is, say, the number of gates or the number of wires.

NP-Hardness of oracle MCSP (MOCSP)

Some work has been done trying to approach the NP-hardness of MCSP "from below," that is, proving that the circuit minimization problem is NP-hard for restricted classes of circuits. As mentioned previously, we know that (DNF)-MCSP [20] and $(\mathsf{OR} \circ \mathsf{AND} \circ \mathsf{MOD}_m)$-MCSP [14] are NP-hard.

Instead, we attempt to approach MCSP from "above." We formulate the Minimum Oracle Circuit Size Problem, denoted MOCSP, that takes as input a truth table T, a size parameter $s \in \mathbb{N}$, and auxiliary truth tables $T_1, \dots, T_t$ and asks whether there is an oracle circuit of size at most s that computes T when given access to $T_1, \dots, T_t$. It is easy to see that $\mathsf{MOCSP} \in \mathsf{NP}$ (the oracle circuit of size s acts as a witness).

We note that this is not the first time someone has considered an "oracle version" of MCSP. Allender *et al.* [1] and Allender, Holden, and Kabanets [4] consider the problem of minimizing oracle circuits for a fixed oracle A. We will denote this problem $\mathsf{MCSP}^{\mathsf{A}}$. An important result for this problem that [1] proves is that $\mathsf{MCSP}^{\mathsf{QBF}}$ is complete for PSPACE under ZPP reductions. MOCSP differs from $\mathsf{MCSP}^{\mathsf{A}}$ in that the oracle circuit gets access to a finite number of Boolean functions, not a language, and the functions the oracle circuit has access to are *inputs* to the problem.

In our view, MOCSP has two advantages over $\mathsf{MCSP}^{\mathsf{A}}$. First, $\mathsf{MOCSP} \in \mathsf{NP}$ while the complexity of $\mathsf{MCSP}^{\mathsf{A}}$ depends on the oracle A. Second, there is an easy reduction from MCSP to MOCSP, simply provide no oracle truth tables. Therefore, we can use MOCSP as a testing ground for hardness results we conjecture for MCSP. Thus, the most natural question is whether we can prove that MOCSP is NP-hard. We prove that MOCSP is indeed NP-hard under non-uniform AC^0 reductions and under uniform randomized reductions.

▶ **Theorem 1.3.**
- $\mathsf{NP} \leq_m^{\mathsf{AC}^0} \mathsf{MOCSP}$
- $\mathsf{NP} \leq_m^{\mathsf{RP}} \mathsf{MOCSP}$
- $\mathsf{NP} \leq_{tt}^{\mathsf{ZPP}} \mathsf{MOCSP}$

These NP-hardness results are all proved by giving a reduction from approximating r-bounded set cover to MOCSP. It is worth noting that the NP-hardness results of (DNF)-MCSP [20] and $(\mathsf{OR} \circ \mathsf{AND} \circ \mathsf{MOD}_m)$-MCSP [14] are also proved via set cover problems.

Given that we can show MOCSP is NP-hard under randomized reductions, one might even begin to hope that we can prove hardness under, say, polynomial-time truth table reductions. Unfortunately, this seems difficult. Essentially the same proofs Murray and Williams [22] or Hitchcock and Pavan [17] use to show that MCSP being NP-hard under polynomial-time truth table reductions implies $\mathsf{EXP} \neq \mathsf{ZPP}$ also works for MOCSP.

▶ **Theorem 1.4** (Essentially proven in [17] and [22]). *If* $\mathsf{NP} \leq_{tt}^{\mathsf{P}} \mathsf{MOCSP}$*, then* $\mathsf{EXP} \neq \mathsf{ZPP}$*.*

Thus, improving our ZPP reduction to a P reduction requires separating EXP from ZPP, a longstanding open problem. For completeness, we give the MOCSP version of Murray and Williams' proof in Appendix B.

Even so, we expect that the ground truth is that MOCSP is NP-hard under, at least, polynomial-time Turing reductions.

▶ **Conjecture 1.5.** $\mathsf{NP} \leq_{\mathsf{T}}^{\mathsf{P}} \mathsf{MOCSP}$.

We give some details on why we believe Conjecture 1.5 near the end of Section 4. Even so, we believe proving such a hardness result is beyond current techniques. Perhaps one could even prove there are some barriers.

A perspective on MOCSP and some questions.

In light of the fact that hardness for MCSP beyond SZK under even non-uniform reductions is unknown, we found these MOCSP hardness results to be quite surprising. To an optimist, NP-hardness results for MOCSP could even be a first step towards proving hardness for MCSP. Indeed, a PSPACE-hardness result was first proved by Buhrman and Torenvliet [8] for an "oracle" version of space-bounded Kolmogorov complexity before Allender *et al.* [1] showed PSPACE-hardness for the non-oracle version about four years later.[5]

Even if stronger hardness results for MCSP remain out of reach, MOCSP could still yield valuable insights about MCSP. For instance, it would be interesting to see which of the barriers and non-hardness results known for MCSP carry over to MOCSP.

▶ **Open Question 1.6.** *Can one show that other barriers or non-hardness results that hold for* MCSP *also hold for* MOCSP*?*

As an example of the insight given by answers to this question, consider Murray and Williams' [22] result that proving MCSP is NP-hard under polynomial-time many-one reductions implies $\mathsf{EXP} \neq \mathsf{ZPP}$. A natural question one might ask is whether we can improve this theorem to show that MCSP being NP-hard under randomized reductions implies unknown class separations. As we note in Theorem 1.4, however, Murray and Williams' proof carries over to MOCSP, and Theorem 1.3 shows MOCSP is indeed NP-hard under randomized reductions. Thus, any improvement of Murray and Williams' result to randomized reductions likely requires a fact about MCSP that we do not know for MOCSP.

In another direction, our results seem to imply that proving hardness for MOCSP is easier than proving hardness for MCSP. Indeed, since MCSP easily reduces to MOCSP, any hardness result that is true for MCSP must also be true for MOCSP. Therefore, we can use MOCSP as a testing ground for hardness results we conjecture about MCSP. For example, Hirahara's [13] worst-case to average-case reduction for NP can be based on a certain approximation of MCSP being NP-hard, which would imply that a certain approximation of MOCSP is

[5] The conference versions of [8] and [1] are four years apart.

also NP-hard. Given that we can prove the NP-hardness of MOCSP under uniform ZPP reductions and non-uniform AC^0 reductions, we ask if one can prove something similar for the approximation version of MOCSP.

▶ **Open Question 1.7.** *Can one prove that, for some $\epsilon > 0$, approximating* MOCSP *on n-inputs to a factor of n^ϵ is* NP*-hard under, say,* P/poly *reductions? Conversely, can one prove that there is any barrier to showing such a hardness result?*

We note that the techniques we use to prove NP-hardness results for MOCSP seem to break down completely in the case of even super-constant approximation, so answering this question will likely require new ideas.

1.4 Proof Overviews

In this section, we give fairly detailed overviews of our proofs. In doing this, we will often state results without filling in low-level details. To make clear to the reader when we are doing this, we mark such sentences with an italicized *we observe*.

Majority Hardness for (AC^0_d)-MCSP

Recall, AC^0_d is the class of depth-d formulas. We also define $\mathsf{AND} \circ \mathsf{AC}^0_{d-1}$ and $\mathsf{OR} \circ \mathsf{AC}^0_{d-1}$ be the classes of AC^0_d formulas with a top AND and top OR gate respectively. For $\mathcal{F} \in \{\mathsf{AC}^0_d, \mathsf{AND} \circ \mathsf{AC}^0_{d-1}, \mathsf{OR} \circ \mathsf{AC}^0_{d-1}\}$ and a truth table T, we let $\mathsf{CC}_{\mathcal{F}}(T)$ denote the size of the minimum $\mathcal{F}$-formula computing T where the size of a formula is the number of input leaves.

Our analysis proceeds by considering each $n \in \mathbb{N}$ and splitting into cases depending on whether $\mathsf{CC}_{\mathsf{AC}^0_d}$ is Lipchitz on truth tables of length around n. In more detail, fix some sufficiently large n. Let $q = \Theta(n^2)$ be a power of two. We divide into cases depending on whether there exists an $m \in \{q^{10}, q^{50}\}$ such that $\mathsf{CC}_{\mathsf{AC}^0_d}$ is $(m^{.25})$-Lipchitz on truth tables of length m.

Case 1: Lipchitzness holds for some m.

If there does exist an $m \in \{q^{10}, q^{50}\}$ such that $\mathsf{CC}_{\mathsf{AC}^0_d}$ is $(m^{.25})$-Lipchitz on truth tables of length m, then the techniques of [11] yield an AC^0 truth table reduction from MAJORITY on n-bits to (AC^0_d)-MCSP on m-bits. For completeness, we include a self-contained proof of this case in Appendix A.

Case 2: Lipchitzness fails.

Assume that for all $m \in \{q^{10}, q^{50}\}$ $\mathsf{CC}_{\mathsf{AC}^0_d}$ is not $(m^{.25})$-Lipchitz on truth tables of length m. Let $u = q^{10}$ and $v = q^{50}$.

Lipchitzness failing $\implies$ functions easier to compute with a top* AND *gate. *We observe*, as a straight forward consequence of Lipchitzness failing, that there exists a truth table of length u that has an optimal formula with large top fan-in and and a truth table of length v that is easier to compute with a top AND gate:

1. There exists a Boolean function f^u that takes $\log u$ inputs and an AC^0_d formula ϕ^u such that ϕ^u is an optimal AC^0_d formula for f^u and $\phi^u = \phi^u_1 \wedge \cdots \wedge \phi^u_t$ for some $t \geq n$ and some $\phi^u_1, \ldots, \phi^u_t \in \mathsf{AC}^0_{d-1}$.

2. There exists a Boolean function f^v that takes $\log v$ inputs such that $\mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f^v) > \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) + u\log u$.

We will make use of f^u and ϕ^u to reduce $\mathsf{MAJORITY}$ to $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$ and we will use f^v to reduce $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$ to $\mathsf{CC}_{\mathsf{AC}^0_d}$.

Using $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$ and optimal subformulas of ϕ^u to compute a dot product. The heart of our $\mathsf{MAJORITY}$ reduction is a fairly elementary observation about optimal $(\mathsf{AND}\circ\mathsf{AC}^0_{d-1})$ formulas. Recall, $\phi^u = \phi^u_1 \wedge \cdots \wedge \phi^u_t$ is an optimal (AC^0_d) formula for f^u and, hence, also an optimal $(\mathsf{AND}\circ\mathsf{AC}^0_{d-1})$ formula for f^u. *We observe* that for any $A \subseteq [t]$, the $(\mathsf{AND}\circ\mathsf{AC}^0_{d-1})$-optimality of ϕ^u implies that $\bigwedge_{i\in A}\phi^u_i$ is also an optimal $(\mathsf{AND}\circ\mathsf{AC}^0_{d-1})$ formula for the function it computes.

