One-Tape Turing Machine and Branching Program Lower Bounds for MCSP

Mahdi Cheraghchi
Department of EECS, University of Michigan, Ann Arbor, MI, USA

Shuichi Hirahara
National Institute of Informatics, Tokyo, Japan

Dimitrios Myrisiotis
Department of Computing, Imperial College London, London, UK

Yuichi Yoshida
National Institute of Informatics, Tokyo, Japan

Abstract

For a size parameter $s\colon \mathbb{N} \to \mathbb{N}$, the Minimum Circuit Size Problem (denoted by MCSP$[s(n)]$) is the problem of deciding whether the minimum circuit size of a given function $f\colon \{0,1\}^n \to \{0,1\}$ (represented by a string of length $N := 2^n$) is at most a threshold $s(n)$. A recent line of work exhibited "hardness magnification" phenomena for MCSP: A very weak lower bound for MCSP implies a breakthrough result in complexity theory. For example, McKay, Murray, and Williams (STOC 2019) implicitly showed that, for some constant $\mu_1 > 0$, if MCSP$[2^{\mu_1 \cdot n}]$ cannot be computed by a one-tape Turing machine (with an additional one-way read-only input tape) running in time $N^{1.01}$, then P $\neq$ NP.

In this paper, we present the following new lower bounds against one-tape Turing machines and branching programs:

1. A randomized two-sided error one-tape Turing machine (with an additional one-way read-only input tape) cannot compute MCSP$[2^{\mu_2 \cdot n}]$ in time $N^{1.99}$, for some constant $\mu_2 > \mu_1$.
2. A non-deterministic (or parity) branching program of size $o(N^{1.5}/\log N)$ cannot compute MKTP, which is a time-bounded Kolmogorov complexity analogue of MCSP. This is shown by directly applying the Nečiporuk method to MKTP, which previously appeared to be difficult.
3. The size of any non-deterministic, co-non-deterministic, or parity branching program computing MCSP is at least $N^{1.5-o(1)}$.

These results are the first non-trivial lower bounds for MCSP and MKTP against one-tape Turing machines and non-deterministic branching programs, and essentially match the best-known lower bounds for any explicit functions against these computational models.

The first result is based on recent constructions of pseudorandom generators for read-once oblivious branching programs (ROBPs) and combinatorial rectangles (Forbes and Kelley, FOCS 2018; Viola 2019). En route, we obtain several related results:

1. There exists a (local) hitting set generator with seed length $\widetilde{O}(\sqrt{N})$ secure against read-once polynomial-size non-deterministic branching programs on N-bit inputs.
2. Any read-once co-non-deterministic branching program computing MCSP must have size at least $2^{\widetilde{\Omega}(N)}$.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Minimum Circuit Size Problem, Kolmogorov Complexity, One-Tape Turing Machines, Branching Programs, Lower Bounds, Pseudorandom Generators, Hitting Set Generators

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.23

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2020/103/

Funding *Mahdi Cheraghchi*: M. Cheraghchi's research is supported in part by the NSF award CCF-2006455.

38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021).
Editors: Markus Bläser and Benjamin Monmege; Article No. 23; pp. 23:1–23:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements We would like to express our gratitude to Emanuele Viola and Osamu Watanabe for bringing to our attention the works by Kalyanasundaram and Schnitger [26] and Watanabe [39], respectively, and for helpful discussions. In particular, we thank Emanuele Viola for explaining to us his works [15, 38]. We thank Rahul Santhanam for pointing out that Nečiporuk's method can be applied to not only MKtP but also MKTP. We thank Chin Ho Lee for answering our questions regarding his work [27]. We thank Paul Beame for bringing his work [6] to our attention. We thank Valentine Kabanets, Zhenjian Lu, Igor C. Oliveira, and Ninad Rajgopal for illuminating discussions. Finally, we would like to thank the anonymous reviewers for their constructive feedback.

1 Introduction

The Minimum Circuit Size Problem (MCSP) asks whether a given Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ can be computed by some Boolean circuit of size at most a given threshold s. Here the function f is represented by the truth table of f, i.e., the string of length $N := 2^n$ that is obtained by concatenating all the outputs of f. For a size parameter $s\colon \mathbb{N} \to \mathbb{N}$, its parameterized version is denoted by MCSP$[s]$: That is, MCSP$[s]$ asks if the minimum circuit size of a function $f\colon \{0,1\}^n \to \{0,1\}$ is at most $s(n)$.