Introducing some notation, for a string $x \in \{0,1\}^n$, we let f^u_x be the function given by $\bigwedge_{i\in O_x}\phi^u_i$ where $O_x \subseteq [n]$ are the bits in x that are one. Using the above observation about subformulas being optimal, we have that $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x) = \sum_{i\in O_x}|\phi^u_i|$.[6] Thus, one can think of $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)$ as computing the dot product between x and the vector $\langle|\phi^u_1|,\ldots,|\phi^u_n|\rangle$.

Note that that the definition of f^u_x depends on the labeling of $\phi^u_1,\ldots,\phi^u_t$, in particular the choice of which ϕ^u_i have $i \le n$. We will later choose an labeling of the ϕ^u_i that is convenient.

Computing $\mathsf{MAJORITY}$ (non-uniformly) using $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$. Our goal is to compute $\mathsf{MAJORITY}$ on a string $x \in \{0,1\}^n$ using the above "dot product" observation. Before we show how to do this, we give some intuition on how we came up with the idea.

Instead of trying to compute $\mathsf{MAJORITY}$, suppose we relaxed the problem to computing PARITY given access to the integer produced by the dot product $x\cdot\langle|\phi^u_1|,\ldots,|\phi^u_n|\rangle$. Well, if it so happened that all the entries in the vector $\langle|\phi^u_1|,\ldots,|\phi^u_n|\rangle$ were odd, then it is clear that the integer produced by $x\cdot\langle|\phi^u_1|,\ldots,|\phi^u_n|\rangle$ is odd if and only if x has an odd number of ones. Our approach for $\mathsf{MAJORITY}$ is a generalization of this.

Let $p = O(n)$ be prime greater than n. *We observe*, via an averaging argument, that there exists integers $k \ge 0$ and $1 \le r \le p-1$ such that (after relabeling the ϕ^u_i)

$$|\phi^u_1|/p^k \equiv \cdots \equiv |\phi^u_n|/p^k \equiv r \mod p.$$

Thus, we can determine the weight w of x (and hence compute $\mathsf{MAJORITY}$ of x) by computing the value of

$$\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)/p^k = \sum_{i\in O_x}|\phi^u_i|/p^k \equiv rw \mod p$$

and multiplying by the inverse of r modulo p. [7]

Reducing computing $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$ to computing $\mathsf{CC}_{\mathsf{AC}^0_d}$. Ultimately, we want to compute $\mathsf{MAJORITY}$ using $\mathsf{CC}_{\mathsf{AC}^0_d}$ not $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}$. By the above procedure, it suffices to show how to compute $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)$ using $\mathsf{CC}_{\mathsf{AC}^0_d}$.

---

[6] Recall, our notion of formula size is the number of input leaves.

[7] In case the reader is unsure of whether the last parts of this procedure are implementable in AC^0, realize that the output of $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)$ is a binary string of length $O(\log n)$ and that any function on a string of length $O(\log n)$ can be computed by a polynomial-sized DNF. See the proof in Section 3 for more details.

We can make such a computation as follows. Recall that f^v is a function satisfying

$$\mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f^v) > \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) + u\log u$$

whose existence is guaranteed by the failure of Lipchitzness. Take the direct product of f_x^u with f^v to obtain a function $g_x(y,z) = f_x^u(y) \wedge f^v(z)$. Since the difference between computing f^v with a top AND gate and a top OR gate is larger than $u\log u$ (which is the maximum complexity of f_x^u), *we observe*[8] any optimal AC^0_d formula for g_x must have a top AND gate, so

$$\mathsf{CC}_{\mathsf{AC}^0_d}(g_x) = \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x).$$

Then, *we observe* that

$$\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x) = \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f_x^u) + \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v).$$

Hence, if we are non-uniformly given the value of $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v)$, we can subtract $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v)$ from $\mathsf{CC}_{\mathsf{AC}^0_d}(g_x)$ to find $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f_x^u)$.

NP-hardness of Oracle MCSP (MOCSP)

We define the size of an oracle circuit to be the total number of AND, OR, and oracle gates. The Minimum Oracle Circuit Size Problem, MOCSP, takes as input a truth table T, a threshold $s \in \mathbb{N}$, and auxiliary truth tables $T_1, \dots T_t$ and outputs whether there is an oracle circuit of size at most s that computes T when given oracle access to $T_1, \dots, T_t$. We denote the output of MOCSP on such an input as $\mathsf{MOCSP}(T, s; T_1, \dots, T_t)$. We denote the minimum size of any oracle circuit computing T when given access to $T_1, \dots, T_t$ as $\mathsf{CC}^{T_1,\dots,T_t}(T)$.

We prove that MOCSP is NP-hard under various reductions by giving a reduction from 4-approximating r-bounded set cover, denoted $4\text{-}\mathsf{SetCover}_r$, to MOCSP. As a reminder, $4\text{-}\mathsf{SetCover}_r$ is the promise problem takes as input sets $S_1, \dots, S_t \subseteq [n]$ of cardinality at most r whose union is $[n]$ as well as an integer $c \in [n]$ and requires outputting YES when $c \geq \ell$ and NO when $c < \ell/4$ where ℓ is the optimal cover size, i.e.

$$\ell = \min\{|I| : I \subseteq [t] \text{ and } \bigcup_{i\in I} S_i = [n]\}.$$

For sufficiently large r, $4\text{-}\mathsf{SetCover}_r$ is known to be NP-hard (see Theorem 2.6).

Informal idea. We begin by giving a high-level overview of the reduction to orient the reader. (It will be very informal, but we are building to a more detailed description.) Say we are given sets $S_1, \dots, S_t \subseteq [n]$ of cardinality at most r whose union is $[n]$. One can think of each of these sets S_i as "seeing" a small portion of the ground set $[n]$. For some carefully chosen truth table T of length $m \geq n$, we let each set $S_1, \dots, S_t$ induce truth tables $T_{S_1}, \dots, T_{S_t}$ respectively where each truth table T_{S_i} "sees" roughly the same part of T as S_i "sees" of $[n]$. Finally, we ask how hard it is for a circuit to compute T given oracle access to $T_1, \dots, T_t$, and we show that, if T has a certain property, then the answer to this question is the answer to the set cover problem up to a constant factor.

We now illustrate the algorithm in more detail. Fix sets $S_1, \dots, S_t \subseteq [n]$ of cardinality at most r whose union is $[n]$ and fix a truth table T of length m. Assume the optimal cover size of $[n]$ by $S_1, \dots, S_t$ is ℓ.

---

[8] both the "*we observe*" *statements in this paragraph are consequences of standard direct product theorems for formulas.*

The truth tables induced by $S_1, \dots, S_t$ and T. We rigorously define the truth tables $T_{S_1}, \dots, T_{S_t}$ of length m induced by $S_1, \dots, S_t$ and T. First, we fix a partition of $[m]$ into n sets. It does not really matter what partition we choose as long as the sets are roughly the same size and the partition is easily computable, but, for concreteness, let $P_1, \dots, P_n \subseteq [m]$ be the partition of $[m]$ given by $P_i = \{j \in [m] : j \equiv i \mod n\}$.

We can then use this partition to "lift" any subset of $[n]$ into a subset of $[m]$ as follows. For a subset S of $[n]$, let S^m denote the subset of $[m]$ given by $S^m = \bigcup_{i \in S} P_i$.

Next, for a subset $P \subseteq [m]$, we let T_P be the truth table of length m that "sees" T on the elements of P and zeroes everywhere else, that is the ith bit of T_P is

$$\begin{cases} \text{the } i\text{th bit of } T & \text{, if } i \in P \\ 0 & \text{, otherwise.} \end{cases}$$

Finally, we define the truth table T_{S_i} induced by S_i to be the truth table $T_{S_i^{\mathcal{P}}}$ given by the above notation (we are dropping the m superscript for concision).

$\mathsf{CC}^{T_{S_1}, \dots, T_{S_t}}(T)$ is at most 2ℓ. Suppose, without loss of generality, that $S_1 \cup \dots \cup S_\ell$ is an optimal cover of $[n]$. Then, by construction, the function computed by $T_{S_1} \vee \dots \vee T_{S_\ell}$ is T. This is an oracle circuit of size $2\ell - 1$, so $\mathsf{CC}^{T_{S_1}, \dots, T_{S_t}}(T) \leq 2\ell$.

If T is (rn)-irritable, then $\mathsf{CC}^{T_{S_1}, \dots, T_{S_t}}(T) > \ell/2$. Recall the notation defined previously that, for a set $P \subseteq [m]$, T_P denotes the string of length m that equals T on the bits in P and is zero everywhere else. Also recall that we fixed a partition $P_1, \dots, P_n$ of $[m]$. We say that T is (s)-*irritable* if for all $i \in [n]$ we have that

$$\mathsf{CC}^{T_{P_1}, \dots, T_{P_{i-1}}, T_{P_{i+1}}, \dots, T_{P_n}}(T) > s.$$

Informally, T being (rn)-irritable means that if you take away access to any particular T_{P_i} oracle, then computing T requires an oracle circuit of size greater than rn, which is a $(r/2)$-factor jump over the trivial $2n$-sized oracle circuit given by $T_{P_1} \vee \dots \vee T_{P_n}$ if one had full oracle access.

Now assume T is (rn)-irritable. We need to show that $\mathsf{CC}^{T_{S_1}, \dots, T_{S_t}}(T) > \ell/2$. For contradiction, suppose that C is an oracle circuit computing T with at most $\ell/2$ gates. Then C uses at most $q \leq \ell/2$ unique oracle gates. Without loss of generality, assume C uses as oracles only $T_{S_1}, \dots, T_{S_q}$. Next, note that for any $i \in [q]$, T_{S_i} can, by construction, be computed by the oracle circuit $\bigvee_{j \in S_i} T_{P_j}$. Moreover, this is an oracle circuit for T_{S_i} of size at most $2r$ since $|S_i| \leq r$. Thus, replacing each T_{S_i} oracle gate in C with the oracle circuit $\bigvee_{j \in S_i} T_{P_j}$, we can transform C into an oracle circuit D of size at most $r \cdot |C| \leq r\ell$ such that D computes T when given access to the oracles in the set $O = \{T_{P_j} : j \in S_1 \cup \dots \cup S_q\}$. However, since $q \leq \ell/2$ is less than the optimal cover size, $|S_1 \cup \dots \cup S_q| < n$ and so $|O| < n$, so O is missing $T_{P_{i^\star}}$ for some $i^\star \in [n]$. But then D is an oracle circuit of size at most $r\ell \leq rn$ that computes T when given access to $T_{P_1}, \dots, T_{P_n}$ without using $T_{P_i^\star}$ as an oracle gate, which contradicts that T is (rn)-irritable.

$\mathsf{RP}, \mathsf{ZPP}$ and AC^0 reductions. At this point, we have shown that one can compute whether $S_1, \dots, S_t$ admits a c-cover (up to a 4-approximation) by outputting $\mathsf{MOCSP}(T, 2c; T_{S_1}, \dots, T_{S_t})$ for some T that is (rn)-irritable.

We observe by a counting argument that a truth table T of length $m \geq n^3$ picked uniformly at random is (rn)-irritable with high probability. Thus, picking a random truth table T of length $\Theta(n^3)$ and outputting $\mathsf{MOCSP}(T, 2c; T_{S_1}, \ldots, T_{S_t})$ gives an RP many-one reduction from 4-$\mathsf{SetCover}_r$ to MOCSP (note that we get one-sided error because irritability was only required for the $\ell/2$ lower bound and not required for the 2ℓ upper bound). Additionally, since we can check if a random T is (rn)-irritable using an oracle to MOCSP, *we observe* that 4-$\mathsf{SetCover}_r \leq_{tt}^{\mathsf{ZPP}}$ MOCSP. Finally, *we observe* that there is an AC^0 circuit C such that $C(T, c, S_1, \ldots, S_t) = (T, 2c; T_{S_1}, \ldots, T_{S_t})$. Therefore, by non-uniformly hardcoding an (rn)-irritable truth table T into C, we get that 4-$\mathsf{SetCover}_r$ reduces to MOCSP under (non-uniform) AC^0 many-one reductions.

1.5 Paper Organization

In Section 2 we fix notation and review precise definitions. In Section 3 we prove that MAJORITY reduces to (AC^0_d)-MCSP, and in Section 4 we prove our MOCSP results.

2 Preliminaries

For an integer n, we let $[n]$ denote the set $\{1, \ldots, n\}$. For a binary string $x \in \{0,1\}^*$, the *weight* of x, denoted $\mathsf{wt}(x)$, is the number ones in x. We identify a Boolean function $f : \{0,1\}^n \to \{0,1\}$ with its truth table $T \in \{0,1\}^{2^n}$ and often use them interchangeably.

We let log denote the base-2 logarithm and ln represent the base-e logarithm. For functions f and g, we say $f = \tilde{O}(g)$ if there exists a c such that $f(x) \leq \log^c(g(x))g(x)$ for all sufficiently large x. We say that $f = \tilde{\Omega}(g)$ if $g = \tilde{O}(f)$.

We say a function $f : \{0,1\}^n \to \mathbb{R}$ is *c-Lipchitz* if for all $x, y \in \{0,1\}^n$ that differ in at most one bit, $|f(x) - f(y)| \leq c$.

Complexity classes and reductions

We assume the reader is familiar with the standard complexity classes such as AC^0, P, ZPP, RP, NP, E and the notion of Turing machines. For background on these, we refer to Arora and Barak's excellent textbook [7]. For us, AC^0 always refers to *non-uniform* AC^0.

We review the types of reductions we use in case the reader is not familiar with randomized reductions, truth table reductions, or our notation.

Many-one reductions. We will make use of the follow notions of many-one reduction.

- $L \leq_m^{\mathsf{AC}^0} L'$ if there is a non-uniform (polynomial-sized) AC^0 circuit C such that $x \in L \iff C(x) \in L'$.
- $L \leq_m^{\mathsf{P}} L'$ if there is a polynomial-time Turing machine M such that $x \in L \iff M(x) \in L'$.
- $L \leq_m^{\mathsf{RP}} L'$ if there is a polynomial-time probabilistic Turing machine M taking in a "random" auxiliary input r such that

$$x \in L \implies \forall r\, M(x, r) \in L', \text{ and}$$

$$x \notin L \implies \Pr_r[M(x, r) \in L'] \geq 2/3,$$

  and $|r|$ is polynomial in the length of x.

Truth table reductions. We will also make use of the following notions of truth table reductions.

- We say an oracle circuit C is a *truth table oracle circuit* if there is no directed path between oracle gates in C.
- $L \leq_{tt}^{\mathsf{AC}^0} L'$ if there is a non-uniform (polynomial-sized) AC^0 truth table oracle circuit C such that C computes L when given oracle access to L'.
- $L \leq_{tt}^{\mathsf{ZPP}} L'$ if L can be computed with zero-error by a polynomial-time probabilistic oracle Turing machine M with oracle access to L' with the caveat that all of M's oracle queries must be answered simultaneously (i.e. so no oracle query can depend on another oracle query). On any single input, M is allowed to output "don't know" with probability at most 1/2.

AC^0_d formulas, (AC^0_d)-MCSP, and $\mathsf{CC}_{\mathsf{AC}^0_d}$

For an integer $d \geq 2$, we let AC^0_d denote the class of depth-d formulas that use AND and OR gates with unbounded fan-in and fan-out 1 and that takes as "input leaves" the bits of a binary string and the negation of each of those bits.

For an AC^0_d formula ϕ, we define the size of ϕ, denoted $|\phi|$, to be the total number of input leaves ϕ uses. For a Boolean function f, we let $\mathsf{CC}_{\mathsf{AC}^0_d}(f)$ be the size of the smallest AC^0_d formula computing f.

▶ **Definition 2.1** (Minimum Circuit Size Problem for constant depth formulas). (AC^0_d)-MCSP, *is the language given by*

$$\{(T, s) \in \{0,1\}^\star \times \mathbb{N} : T \text{ is the truth table of a Boolean function, and } \mathsf{CC}_{\mathsf{AC}^0_d}(T) \leq s\}.$$

We will also make use of the classes of formulas $\mathsf{OR} \circ \mathsf{AC}^0_{d-1}$ and $\mathsf{AND} \circ \mathsf{AC}^0_{d-1}$, defined as the subclasses of AC^0_d formulas with a top OR gate and a top AND gate respectively. For $\mathcal{C} \in \{\mathsf{OR} \circ \mathsf{AC}^0_{d-1}, \mathsf{AND} \circ \mathsf{AC}^0_{d-1}\}$, we define We define $\mathsf{CC}_{\mathcal{C}}$ and $(\mathcal{C})$-MCSP analogous to $\mathsf{CC}_{\mathsf{AC}^0_d}$ and (AC^0_d)-MCSP.

We also require the following elementary lemmas regarding AC^0_d formulas.

▶ **Lemma 2.2.** *Let f be a Boolean function. Then $\mathsf{CC}_{\mathsf{AC}^0_d}(f) = \mathsf{CC}_{\mathsf{AC}^0_d}(\neg f)$.*

Proof. One can use DeMorgan's laws to turn any AC^0_d formula for f of size s into an AC^0_d formula for $\neg f$ of size s. ◀

We note that our specific notion of AC^0_d formula size is crucial for the next lemma.

▶ **Lemma 2.3** (Direct product theorem for formulas). *Let $f : \{0,1\}^n \to \{0,1\}$ and $g : \{0,1\}^m \to \{0,1\}$ be Boolean functions that are both not the constant zero function. Define $h : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ by $h(x,y) = f(x) \wedge g(y)$. Then*

$$\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(h) = \mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f) + \mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(g), \text{ and}$$

$$\mathsf{CC}_{\mathsf{OR} \circ \mathsf{AC}^0_{d-1}}(h) \geq \mathsf{CC}_{\mathsf{OR} \circ \mathsf{AC}^0_{d-1}}(f) + \mathsf{CC}_{\mathsf{OR} \circ \mathsf{AC}^0_{d-1}}(g).$$

Proof. It is easy to see that

$$\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(h) \leq \mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f) + \mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(g).$$

On the other hand, since f is not the constant 0 function, it has a 1-valued input x_0. Then, $h(x_0, y)$ computes $g(y)$. Thus, if ϕ is an $\mathsf{OR} \circ \mathsf{AC}^0_{d-1}$ formula for h, then ϕ has at least

$\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g)$ y-input leaves. A similar argument shows that ϕ has at least $\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f)$ x-input leaves. Hence

$$\mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(h) \geq \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f) + \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g).$$

A similar argument shows that

$$\mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(h) \geq \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f) + \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(g).$$

◀

Oracle Circuits and Oracle MCSP: MOCSP

An oracle circuit C is made up of NOT gates, fan-in two AND and OR gates, and oracle gates $g_1, \ldots, g_t$ with fan-in $i_1, \ldots, i_t$ respectively for some integers $t, i_1, \ldots, i_t \geq 1$. When given functions $f_1 : \{0,1\}^{i_1} \to \{0,1\}, \ldots, f_t : \{0,1\}^{i_t} \to \{0,1\}$, we let $C^{f_1,\ldots,f_t}(x)$ be the value obtained when evaluating C on input x by using the $f_1, \ldots, f_t$ functions as the outputs of the $g_1, \ldots, g_t$ gates respectively.

We define the size of an oracle circuit C, denoted $|C|$, to be the sum of the number of OR gates, the number of AND gates, and the number of oracle gates in C.

For Boolean functions $f, f_1, \ldots, f_t$, we let $\mathsf{CC}^{f_1,\ldots,f_t}(f)$ be the size of the smallest oracle circuit that computes f when given access to $f_1, \ldots, f_t$. Analogous to MCSP, we define the following.

▶ **Definition 2.4** (The Minimum Oracle Circuit Size Problem)**.** *The Minimum Oracle Circuit Size Problem, denoted* MOCSP*, takes as as input a truth table T, a threshold $s \in \mathbb{N}$, and oracle truth tables $T_1, \ldots, T_t$ and outputs whether $\mathsf{CC}^{T_1,\ldots,T_t}(T) \leq s$. The output of* MOCSP *on such an input is written as* $\mathsf{MOCSP}(T, s; T_1, \ldots, T_t)$.

r-Bounded Set Cover

We will make use of the following well known NP-complete problem.

▶ **Definition 2.5** (r-Bounded Set Cover)**.** *r-Bounded Set Cover, denoted* $\mathsf{SetCover}_r$*, is the problem that takes as input a tuple $(n, c, S_1, \ldots, S_t)$, where $n \in \mathbb{N}$ is a universe size, $c \in \mathbb{N}$ is a proposed cover size $1 \leq c \leq n$, and $S_1, \ldots, S_t \subseteq [n]$ are sets of cardinality at most r whose union is $[n]$, and outputs whether $c \geq \ell$ where ℓ is the optimal cover size given by*

$$\ell = \min\{|I| : I \subseteq [t] \text{ and } \cup_{i\in I} S_i = [n]\}.$$

We will also make use of the following restricted version of set cover. Let $\mathsf{SetCover}_{r,n,t}$ denote r-bounded set cover on t subsets of $[n]$. We encode inputs to $\mathsf{SetCover}_{r,n,t}$ as the tuple $(c, S_1, \ldots, S_t)$ (with n implicit) where c is represented in binary and the set S_i, for each $i \in [t]$, is represented by a bit string of length $r\lceil\log(n+1)\rceil$ that is a concatenated list of the elements of S_i in binary, padded with zeroes if $|S_i| < r$ (note that zero is not an element of $[n]$, so padding with zeroes is not ambiguous).

We will use that $\mathsf{SetCover}_r$ is NP-hard even to approximate to a roughly $\ln r$ factor.

▶ **Theorem 2.6** (Feige [10] and Trevisan [31])**.** *Let r be a sufficiently large constant, and let L be a language. If for every instance $x = (n, c, S_1, \ldots, S_t)$ of* $\mathsf{SetCover}_r$*, we have that both that*

- *$c \geq \ell$ implies $x \in L$, and*
- *$c \leq \ell/(\ln r - O(\ln\ln r))$ implies $x \notin L$,*

where ℓ is the optimal cover size, then L is NP*-hard under polynomial-time many-one reductions.*

3 MAJORITY $\leq_{tt}^{\mathsf{AC}^0}$ (AC_d^0)-MCSP

Let $d \geq 2$. Our goal in this section is to prove the following theorem.