MCSP is one of the most fundamental problems in complexity theory, because of its connection to various research areas, such as circuit complexity [35, 25, 22, 31, 21, 2], learning theory [8], and cryptography [35, 16, 18]. It is easy to see that MCSP $\in$ NP because, given a circuit C of size s as an NP certificate, one can check whether C computes the given function f in time $N^{O(1)}$. On the other hand, its NP-completeness is a long-standing open question, which dates back to the introduction of the theory of NP-completeness (cf. [4]), and it has an application to the equivalence between the worst-case and average-case complexity of NP (cf. [18]).

Recently, a line of work exhibited surprising connections between very weak lower bounds of MCSP and important open questions of complexity theory, informally termed as "hardness magnification" phenomena. Oliveira and Santhanam [34] (later with Pich [33]) showed that, if an approximation version of MCSP cannot be computed by a circuit of size $N^{1.01}$, then $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ (in particular, $\mathsf{P} \neq \mathsf{NP}$ follows). Similarly, McKay, Murray, and Williams [30] showed that, if MCSP$[s(n)]$ cannot be computed by a 1-pass streaming algorithm of poly$(s(n))$ space and poly$(s(n))$ update time, then $\mathsf{P} \neq \mathsf{NP}$. Therefore, in order to obtain a breakthrough result, it is sufficient to obtain a very weak lower bound for MCSP.

Are hardness magnification phenomena plausible approaches for resolving the P versus NP question? We do not know the answer yet. However, it should be noted that, as argued in [3, 34], hardness magnification phenomena appear to bypass the *natural proof* barrier of Razborov and Rudich [35], which is one of the major barriers of complexity theory for resolving the P versus NP question. Most of lower bound proof techniques of complexity theory are "natural" in the following sense: Given a lower bound proof for a circuit class $\mathfrak{C}$, one can interpret it as an efficient average-case algorithm for solving $\mathfrak{C}$-MCSP (i.e., one can efficiently decide whether a given Boolean function f can be computed by a small $\mathfrak{C}$-circuit when the input f is chosen uniformly at random; cf. Hirahara and Santhanam [20]). Razborov and Rudich [35] showed that such a "natural proof" technique is unlikely to resolve $\mathsf{NP} \not\subseteq \mathsf{P/poly}$; thus we need to develop fundamentally new proof techniques. There seems to be no simple argument that naturalizes proof techniques of hardness magnification phenomena; hence, investigating hardness magnification phenomena could lead us to a new non-natural proof technique.

1.1 Our results

1.1.1 Lower bounds against one-tape Turing machines

Motivated by hardness magnification phenomena, we study the time required to compute MCSP by using a one-tape Turing machine. We first observe that the hardness magnification phenomena of [30] imply that a barely superlinear time lower bound for a one-tape Turing machine is sufficient for resolving the P versus NP question.

▶ **Theorem 1** (A corollary of McKay, Murray, and Williams [30]; see the full version)**.** *There exists a small constant* $\mu > 0$ *such that if* $\mathrm{MCSP}[2^{\mu\cdot n}] \notin \mathsf{DTIME}_1\left[N^{1.01}\right]$*, then* $\mathsf{P} \neq \mathsf{NP}$*.*

Here, we denote by $\mathsf{DTIME}_1[t(N)]$ the class of languages that can be computed by a Turing machine equipped with a one-way read-only input tape and a two-way read/write work tape running in time $O(t(N))$ on inputs of length N. We note that it is rather counter-intuitive that there is a *universal* constant $\mu > 0$; it is instructive to state Theorem 1 in the following logically equivalent way: If $\mathrm{MCSP}[2^{\mu\cdot n}] \notin \mathsf{DTIME}_1\left[N^{1.01}\right]$ for *all constants* $\mu > 0$, then $\mathsf{P} \neq \mathsf{NP}$.[1]

One of our main results is a nearly quadratic lower bound on the time complexity of a *randomized* one-tape Turing machine (with one additional read-only one-way input tape) computing MCSP.

▶ **Theorem 2.** *There exists some constant* $0 < \mu < 1$ *such that* $\mathrm{MCSP}[2^{\mu\cdot n}]$ *is not in* $\mathsf{BPTIME}_1\left[N^{1.99}\right]$*.*

Here, $\mathsf{BPTIME}_1[t(N)]$ denotes the class of languages that can be computed by a *two-sided-error randomized* Turing machine equipped with a one-way read-only input tape and a two-way read/write work tape running in time $t(N)$ on inputs of length N; we say that a two-sided-error randomized algorithm *computes* a problem if it outputs a correct answer with high probability (say, with probability at least 2/3) over the internal randomness of the algorithm.