▶ **Theorem 3.1.** MAJORITY $\leq_{tt}^{\mathsf{AC}^0}$ (AC_d^0)-MCSP.

We will do this by showing that for all sufficiently large $n \in \mathbb{N}$, there exists an AC^0 truth table oracle circuit that computes MAJORITY on n-bits when given access to (AC_d^0)-MCSP.

Fix some n, and let q be the least power of two such that $n \leq \sqrt{q}/2$. We will split our analysis into cases depending on whether there exists an $m \in \{q^{10}, q^{50}\}$ such that $\mathsf{CC}_{\mathsf{AC}_d^0}$ is $(m^{.25})$-Lipchitz on inputs of length m.

3.1 Case 1: Lipchitzness Holds

If Lipchitzness holds, then the desired (AC_d^0)-MCSP oracle circuit C exists for computing MAJORITY on n-inputs by the work of Golovnev *et al.* [11]. At a high-level, C works by using the input string to sample a random variable whose circuit complexity spikes (in expectation) depending on the weight of the input and using Lipchitzness to show that this spike happens with such high probability that it can be derandomized using non-uniformity.

For completeness, we give a self-contained proof of this case in Appendix A.

3.2 Case 2: Lipchitzness fails

Assume that for all $m \in \{q^{10}, q^{50}\}$, $\mathsf{CC}_{\mathsf{AC}_d^0}$ is not $(m^{.25})$-Lipchitz on inputs of length m. Thus, for all $m \in \{q^{10}, q^{50}\}$ there exist functions $f^m, h^m : \{0,1\}^{\log m} \to \{0,1\}$ that differ only on a single input $z^m \in \{0,1\}^{\log m}$ such that $\mathsf{CC}_{\mathsf{AC}_d^0}(h^m) - \mathsf{CC}_{\mathsf{AC}_d^0}(f^m) > m^{.25}$.

We assume, without loss of generality, that for all $m \in \{q^{10}, q^{50}\}$, $f^m(z^m) = 0$ and $h^m(z^m) = 1$. (If this is not the case, then replace f^m and h^m by $\neg f^m$ and $\neg h^m$ respectively and apply Lemma 2.2.)

First, we show that the failure of Lipchitzness implies the existence of functions that are much easier to compute by formulas with an AND gate on top. For $m \in \{q^{10}, q^{50}\}$ let $\mathbb{1}_{z^m} : \{0,1\}^{\log m} \to \{0,1\}$ denote the indicator function that accepts just the string z^m.

▶ **Proposition 3.2.** *Let $m \in \{q^{10}, q^{50}\}$. For sufficiently large n, $\mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}_{d-1}^0}(f^m) \geq \mathsf{CC}_{\mathsf{AC}_d^0}(f^m) + m^{.24}$, and so any optimal AC_d^0 formula for f^m has an* AND *gate on top.*

Proof. Suppose ϕ is an $\mathsf{OR} \circ \mathsf{AC}_{d-1}^0$ formula computing f^m, that is, f^m is computed by $\phi = \phi_1 \vee \cdots \vee \phi_t$ for some AC_{d-1}^0 formulas $\phi_1, \ldots, \phi_t$. Then

$$\mathbb{1}_{z^m} \vee \phi_1 \vee \cdots \vee \phi_t$$

computes h^m. Since $\mathbb{1}_{z^m}$ can be computed by a single AND gate of formula size $\log m$, this shows that $\mathsf{CC}_{\mathsf{AC}_d^0}(h^m) \leq |\phi| + \log m$. Combining this with the fact that $\mathsf{CC}_{\mathsf{AC}_d^0}(h^m) - \mathsf{CC}_{\mathsf{AC}_d^0}(f^m) \geq m^{.25}$ gives the desired result. ◀

At this point, we will need to refer to both q^{10} and q^{50} individually, so for convenience let $u = q^{10}$ and $v = q^{50}$.

Let ϕ^u be an optimal AC_d^0 formula for f^u. By Proposition 3.2, for sufficiently large n, we know that $\phi^u = \phi_1^u \wedge \cdots \wedge \phi_t^u$ for some AC_{d-1}^0 formulas $\phi_1^u, \ldots, \phi_t^u$. Moreover, we can assume, without loss of generality, that the top gate of ϕ_i^u is OR for all $i \in [t]$. (If some ϕ_i^u has an AND gate on top, then this AND can be carried out by the AND gate on top of ϕ^u without increasing the size of the formula.)

Our next Proposition shows that ϕ^u has high top fan-in.

▶ **Proposition 3.3.** *For sufficiently large n,*

$$t \geq u^{.24}.$$

Proof. We divide into cases depending on d.

Case 1: $d \geq 3$. Realize that

$$(\phi_1^u \vee \mathbb{1}_{z^u}) \wedge \cdots \wedge (\phi_t^u \vee \mathbb{1}_{z^u})$$

computes h^u. Since $\mathbb{1}_{z^u}$ can be computed by a single AND gate of formula size $\log u$ and the top gate of each ϕ_i^u is an OR gate and $d \geq 3$, this yields a depth-d formula for h^u of size $\mathsf{CC}_{\mathsf{AC}^0_d}(f^u) + t \log u$. Since $\mathsf{CC}_{\mathsf{AC}^0_d}(h^u) - \mathsf{CC}_{\mathsf{AC}^0_d}(f^u) \geq u^{.25}$, the desired bound on t follows.

Case 2: $d = 2$. Let $\mathbb{1}_{z^u,j} : \{0,1\}^{\log u} \to \{0,1\}$ be the function that accepts a string x if and only if the jth bit of x equals the jth bit of z^u. Observe that, since $\bigwedge_{i\in[t]} \phi_i^u$ computes f^u, we have that

$$\bigwedge_{i\in[t]} \bigwedge_{j\in[\log u]} (\phi_i^u \vee \mathbb{1}_{z^u,j})$$

computes h^u. Since $\mathbb{1}_{z^u,j}$ is computed by a single input leaf and ϕ_i^u has an OR gate on top, this yields a depth-2 formula for h^u of size $(|\phi^u| + 1)\log u$. Since ϕ^u is an optimal CNF, each clause ϕ_i^u of ϕ^u is the OR of at most $\log u$ input leaves. In other words, $|\phi_i^u| \leq \log u$. Therefore, we have that

$$\mathsf{CC}_{\mathsf{AC}^0_d}(h^u) \leq (|\phi^u| + 1)\log u \leq (\sum_{i\in[t]} |\phi_i^u| + 1)\log u \leq (t\log u + 1)\log u = t\log^2 u + \log u.$$

On the other hand, we know by assumption that

$$\mathsf{CC}_{\mathsf{AC}^0_d}(h^u) > \mathsf{CC}_{\mathsf{AC}^0_d}(f^u) + u^{.25} \geq u^{.25}.$$

Combining these two inequalities gives us the desired bound on t.

◀

Let p be smallest prime greater than n. (Note that $p \leq 2n$ by Betrand's postulate, also known as Chebyshev's theorem. See [12] for a proof.) We say that an integer j is (k,r)-*good* for integers $k \geq 0$ and $1 \leq r \leq p-1$ if p^k divides j and $j/p^k \equiv r \mod p$. In other words, an integer j is (k,r)-good for $k \geq 0$ and $r \in [p-1]$ if the kth largest entry of the base-p representation of the integer j equals r and all previous entries equal zero. From this "base-p" perspective, it is clear that all positive integers j are (k,r)-good for some $k \geq 0$ and $r \in [p-1]$.

We show that, for some k and r, a large subset of the $|\phi_i^u|$ are (k,r)-good.

▶ **Proposition 3.4.** *For all sufficiently large n, there exist integers $k \geq 0$ and $1 \leq r \leq p-1$ and a set $S \subseteq [t]$ of cardinality n such that, for all $i \in S$, the integer $|\phi_i^u|$ is (k,r)-good.*

Proof. We do this by an averaging argument. First, we show that each $|\phi_i^u|$ is (k,r)-good for a k not too large.

▷ Claim 3.5. For all $i \in [t]$, $|\phi_i^u|$ is (k,r)-good for some $0 \leq k \leq \log(u\log u) + 1$ and some $r \in [p-1]$.

Proof. Fix some $i \in [t]$. $|\phi_i^u|$ is a positive integer, so $|\phi_i^u|$ is (k,r)-good for some $k \geq 0$ and some $r \in [p-1]$. We still need to upper bound this k. Note that the size of $|\phi_i^u|$ is at most by $u \log u$ since ϕ^u is optimal for f^u and f^u can be computed by a DNF of size $u \log u$. Thus, for p^k to divide $|\phi_i^u|$, we must have that $k \leq \log_p(u \log u) + 1 \leq \log(u \log u) + 1$. ◁

Since for all $i \in [t]$ we have shown that $|\phi_i^u|$ is (k,r) good for some $0 \leq k \leq \log(u \log u) + 1$ and some $r \in [p-1]$, a standard averaging argument implies that there exists a set $S \subseteq [t]$ of cardinality at least

$$\frac{t}{(\log(u \log u) + 1)(p-1)}$$

such that for all $i \in S$, $|\phi_i^u|$ is (k,r)-good for some fixed $k \geq 0$ and $1 \leq r \leq p-1$. For sufficiently large n, we have that

$$\frac{t}{(\log(u \log u) + 1)(p-1)} \geq \frac{u^{.24}}{4n \log u} \geq n$$

using that $u = q^{10} \geq n^{10}$. We then can truncate S so that it has only n elements as desired. ◀

Assume that n is large enough that all the sufficiently large hypotheses in Propositions 3.2, 3.3, and 3.4 apply. For convenience, relabel $\phi_1, \ldots, \phi_t$ so that the set S guaranteed by Proposition 3.4 is just $S = [n]$. Fix $k \geq 0$ and $r \in [p-1]$ to be the values such that for all $i \in S = [n]$, $|\phi_i^u|$ is (k,r)-good.

Introducing notation, for a set $A \subseteq [n]$, let f_A^u be the function computed by $\bigwedge_{i \in A} \phi_i^u$.

▶ **Lemma 3.6.** *Let $A \subseteq [n]$. Then $\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f_A^u) = \sum_{i \in A} |\phi_i^u|$.*

Proof. By construction, we have that $\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f_A^u) \leq \sum_{i \in A} |\phi_i|$. Suppose for contradiction that $\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f_A^u) < \sum_{i \in A} |\phi_i^u|$.

Let $\theta_1 \wedge \cdots \wedge \theta_\ell$ be a minimum-sized $(\mathsf{AND} \circ \mathsf{AC}^0_{d-1})$ formula for f_A^u. By assumption, we have that $\sum_{j=1}^{\ell} |\theta_j| < \sum_{i \in A} |\phi_i^u|$. We can thus replace the $\bigwedge_{i \in A} \phi_i^u$ in the optimal formula for f^u with $\theta_1 \wedge \cdots \wedge \theta_\ell$ and get a smaller formula. In more detail, we have that

$$f^u = f_A^u \wedge (\bigwedge_{i \in [t] \setminus A} \phi_i^u) = (\theta_1 \wedge \cdots \wedge \theta_\ell) \wedge (\bigwedge_{i \in [t] \setminus A} \phi_i^u)$$

which is a formula of size

$$\sum_{j=1}^{\ell} |\theta_j| + \sum_{i \in [t] \setminus A} |\phi_i^u| < \sum_{i \in A} |\phi_i^u| + \sum_{i \in [t] \setminus A} |\phi_i^u| = \sum_{i=1}^{t} |\phi_i^u| = |\phi^u|$$

which contradicts the optimality of ϕ^u for f^u. ◀

For a string $x \in \{0,1\}^n$, let f_x^u be shorthand for $f_{A_x}^u$ where $A_x \subseteq [n]$ is the set of indices where x is one.

▶ **Proposition 3.7.** *Let $x \in \{0,1\}^n$. Then x has weight w if and only if the integer $\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f_x^u)$ is (k, rw)-good.*

Proof. By Lemma 3.6 and the fact that $|\phi_i^u|$ is (k,r)-good for all $i \in [n]$, we have that

$$\frac{\mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f_x^u)}{p^k} = \frac{\sum_{i \in A_x} |\phi_i^u|}{p^k} \equiv w \cdot r \mod p$$

where $A_x \subseteq [n]$ are bits of x that are ones. The “only if” part of the statement is guaranteed by the fact that $1 \leq r \leq p-1$ has a multiplicative inverse modulo p since p is prime. ◀

▶ **Theorem 3.8.** *Assume n is sufficiently large. Then there is a depth-8 AC^0 truth table oracle circuit C with $O(n^{250})$ wires such that $C^{(\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}}$ computes* MAJORITY *on n-bits.*

Proof. It suffices to show that for every $w \in [n]$, there exists a depth-7 AC^0 oracle circuit C_w with $O(n^{249})$ wires such that $C_w^{(\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}}(x) = 1 \iff \mathsf{wt}(x) = w$. Then $\mathsf{MAJORITY}(x) = \bigvee_{w \geq n/2} C_w(x)$.

Fix some $w \in [n]$. The circuit C_w works as follows. On input $x \in \{0,1\}^n$, first check if x is the all zeroes string. If so, then reject. Otherwise, compute the truth table of the direct product function $g_x : \{0,1\}^{\log u} \times \{0,1\}^{\log v} \to \{0,1\}$ given by $g_x(y,z) = f^u_x(y) \wedge f^v(z)$. Compute $s = \mathsf{CC}_{\mathsf{AC}^0_d}(g_x)$ in binary using oracle access to (AC^0_d)-MCSP. Finally accept if the integer s has the property that $s - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v)$ is (k, rw)-good. Reject otherwise.

We now verify this yields a (non-uniform) AC^0 truth table oracle circuit. We can check if x is the all zeroes string with a single OR gate. This requires one level of depth and $O(n)$ wires. Next, realize the jth bit in the truth table of g_x is either zero for all x or equal to

$$f^u_x(j) = \bigvee_{i \in [n] : \phi^u_i(j) = 1} x_i$$

where x_i denotes the ith bit of x. Thus, using non-uniformity, we can compute the truth table of g_x with $O(nuv) = O(nq^{60}) = O(n^{121})$ wires and depth-one. Next, we can compute $s = \mathsf{CC}_{\mathsf{AC}^0_d}(g_x)$ in binary with $O(uv\log(uv))$ calls to (AC^0_d)-MCSP using the fact that $\mathsf{CC}_{\mathsf{AC}^0_d}(g_x) \leq uv\log(uv)$ by the DNF bound and the fact that

$$\mathsf{CC}_{\mathsf{AC}^0_d}(g_x) = s \iff (\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}(g_x, s) = 1 \text{ and } (\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}(g_x, s-1) = 0.$$

This takes at most $\tilde{O}((uv)^2) = O(n^{241})$ wires, an additional three layers of depth, and $2uv\log(uv)$ oracle calls that all do not depend on each other. Finally, the DNF upper bound guarantees that $\mathsf{CC}_{\mathsf{AC}^0_d}(g_x) \leq uv\log(uv) \leq n^{61}$, so the length of the integer $s = \mathsf{CC}_{\mathsf{AC}^0_d}(g_x)$ in binary is at most $61\log n$. Therefore we can check if s has the property that $s - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v)$ is (k, rw)-good using a DNF with at most n^{62} wires and at most an additional two layers of depth. Combining all this yields a AC^0 circuit of depth-7 with at most $O(n^{241})$ wires and no directed path between oracle gates.

Next, we argue for correctness. Clearly, C_w rejects the all zero string, so assume $x \neq 0^n$. By Proposition 3.7, it suffices to show that, for $s = \mathsf{CC}_{\mathsf{AC}^0_d}(g_x)$,

$$s - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) = \mathsf{CC}_{\mathsf{AND} \circ \mathsf{AC}^0_{d-1}}(f^u_x).$$

We confirm that neither f^u_x nor f^v is the constant zero function, so that we can use the direct product theorems in Lemma 2.3.

▷ Claim 3.9. Neither f^u_x nor f^v is the constant zero function.

Proof. If f^v were the constant zero function, then $\mathsf{CC}_{\mathsf{AC}^0_d}(h^v) \leq \log v$ by DNF computation which contradicts that

$$\mathsf{CC}_{\mathsf{AC}^0_d}(h^v) - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) \geq v^{.25}.$$

Next, let $i \in [n]$ be a bit of x that is not zero. (Recall, we assumed that $x \neq 0^n$.) Then f^u_x has accepts every input that that ϕ^u_i accepts. For contradiction, suppose that ϕ^u_i had no ones. Then we can remove ϕ^u_i from the optimal formula $\phi^u = \phi^u_1 \wedge \ldots \phi^u_t$ for f^u and get a smaller formula for f^u which contradicts the optimality of ϕ^u. ◀

Next we show that the optimal AC^0_d formula for g_x has an AND gate on top.

▷ Claim 3.10. $\mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(g_x) > \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x)$. Consequently, $\mathsf{CC}_{\mathsf{AC}^0_d}(g_x) = \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x)$.

Proof. Let $\Delta = \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(g_x) - \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x)$. We need to show $\Delta > 0$.

$$\begin{aligned}
\Delta &\geq \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f^v) - \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) + \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f^u_x) && \text{(by Lemma 2.3)}\\
&\quad - \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)\\
&\geq (v)^{.24} + \mathsf{CC}_{\mathsf{OR}\circ\mathsf{AC}^0_{d-1}}(f^u_x) - \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x) \geq && \text{(by Proposition 3.2)}\\
&\geq (v)^{.24} - u\log u && \text{(by DNF bound on } f^u_x)\\
&\geq n^{50\cdot .24} - n^{10}\log(n^{10}) && \text{(by definition of } u \text{ and } v)\\
&> 0 && \text{(for sufficiently large } n)
\end{aligned}$$

◁

Using the claim we have that

$$\begin{aligned}
s - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) &= \mathsf{CC}_{\mathsf{AC}^0_d}(g_x) - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) && \text{(by definition)}\\
&= \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(g_x) - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) && \text{(by Claim 3.10)}\\
&= \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x) + \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) - \mathsf{CC}_{\mathsf{AC}^0_d}(f^v) && \text{(by Lemma 2.3)}\\
&= \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x) + \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) - \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^v) && \text{(by Proposition 3.2)}\\
&= \mathsf{CC}_{\mathsf{AND}\circ\mathsf{AC}^0_{d-1}}(f^u_x)
\end{aligned}$$

as desired. ◀

▶ Remark 3.11. We remark that the only time we use the failure of Lipchitzness in Case 2 is to show the existence of functions like f^u with high top fan-in and functions like f^v with a large difference between top AND gate and top OR gate complexity. Using known PARITY lower bounds and depth-hierarchy theorems for AC^0 circuits, we can unconditionally prove the existence of f^u and f^v respectively but with slightly worse parameters that would yield a quasi-polynomial reduction (at least in the $d \geq 3$ case) rather than the polynomial reduction we present.

4 On the NP-hardness of MOCSP

First, we introduce some useful notation and definitions. For a truth table T of length m and a set $P \subseteq [m]$, let T_P be the truth table of length m where the jth bit of T_P equals

$$\begin{cases} \text{the } j\text{th bit of } T & \text{, if } j \in P \\ 0 & \text{, otherwise.} \end{cases}$$

Next, we say a truth table T of length m is (s)-*irritable on a partition* $\mathcal{P} = (P_1, \ldots, P_n)$ *of* $[m]$ if for all $i \in [n]$

$$\mathsf{CC}^{T_{P_1}, \ldots, T_{P_{i-1}}, T_{P_{i+1}}, \ldots, T_{P_n}}(T) > s.$$

Finally, for a partition $\mathcal{P} = (P_1, \ldots, P_n)$ of $[m]$ and any set $S \subseteq [n]$, we define the $\mathcal{P}$*-lift of* S, denoted $S^{\mathcal{P}}$, to be the subset of $[m]$ given by

$$S^P = \bigcup_{i \in S} P_i.$$

Our first theorem shows that one can use an irritable truth table and MOCSP to approximate r-bounded set cover.

▶ **Theorem 4.1.** *Let $S_1, \ldots, S_t \subseteq [n]$ be sets of cardinality at most r that cover $[n]$. Let T be a truth table of length m, and let $\mathcal{P} = (P_1, \ldots, P_n)$ be a partition of $[m]$. Then*

- $\mathsf{CC}^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) \leq 2\ell$, *and*
- $\mathsf{CC}^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) > \ell/2$ *if T is (rn)-irritable on $\mathcal{P}$*

where ℓ is size of the optimal cover of $[n]$ by $S_1, \ldots, S_t$.

Proof. We split this proof into two claims to make clear that our two "without loss of generality" assumptions do not conflict with each other.

▷ Claim 4.2. $\mathsf{CC}^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) \leq 2\ell$

Proof. Without loss of generality, assume that the optimal cover size ℓ is witnessed by $S_1 \cup \cdots \cup S_\ell = [n]$. Then, by construction, the function computed by oracle circuit $T_{S_1^{\mathcal{P}}} \vee \cdots \vee T_{S_\ell^{\mathcal{P}}}$ of size $2\ell - 1$ is T. In more detail,

$$\bigvee_{i \in [\ell]} T_{S_i^{\mathcal{P}}} = \bigvee_{i \in [\ell]} \bigvee_{j \in S_i} T_{P_j} = \bigvee_{j \in S_1 \cup \cdots \cup S_\ell} T_{P_j} = \bigvee_{j \in [n]} T_{P_j} = T.$$

Therefore $\mathsf{CC}^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) \leq 2\ell$. ◁

▷ Claim 4.3. If T is (rn)-irritable on $\mathcal{P}$, then $\mathsf{CC}^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}} > \ell/2$.

Proof. For contradiction, suppose there is an oracle circuit D with at most $\ell/2$ gates such that $D^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}$ computes T. Since D has at most $q \leq \ell/2$ unique oracle gates, assume, without loss of generality, that $D^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_q^{\mathcal{P}}}}$ computes T.