Previously, no non-trivial lower bound on the time complexity required for computing MCSP by a Turing machine was known. Moreover, Theorem 2 essentially matches the best-known lower bound for this computational model; namely, the lower bound due to Kalyanasundaram and Schnitger [26], who showed that Element Distinctness is not in $\mathsf{BPTIME}_1\left[o\left(N^2/\log N\right)\right]$.

Our lower bound against $\mathsf{BPTIME}_1\left[N^{1.99}\right]$ is much stronger than the required lower bound (i.e, $\mathsf{DTIME}_1\left[N^{1.01}\right]$) of the hardness magnification phenomenon of Theorem 1. However, Theorem 2 falls short of the hypothesis of the hardness magnification phenomenon of Theorem 1 because of the choice of the size parameter. In the hardness magnification phenomenon, we need to choose the size parameter to be $2^{\mu\cdot n}$ for some small constant $\mu > 0$, whereas, in our lower bound, we will choose μ to be some constant close to 1. That is, what is missing for proving $\mathsf{P} \neq \mathsf{NP}$ is to decrease the size parameter from $2^{(1-o(1))\cdot n}$ to $2^{o(n)}$ in Theorem 2, or to increase the size parameter from $2^{o(n)}$ to $2^{(1-o(1))\cdot n}$ in Theorem 1.

Next, we investigate the question of whether hardness magnification phenomena on $\mathrm{MCSP}[s(n)]$ such as Theorem 1 can be proved when the size parameter $s(n)$ is large, as posed by Chen, Jin, and Williams [10]. As observed in [9], most existing proof techniques on hardness magnification phenomena are shown by constructing an oracle algorithm which makes short queries to some oracle. For example, behind the hardness magnification

[1] Observe that $\exists\mu, (P(\mu) \Rightarrow Q)$ is logically equivalent to $\exists\mu, (\neg P(\mu) \vee Q)$, which is equivalent to $\neg(\forall\mu, P(\mu)) \vee Q$.

phenomena of Theorem 1 is a nearly-linear-time oracle algorithm that solves MCSP$[2^{o(n)}]$ by making queries of length $2^{o(n)}$ to some PH oracle (see Corollary 18 for a formal statement). Chen, Hirahara, Oliveira, Pich, Rajgopal, and Santhanam [9] showed that most lower bound proof techniques can be generalized to such an oracle algorithm, thereby explaining the difficulty of combining hardness magnification phenomena with lower bound proof techniques. Following [9], we observe that our lower bound (Theorem 3) can be generalized to a lower bound against an oracle algorithm which makes short queries.

▶ **Theorem 3.** *Let $O \subseteq \{0,1\}^*$ be any oracle. Then, for every constant $1/2 < \mu < 1$,* MCSP$[2^{\mu \cdot n}]$ *on truth tables of size $N := 2^n$ is not in* $\mathsf{BPTIME}_1^O\left[N^{1+\mu'}\right]$ *for some constant $\mu' > 0$, where all of the strings queried to O are of length $N^{o(1)}$.*

Theorem 3 can be seen as a partial answer to the question posed by [10]: It is impossible to extend the hardness magnification phenomena of Theorem 1 to MCSP$[2^{\mu n}]$ for $\mu > 1/2$ by using similar techniques used in [30]. Recall that the proof techniques behind [30] are to construct a nearly-linear-time oracle algorithm that solves MCSP$[2^{\mu n}]$ by making short queries to some oracle; the existence of such an oracle algorithm is ruled out by Theorem 3 when $\mu > 1/2$. Therefore, in order to obtain a hardness magnification phenomenon for MCSP$[2^{0.51n}]$, one needs to develop a completely different proof technique that does not rely on constructing an oracle algorithm that makes short queries.

1.1.2 Lower bounds against branching programs

Another main result of this work is a lower bound against non-deterministic branching programs. We make use of *Nečiporuk's method*, which is a standard proof technique for proving a lower bound against branching programs. However, it appeared previously that Nečiporuk's method is not directly applicable to the problems such as MCSP [20]. In this paper, we develop a new proof technique for applying Nečiporuk's method to a variant of MCSP, called MKTP. MKTP is the problem of deciding whether $\mathrm{KT}(x) \leq s$ given (x, s) as input. Here $\mathrm{KT}(x)$ is defined as the minimum, over all programs M and integers t, of $|M| + t$ such that, for every i, M outputs the i-th bit of x in time t given an index i as input [1]. We prove lower bounds against general branching programs and non-deterministic branching programs by using Nečiporuk's method.