Recall, by the definition of $T_{S_i^{\mathcal{P}}}$, we have that $T_{S_i^{\mathcal{P}}} = \bigvee_{j \in S_i} T_{P_j}$. Note that $\bigvee_{j \in S_i} T_{P_j}$ is an oracle circuit of size at most $2r$ since $|S_i| \leq r$. Thus, by replacing each $T_{S_i^{\mathcal{P}}}$ oracle gate in D with the oracle circuit $\bigvee_{j \in S_i} T_{P_j}$, we can transform D into an oracle circuit E of size at most $2r|C| \leq r\ell$ that computes T when given access to the oracles in the set $O = \{T_{P_j} : j \in S_1 \cup \cdots \cup S_q\}$. However, since the optimal cover of n is of size ℓ and $q \leq \ell/2 < \ell$, it follows that $|S_1 \cup \cdots \cup S_q| < n$ and hence $|O| < n$. Thus, there is an element $i^\star \in [n]$ such that $T_{P_{i^\star}} \notin O$. Therefore, the circuit E witnesses that

$$\mathsf{CC}^{T_{P_1}, \ldots, T_{P_{i^\star - 1}}, T_{P_{i^\star + 1}}, \ldots, T_{P_n}}(T) \leq r\ell \leq rn$$

which contradicts that T is (rn)-irritable on $\mathcal{P}$. ◁

◀

Moreover, given a truth table T and the partition $\mathcal{P}$ on which T is sufficiently irritable, we show it is easy to build a constant-depth circuit that approximates r-bounded set cover using MOCSP.

▶ **Theorem 4.4.** *Let $r \in \mathbb{N}$. There exists an polynomial-time algorithm A such that if $\mathcal{P} = (P_1, \ldots, P_n)$ is a partition of $[m]$, and T is a truth table of length m, then*

1. *$A(0^n, 0^t, T, \mathcal{P})$ outputs a depth-2 AC^0 circuit $C_{n,t,T,\mathcal{P}}$ with $O(mnrt)$ wires,*
2. *$\mathsf{MOCSP} \circ C_{n,t,T,\mathcal{P}}$ accepts all* YES *instances of* $\mathsf{SetCover}_{r,n,t}$ *, and*
3. *$\mathsf{MOCSP} \circ C_{n,t,T,\mathcal{P}}$ computes a 4-approximation to* $\mathsf{SetCover}_{r,n,t}$ *if T is (rn)-irritable on $\mathcal{P}$.*

Proof. First, we show that there is a polynomial-time algorithm A that outputs a small depth-2 circuit computing a specific function. Then we show that this specific function is helpful in computing $\mathsf{SetCover}_{r,n,t}$.

▷ Claim 4.5. There is a polynomial-time algorithm A such that $A(0^n, 0^t, T, \mathcal{P})$ outputs a depth-2 AC^0 circuit $C_{n,t,T,\mathcal{P}}$ with $O(mnrt)$ wires satisfying

$$C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t) = (T, 2c; T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}})$$

for any instance $(c, S_1, \ldots, S_t)$ of $\mathsf{SetCover}_{r,n,t}$.

Proof. On input $(0^n, 0^t, T, \mathcal{P})$, A builds the circuit $C_{n,t,T,\mathcal{P}}$ as follows. First, A will hardwire $C_{n,t,T,\mathcal{P}}$ to output T. This requires $O(m)$ wires and depth one. Next, A adds circuitry to $C_{n,t,T,\mathcal{P}}$ that outputs $2c$ by adding an extra zero to the binary expansion of c. This uses $O(\log n)$ wires and depth one.

Finally, A adds circuitry to $C_{n,t,T,\mathcal{P}}$ that outputs $T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}$ as follows. Observe that for any $i \in [t]$ and $j \in [m]$, the jth bit of $T_{S_i^{\mathcal{P}}}$ is one if and only if S_i contains the unique element $k_j \in [n]$ such that $j \in P_{k_j}$. Thus, since A has access to $\mathcal{P}$, A can calculate k_j for all $j \in [m]$ and then add circuitry to $C_{n,t,T,\mathcal{P}}$ that calculates the jth bit of $T_{S_i^{\mathcal{P}}}$ by ORing over all the elements of S_i and using an AND to check if any one of those elements is k_j. This requires $O(mnrt)$ wires and depth-two.

Therefore, $C_{n,t,T,\mathcal{P}}$ is a depth-2 AC^0 circuit with $O(mnrt)$ wires as desired. Moreover, it is clear from this description that A runs in polynomial-time. ◁

It remains to show that the algorithm A given in Claim 4.5 satisfies (2) and (3). Let $(c, S_1, \ldots, S_t)$ be an instance of $\mathsf{SetCover}_{r,n,t}$. Let ℓ be the minimum size of any cover of $[n]$ by $S_1, \ldots, S_t$.

First, we show (2) holds. Suppose $c \geq \ell$. Then Theorem 4.1 implies that

$$CC^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) \leq 2\ell \leq 2c$$

so

$$\mathsf{MOCSP}(C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t)) = \mathsf{MOCSP}(T, 2c; T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}) = \mathsf{YES}$$

as desired.

Finally, we show (3) holds. Suppose $c < \ell/4$ and T is (rn)-irritable. Then Theorem 4.1 implies that

$$CC^{T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}}(T) > \ell/2 \geq 2c$$

so

$$\mathsf{MOCSP}(C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t)) = \mathsf{MOCSP}(T, 2c; T_{S_1^{\mathcal{P}}}, \ldots, T_{S_t^{\mathcal{P}}}) = \mathsf{NO}.$$

Hence (3) holds. ◀

Of course to make use of Theorem 4.4, we need to actually find truth tables T and partitions $\mathcal{P}$ on which T is sufficiently irritable. Fortunately, such T and $\mathcal{P}$ are abundant. We show that, with high probability, any choice of $\mathcal{P}$ and a random choice of a truth table T suffices.

▶ **Lemma 4.6.** *Let $n, r \in \mathbb{N}$. Let m be the least power of two greater than n^3. Let $\mathcal{P} = (P_1, \ldots, P_n)$ be any partition of $[m]$ such that $|P_i| \geq m/n - 1$ for all $i \in [n]$. Pick a truth table $T \in \{0,1\}^m$ uniformly at random. Then T is (rn)-irritable on $\mathcal{P}$ except with probability $2^{-\Omega(n^2)}$.*

Proof. We prove this by bounding the probability that, for some fixed $i^\star$ and some fixed oracle circuit C,

$$C^{T_{P_1},\ldots,T_{P_{i^\star-1}},T_{P_{i^\star+1}},\ldots,T_{P_n}} \text{ computes } T$$

and then union bounding over all $i^\star$ and all oracle circuits of size at most rn.

Realize that the function computed by $C^{T_{P_1},\ldots,T_{P_{i^\star-1}},T_{P_{i^\star+1}},\ldots,T_{P_n}}$ does not depend on any of the bits in T that lie in $P_{i^\star}$. Therefore, since $|P_{i^\star}| \geq m/n - 1$, this means that the probability that $\mathcal{C}^{T_{P_1},\ldots,T_{P_{i^\star-1}},T_{P_{i^\star+1}},\ldots,T_{P_n}}$ computes T is at most $2^{-m/n+1} \leq 2^{-n^2+1}$.

Now, we union bound. Clearly, there are at most n choices of $i^\star$. Next, we need to count the number of C of size at most rn. For concision, let $s = rn$. We bound the number of oracle circuits of size s, allowing for identity gates to catch circuits of smaller size. For each of the s gates, there are $4 + n$ gate types to chose from (AND, OR, NOT, identity, and the n oracle gates). Then, for each of the s gates, we have to choose the at most $\log(m)$ wires that feed into that gate and there are at most $(s + \log(m))$ choices for where each of these wires comes from. Hence, we get a bound of

$$(4+n)^s(s+\log(m))^{s\log(m)}$$

whose logarithm is

$$s\log(4+n) + s\log(m)\log(s+\log(m)) = \tilde{O}(rn).$$

Thus, probability that T fails to be (rn)-irritable on $\mathcal{P}$ is at most

$$n2^{-n^2+\tilde{O}(rn)} \leq 2^{-\Omega(n^2)}.$$

◀

Thus, using a random choice of T gives us an NP-hardness result under RP-reductions.

▶ **Corollary 4.7.** NP $\leq_m^{\mathsf{RP}}$ MOCSP.

Proof. Let r be sufficiently large that computing a 4-approximation to r-bounded set cover is NP-hard (such an r exists by Theorem 2.6). We will reduce giving a 4-approximation of r-bounded set cover to MOCSP.

The reduction R works as follows. On an instance $(c, S_1, \ldots, S_t)$ of $\mathsf{SetCover}_{r,n,t}$, R first computes the integer m that is the least power of two greater than n^3. Next, R computes the partition $\mathcal{P} = (P_1, \ldots, P_n)$ of $[m]$ where for all $i \in [n]$

$$P_i = \{j \in [m] : j \equiv i \mod n\}.$$

Then, R picks a truth table T of length m uniformly at random. After that, R runs the algorithm A from Theorem 4.4 on the input $(0^n, 0^t, T, \mathcal{P})$ to obtain the circuit $C_{n,t,T,\mathcal{P}}$. Finally, R outputs $\mathsf{MOCSP}(C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t))$.

Now, we argue for correctness. Theorem 4.4 guarantees that $C_{n,t,T,\mathcal{P}}$ correctly answers all YES instances of $\mathsf{SetCover}_{r,n,t}$, so R also correctly answers all YES instances of $\mathsf{SetCover}_{r,n,t}$.

On the other hand, observe that our construction of $\mathcal{P}$ guarantees that $|P_i| \geq m/n - 1$ for all $i \in [n]$, so Lemma 4.6 implies that T is (rn)-irritable on $\mathcal{P}$ with high probability. Therefore, Theorem 4.1 further implies that with high probability $C_{n,t,T,\mathcal{P}}$ (and hence R) computes a 4-approximation to r-bounded set cover. ◀

Using more queries to MOCSP, we can improve the RP reduction to a ZPP reduction by checking if the randomly chosen T is indeed (rn)-irritable on $\mathcal{P}$.

▶ **Corollary 4.8.** NP $\leq_{tt}^{\mathsf{ZPP}}$ MOCSP.

Proof. Run the same reduction as in the proof of Corollary 4.7 except check whether T is (rn)-irritable on $\mathcal{P}$ using the MOCSP oracle. This can be done at the same time the as the MOCSP oracle answers $\mathsf{MOCSP}(C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t))$. If T is indeed (rn)-irritable on $\mathcal{P}$, then we know the output given by $\mathsf{MOCSP}(C_{n,t,T,\mathcal{P}}(c, S_1, \ldots, S_t))$ is correct using Theorem 4.1. Otherwise, output "don't know." ◀

We can also use non-uniform bits to provide the reduction with a truth table T and a partition $\mathcal{P}$ such that T is sufficiently irritable on P. This yields an AC^0 many-one reduction.

▶ **Corollary 4.9.** MOCSP *is* NP*-hard under (non-uniform)* AC^0 *many-one reductions.*

Proof. Let r be large enough that computing a 4-approximation to r-bounded set cover is NP-hard. It suffices to show that for all sufficiently large n and t, there is an AC^0 circuit C such that $\mathsf{MOCSP} \circ C$ computes a 4-approximation to $\mathsf{SetCover}_{r,n,t}$.

By Lemma 4.6 for sufficiently large n, there exists a truth table T of length $O(n^3)$ and a partition $\mathcal{P} = (P_1, \ldots, P_n)$ of $[m]$ such that T is (rn)-irritable on $\mathcal{P}$. Thus, letting A be the algorithm from Theorem 4.4, $A(0^n, 0^t, T, \mathcal{P})$ outputs a depth-2 AC^0 circuit C such that $\mathsf{MOCSP} \circ C$ computes a 4-approximation to $\mathsf{SetCover}_{r,n,t}$, as desired. ◀

At this point, one might begin to speculate whether we can prove that MOCSP is NP-hard under deterministic polynomial-time reductions. Unfortunately, this seems difficult. This is because Murray and Williams' [22] and Hitchcock and Pavan's [17] result that NP $\leq_{tt}^{\mathsf{P}}$ MCSP $\implies$ EXP $\neq$ ZPP also holds for MOCSP with essentially the same proof. For completeness, we give the MOCSP version of Murray and Williams' proof in Appendix B.

▶ **Theorem 4.10** (Essentially proved in [22] and [17])**.** *If* NP $\leq_{tt}^{\mathsf{P}}$ MOCSP*, then* EXP $\neq$ ZPP*.*

Still, it seems plausible to us that MOCSP is hard for NP under Turing reductions. Indeed, Theorem 4.4 implies that, to prove a P-Turing reduction, it suffices to show that that there is a polynomial time algorithm B, *with oracle access to* MOCSP, such that for all large n, $B(0^n)$ outputs a truth table T and a partition $\mathcal{P} = (P_1, \ldots, P_n)$ such that T is (rn)-irritable on $\mathcal{P}$. We stress that B has access to MOCSP, so B can actually check whether T is (rn)-irritable on $\mathcal{P}$ and make adjustments accordingly.

▶ **Conjecture 4.11.** NP $\leq_{\mathsf{T}}^{\mathsf{P}}$ MOCSP.

Maybe it is even possible to prove that such a B exists if $\mathsf{E} \not\subseteq$ i.o-$\mathsf{SIZE}[2^{O(n)}]$.

▶ **Open Question 4.12.** *Can one show that* $\mathsf{E} \not\subseteq$ *i.o-*$\mathsf{SIZE}[2^{O(n)}]$ *implies* NP $\leq_{\mathsf{T}}^{\mathsf{P}}$ MOCSP*?*

References

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

2 Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. *Information and Computation*, 256:2–8, 2017.

3 Eric Allender and Shuichi Hirahara. New insights on the (non-)hardness of circuit minimization and related problems. In *Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 54:1–54:14, 2017.

4 Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. *Computational Complexity*, 26(2):469–496, 2017.

5 Eric Allender, Rahul Ilango, and Neekon Vafa. The non-hardness of approximating circuit size. *Electronic Colloquium on Computational Complexity*, TR18-173, 2018.

6 Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*, 77(1):14 – 40, 2011.

7 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

8 Harry Buhrman and Leen Torenvliet. Randomness is hard. *SIAM Journal on Computing*, 30:200–1, 2000.

9 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Computational Complexity Conference (CCC)*, pages 10:1–10:24, 2016.

10 Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998.

11 Alexander Golovnev, Rahul Ilango, Russell Impagliazzo, Valentine Kabanets, Antonina Kolokolova, and Avishay Tal. AC0[p] lower bounds against MCSP via the coin problem. *Electronic Colloquium on Computational Complexity*, TR19-018, 2019.

12 G.H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Clarendon Press, Oxford, England, 5th edition, 1979.

13 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *Symposium on Foundations of Computer Science (FOCS)*, pages 247–258, 2018.

14 Shuichi Hirahara, Igor C. Oliveira, and Rahul Santhanam. NP-hardness of minimum circuit size problem for OR-AND-MOD circuits. In *Computational Complexity Conference (CCC)*, pages 5:1–5:31, 2018.

15 Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In *Computational Complexity Conference (CCC)*, pages 7:1–7:20, 2017.

16 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *Computational Complexity Conference (CCC)*, pages 18:1–18:20, 2016.

17 John Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, 2015.

18 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The Power of Natural Properties as Oracles. In *Computational Complexity Conference (CCC)*, pages 7:1–7:20, 2018.

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

20 William J. Masek. Some NP-complete set covering problems. Unpublished Manuscript, 1979.

21 Colin McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, (Norwich, 1989), London Math. Soc. Lecture Note Ser. 141 . Cambridge Univ. Press, Cambridge, (1989), pp. 48–188.

22 Cody D. Murray and R. Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Computational Complexity Conference (CCC)*, pages 365–380, 2015.

23 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

24 Igor C. Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds. *Electronic Colloquium on Computational Complexity*, TR18-158, 2018.

25 Igor C. Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness. In *Computational Complexity Conference (CCC)*, pages 18:1–18:49, 2017.

26 Igor C. Oliveira and Rahul Santhanam. Hardness magnification for natural problems. In *Symposium on Foundations of Computer Science (FOCS)*, pages 65–76, 2018.

27 A. A. Razborov. Lower bounds on the size of constant-depth networks over a complete basis with logical addition. *Mathematicheskie Zametki*, 41(4):598–607, 1987.

28 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.

29 R. Smolensky. On representations by low-degree polynomials. In *FOCS*, pages 130–138, 1993.

30 Boris Trakhtenbrot. A survey of Russian approaches to perebor (brute-force searches) algorithms. *IEEE Ann. Hist. Comput.*, 6(4):384–400, October 1984.

31 Luca Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 453–461, 2001.

32 Stanislav Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, 1983.

A MAJORITY reduces to (AC^0_d)-MCSP when Lipchitzness holds

Our goal in this section is to find a small (AC^0_d)-MCSP-oracle circuit that computes MAJORITY on n-bits for sufficiently large n. We can do this using the techniques of Golovnev *et al.* [11]. In order to make our proof relatively self-contained, we differ slightly from the presentation in [11]. In particular, our presentation follows a method for computing MAJORITY that is described in Shaltiel and Viola [28].

At a high-level, this procedure works by using the input string to sample a random variable whose circuit complexity spikes depending on the weight of the input and then using Lipchitzness to prove that this spike occurs with high enough probability that we can derandomize using non-uniformity.

Continuing the notation from Section 3, assume that there is an $m \in \{q^{10}, q^{50}\}$ such that $\mathsf{CC}_{\mathsf{AC}^0_d}$ is $(m^{.25})$-Lipchitz on inputs of length m.

We define the random variable $T_{p,m} \in \{0,1\}^m$ where each bit in $T_{p,m}$ is independently chosen to be one with probability p and zero with probability $1-p$.

▶ **Lemma A.1.** $\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p,m})] = \tilde{O}(pm)$ *if* $p \geq m^{-1/3}$

Proof. By Hoeffding's inequality we have that the probability that $T_{p,m}$ has greater than k ones is at most $\mathsf{exp}(-2\epsilon^2 m)$. Via computation by DNF, if a truth table $T \in \{0,1\}^m$ has at most k ones, $\mathsf{CC}_{\mathsf{AC}^0_d}(T) = k \log m = \tilde{O}(k)$. Similarly, we have that $\max\{C(T) : T \in \{0,1\}^m\} = \tilde{O}(m)$. Hence, we get that

$$\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p,m})] = \tilde{O}(k) + \tilde{O}(\mathsf{exp}(-2\epsilon^2 m)m) = \tilde{O}(pm + pm\epsilon + \mathsf{exp}(-2\epsilon^2 m)m).$$

If we set $\epsilon = \sqrt{\frac{\ln m}{2m}}$, then we have

$$\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p,m})] \leq \tilde{O}(pm + p\sqrt{m}\ln m + 1) \leq \tilde{O}(pm + \sqrt{m}\ln m)$$

Finally, if $p \geq 1/m^{1/3}$, we have $\mathbb{E}[T_{p,m}] = \tilde{O}(pm)$ as desired ◀

We will make use of the following concentration inequality.

▶ **Theorem A.2** (McDiarmid's "bounded differences inequality" [21]). *Let* $f : \{0,1\}^n \to \mathbb{R}$ *be c-Lipchitz. Let* $X_1, \ldots, X_n$ *be independent random variables with values in* $\{0,1\}$. *Let* $\mu = \mathbb{E}_{X_1,\ldots,X_n}[f(X_1,\ldots,X_n)]$. *Then*

$$\Pr[|f(X_1,\ldots,X_n) - \mu| \geq \epsilon] \leq 2\mathsf{exp}(-\frac{\epsilon^2}{nc^2}).$$

For $t \in \mathbb{N}$ and $w_1 \neq w_2 \in [t]$, we say a Boolean function $f : \{0,1\}^t \to \{0,1\}$ computes $\mathsf{WTDIS}_t[w_1, w_2]$ if $\mathsf{wt}(x) = w_1$ implies $f(x) = 1$ and $\mathsf{wt}(x) = w_2$ implies $f(x) = 0$. (WTDIS is short for weight distinguishing.)

▶ **Theorem A.3.** *If n is sufficiently large, then for all $1 \leq b \leq \sqrt{q}/2$, there exists a (non-uniform) NC^0 oracle circuit C with at most $O(n^{100})$ wires such that $C^{(\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}}$ computes $\mathsf{WTDIS}_q[w_1, w_1 + b]$ for some $w_1 \geq \sqrt{q}/2$. Moreover, C has a single gate.*

Proof. For $w \in [q]$, let $p_w = \frac{w}{2q}$. Let w_0 be the largest integer less than $\sqrt{q}$ such that $q - w_0$ is a multiple of b. (Note that $w_0 \geq \sqrt{q} - b \geq \sqrt{q}/2$).

By Lemma A.1, we have that

$$\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p_{w_0},m})] = \tilde{O}(\frac{\sqrt{q}}{q}m) = \tilde{O}(m/\sqrt{q}).$$

On the other hand, since $p_q = 1/2$, $T_{p_q,m}$ is just a binary string of length m picked uniformly at random, so the formula size lower bounds of Shannon and Riordan imply

$$\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p_q,m})] = \mathop{\mathbb{E}}_{x\in\{0,1\}^m}[C(x)] = \tilde{\Omega}(m)$$

(note that an AC^0_d formula of size s implies an unrestricted formula of size s). Hence, by an averaging argument there exists a $w_1 \geq w_0 \geq \sqrt{q}/2$ such that

$$\mathbb{E}[\mathsf{CC}_{\mathsf{AC}^0_d}(T_{p_{w_1+b},m})] - \mathbb{E}[\mathsf{CC}(T_{p_{w_1},m})] \geq \frac{\tilde{\Omega}(m) - \tilde{O}(m/\sqrt{q})}{q} = \tilde{\Omega}(m/q).$$

Let $t = \frac{\mathbb{E}[T_{p_{w_1+b},m}] + \mathbb{E}[T_{p_{w_1},m}]}{2}$. Then we have that $\mathbb{E}[T_{p_{w_1+b},m}] - t = \tilde{\Omega}(m/q)$ and $t - \mathbb{E}[T_{p_{w_1},m}] = \tilde{\Omega}(m/q)$.

We now outline a probabilistic oracle circuit D that we will later make into a deterministic NC^0 circuit. D takes as input a string $x \in \{0,1\}^n$ and takes as its random "inputs" strings $u_1, \ldots, u_m \in \{0,1\}^{\log q}$ and $v_1, \ldots, v_m \in \{0,1\}$. The reduction then computes the string $y := y_1 \ldots y_m$ where y_i is zero if v_i is zero and y_i is the u_ith bit of x if v_i is one (recall, q is a power of two). D then outputs (AC^0_d)-$\mathsf{MCSP}(y, t)$.

We now argue for correctness with high probability. Realize each y_i is independent with probability $\frac{\mathsf{wt}(x)}{2n}$ of being 1. Hence, y is just the random variable $T_{p_w,m}$ where $w = \mathsf{wt}(x)$. Hence, if $\mathsf{wt}(x) = w_1$, then

$$\Pr[R(x) \neq 1] = \Pr[\mathcal{C}(T_{p_{w_1},m}) > t]$$

Recall that $t - \mathbb{E}[T_{p_{w_1},m}] = \tilde{\Omega}(m/q)$ and, by assumption, $\mathsf{CC}_{\mathsf{AC}^0_d}$ on inputs of length m is $(m^{.25})$-Lipchitz, so by Theorem A.2, we have that this probability is bounded by

$$2\mathsf{exp}(-2\frac{\tilde{\Omega}(m^2)}{\tilde{O}(q^2 m^{1.5})}) \leq \mathsf{exp}(-2\frac{\tilde{\Omega}(q^{.5 \cdot 10})}{\tilde{O}(q^2)}) = O(\mathsf{exp}(-q^3))$$

using the fact that $m \geq q^{10}$. A similar analysis shows that the probability D errs if $\mathsf{wt}(x) = w_1 + b$ is at most $O(\mathsf{exp}(-q^3))$. This completes the analysis of D.

We now argue that this reduction can be derandomized using non-uniformity. For each input of weight either w_1 or $w_1 + b$, we have shown the fraction of random strings which err on that input is $O(\mathsf{exp}(-q^3))$. Hence, the fraction of random seeds which err on at least one input of weight w_1 or $w_1 + b$ is at most

$$2^q O(\mathsf{exp}(-q^3)) < 1$$

for large enough n. Thus, there exists some fixed $u_1, \ldots, u_m$ and $v_1, \ldots, v_m$ which work on all inputs of length q. Once we are (non-uniformly) given these $u_1, \ldots, u_m$ and $v_1, \ldots, v_m$ which work on all inputs, we can turn D into an NC^0 oracle circuit C which has just a single gate (an oracle gate) whose inputs are the fixed number t and the string y where each bit of y is either a fixed bit of x or zero. This yields a NC^0 oracle circuit with $O(m) = O(q^{50}) = O(n^{100})$ wires. ◀

▶ **Corollary A.4.** *If n is sufficiently large, then for all distinct $w_1, w_2 \in [n]$ there is an NC^0 oracle circuit C with at most two gates and $O(n^{100})$ wires such that $C^{(\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}}$ computes $\mathsf{WTDIS}_n[w_1, w_2]$.*

Proof. Fix some $w_1 \neq w_2$. Without loss of generality assume $w_1 < w_2$ (if this is not the case, then swap the names of w_1 and w_2 in this proof and add a NOT gate to the top of C). Let $b = w_2 - w_1$. Recall q is the least power of two such that $n \leq \sqrt{q}/2$. Note that $q = \Theta(n^2)$ and $b \leq n \leq \sqrt{q}/2$. Theorem A.3 guarantees there exists an NC^0 oracle circuit D of size $O(n^{20})$ such that D^{MCSP} computes $\mathsf{WTDIS}_q[w_3, w_3 + b]$ for some $w_3 \geq \sqrt{q}/2 \geq n$. Finally, let C be the oracle circuit that on input x outputs $D(y)$ where $y = 1^{w_3 - w_1} 0^{q - n - w_3 + w_1} x$. The correctness of this output is guaranteed by the fact that $\mathsf{wt}(y) = w_3$ if and only if $\mathsf{wt}(x) = w_1$ and $\mathsf{wt}(y) = w_3 + b$ if and only if $\mathsf{wt}(x) = w_2$. ◀

▶ **Corollary A.5.** *If n is sufficiently large, then there exists a depth-4 AC^0 truth table oracle circuit C with $O(n^{102})$ wires such that $C^{(\mathsf{AC}^0_d)\text{-}\mathsf{MCSP}}$ computes $\mathsf{MAJORITY}$ on strings on length n.*

Proof. It suffices to show that, for all $w \in [n]$, one can check if a string $x \in \{0,1\}^n$ has weight w using a depth-3 AC^0 truth-table oracle circuit C_w of size $O(n^{101})$. If one is able to do this, then $\mathsf{MAJORITY}$ is computed by $\bigvee_{w \geq n/2} C_w(x)$.

For $w \in [n]$, let $\mathsf{wt}_w : \{0,1\}^n \to \{0,1\}$ be the Boolean function that outputs one if and only if its input is a string of weight w. Now fix some $w \in [n]$. We claim that

$$\mathsf{wt}_w(x) = \bigwedge_{w' \in [n] : w \neq w'} \mathsf{WTDIS}_n[w, w']$$

If x has weight w, then $\mathsf{WTDIS}_n[w, w'](x) = 1$ for all $w' \neq w$, so

$$\mathsf{wt}_{w'}(x) = 1 = \bigwedge_{w' \in [n] : w \neq w'} \mathsf{WTDIS}_n[w, w'].$$

On the other hand, if x has weight $w' \neq w$,then $\mathsf{WTDIS}_n[w, w'](x) = 0$, so

$$\mathsf{wt}_w(x) = 0 = \bigwedge_{w' \in [n] : w \neq w'} \mathsf{WTDIS}_n[w, w'].$$

Finally, by Corollary A.4 we have that $\bigwedge_{w \in [n] : w \neq w'} \mathsf{WTDIS}_n[w', w]$ is computable by a depth-3 AC^0 truth table oracle circuit with $O(n^{101})$ wires. ◀

B $\mathsf{NP} \leq^{\mathsf{P}}_{tt} \mathsf{MOCSP}$ implies $\mathsf{EXP} \neq \mathsf{ZPP}$

The proof of this result follows essentially exactly from Murray and Williams's [22] proof for MCSP. For completeness, we replicate the proof here (even using their words and structure).

▶ **Proposition B.1.** *If* $\mathsf{NP} \leq^{\mathsf{P}}_{tt} \mathsf{MOCSP}$, *then* $\mathsf{EXP} \subseteq \mathsf{P/poly}$ *implies* $\mathsf{EXP} = \mathsf{NEXP}$.

Proof. Assume $\mathsf{NP} \le_{tt}^{\mathsf{P}} \mathsf{MOCSP}$ and $\mathsf{EXP} \subseteq \mathsf{P/poly}$. Let $L \in \mathsf{NTIME}(2^{n^c})$ for some $c \geq 1$. It suffices to show that $L \in \mathsf{EXP}$.

We pad L into the $L' = \{x01^{2^{|x|^c}} : x \in L\}$. Note that $L' \in \mathsf{NP}$. Hence there is a polynomial-time truth table reduction from L' to MOCSP. Composing the reduction from L to L' with the reduction from L' to MOCSP, we get a $2^{c'n^c}$-time truth table reduction R from n-bit instances of L to $2^{c'n^c}$-bit instances of MOCSP for some constant c'.

Let $Q(x)$ denote the concatenated string of all MOCSP queries produced by R in order on input x. Define the language

$$BITS_Q := \{(x, i) : \text{ the } i\text{th bit of } Q(x) \text{ is } 1\}$$

$BITS_Q$ is clearly in EXP. Since $\mathsf{EXP} \subseteq \mathsf{P/poly}$, for some $d \geq 1$ there is a circuit family C_n of size at most $n^d + d$ computing $BITS_Q$ on n-bit inputs.

Thus, on a given instance x, we have $\mathsf{CC}(Q(x)) \leq s(|x|)$ where $s(|x|) := (|x| + 2c'|x|^c)^d + d$. Therefore, every MOCSP query $(T, s', T_1, \ldots, T_t)$ produced by the reduction R on input x satisfies

$$\mathsf{CC}^{T_1,\ldots,T_t}(T) \leq \mathsf{CC}(T) \leq e \cdot \mathsf{CC}(Q(x)) \leq e \cdot s(|x|)$$

for some constant e since T is a substring of $Q(x)$ (see Lemma 2.2 in [22] for a proof of this substring fact). This leads to the following exponential time algorithm for L:

> On input x, run the exponential-time reduction $R(x)$ by using the following procedure for answering each MOCSP oracle query $(T, s'; T_1, \ldots, T_t)$. If $s' > e \cdot s(|x|)$, then respond YES to the query. Otherwise, cycle through every oracle circuit E of size at most s'. If $E^{T_1,\ldots,T_t}$ computes T, then respond YES. If no such E is found, then respond NO.

It suffices to show the procedure for answering MOCSP oracle queries runs in exponential time. Let $n = |x|$. First, we need to count the number of oracle circuits E on $(\log|T| \leq c'n^c)$-inputs with size at most $s(n)$ As shown in Lemma 4.6, the logarithm of the number of oracle circuit of size at most $s(|x|)$ on $(c'n^c)$-inputs with t oracle functions is at most

$$O(s(n)\log(4 + t) + s(|x|)\log(c'n^c)\log(s(|x|) + \log(c'n^c))).$$

Since $t \leq 2^{c'n^c}$ and s is polynomial in n, it is easy to see that the number of such circuits E is at most exponential. Second, one can check if an oracle circuit E satisfies $E^{T_1,\ldots,T_t}$ computes T in time polynomial in $(|E| + |T| + |T_1| + \cdots + |T_t|)$ and hence exponential in n. As a result, $L \in \mathsf{EXP}$, completing the proof. ◀

▶ **Theorem B.2.** *If* $\mathsf{NP} \le_{tt}^{\mathsf{P}} \mathsf{MOCSP}$*, then* $\mathsf{EXP} \neq \mathsf{NP} \cap \mathsf{P/poly}$*. Consequently,* $\mathsf{EXP} \neq \mathsf{ZPP}$*.*

Proof. For contradiction, suppose $\mathsf{NP} \le_{tt}^{\mathsf{P}} \mathsf{MOCSP}$ and $\mathsf{EXP} = \mathsf{NP} \cap \mathsf{P/poly}$. Then by Proposition B.1 $\mathsf{NEXP} \subseteq \mathsf{EXP} \subseteq \mathsf{NP}$ contradicting the nondeterministic time hierarchy theorem [32]. ◀

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il