▶ **Theorem 4.** *The size of a branching program computing* MKTP *is at least $\Omega(N^2/\log^2 N)$. The size of a non-deterministic branching program or a parity branching program computing* MKTP *is at least $\Omega(N^{1.5}/\log N)$.*

Theorem 4 gives the first non-trivial lower bounds against non-deterministic and parity branching programs for MKTP and, in addition, these are the best lower bounds which can be obtained by using Nečiporuk's method (cf. [6]). Previously, by using a pseudorandom generator for branching programs constructed by [23], it was shown in [33, 11] that (deterministic) branching programs requires $N^{2-o(1)}$ size to compute MCSP and MKTP.[2] However, it is not known whether there is a pseudorandom generator for non-deterministic or parity branching programs. As a consequence, no non-trivial lower bound for MKTP (nor its exponential-time version denoted by MKtP) against these models was known before. Surprisingly, Theorem 4 is proved without using a pseudorandom generator nor a weaker

[2] It is worthy of note that Theorem 4 mildly improves the lower bounds of [33, 11] to $\Omega(N^2/\log^2 N)$ by directly applying Nečiporuk's method, which matches the state-of-the-art lower bound for any explicit function up to a constant factor.

object called a hitting set generator. We emphasize that it is surprising that a lower bound for MKtP can be obtained without using a hitting set generator; indeed, the complexity of MKtP is closely related to a hitting set generator, and in many settings (especially when the computational model is capable of computing XOR), a lower bound for MKtP and the existence of a hitting set generator are equivalent [18, 19].

The proof technique of Theorem 4 is applicable to problems of computing various resource-bounded Kolmogorov complexity measures, such as MKtP. However, we fail to apply Nečiporuk's method to MCSP, despite that circuit complexity can also be regarded as a version of resource-bounded Kolmogorov complexity. The KT-complexity of the truth table of a function f and the minimum circuit size of f are polynomially related to each other [1]; unfortunately, the relationship between circuit complexity and KT-complexity is not tight enough for our argument to work. Nevertheless, we were able to use a different approach to present the first non-trivial lower bound for MCSP against non-deterministic branching programs.

▶ **Theorem 5.** *The size of any non-deterministic, co-non-deterministic, or parity branching program computing* MCSP *is at least* $N^{1.5-o(1)}$.

The proof of Theorem 5 is based on a pseudorandom generator construction of Impagliazzo, Meka, and Zuckerman [23]. We show that their construction actually provides a pseudorandom generator of seed length $s^{2/3+o(1)}$ that fools non-deterministic, co-non-deterministic, and parity branching programs of size s.

Along the way, we obtain several new results regarding a lower bound for MCSP and a hitting set generator. A *hitting set generator* (HSG) $H\colon \{0,1\}^{\lambda(N)} \to \{0,1\}^N$ for a circuit class $\mathfrak{C}$ is a function such that, for any circuit C from $\mathfrak{C}$ that accepts at least $(1/2)\cdot 2^N$ strings of length N, there exists some seed $z \in \{0,1\}^{\lambda(N)}$ such that C accepts $H(z)$. We present a hitting set generator secure against read-once non-deterministic branching programs, based on a pseudorandom generator constructed by Forbes and Kelley [13].

▶ **Theorem 6.** *There exists an explicit construction of a (local) hitting set generator* $H\colon \{0,1\}^{\widetilde{O}\left(\sqrt{N\cdot\log s}\right)} \to \{0,1\}^N$ *for read-once non-deterministic branching programs of size* s.

Previously, Andreev, Baskakov, Clementi, and Rolim [5] constructed a hitting set generator with non-trivial seed length for read-k-times non-deterministic branching programs, but their seed length is as large as $N - o(N)$. Theorem 6 improves the seed length to $\widetilde{O}(\sqrt{N\cdot\log s})$. As an immediate corollary, we obtain a lower bound for MCSP against read-once non-deterministic branching programs.

▶ **Corollary 7.** *Any read-once co-non-deterministic branching program that computes* MCSP *must have size at least* $2^{\widetilde{\Omega}(N)}$.

1.2 Our techniques

1.2.1 Local HSGs for MCSP lower bounds

For a circuit class $\mathfrak{C}$, a general approach for obtaining a $\mathfrak{C}$-lower bound for MCSP is by constructing a "local" hitting set generator (or a pseudorandom generator (PRG), which is a stronger notion) secure against $\mathfrak{C}$. Here, we say that a function $G\colon \{0,1\}^s \to \{0,1\}^N$ is *local* if, for every z, the ith bit of $G(z)$ is "easy to compute" from the index i; more precisely, for every seed z, there exists some circuit C of size at most s such that C outputs the ith bit of $G(z)$ on input $i \in [N]$. Note here that $G(z)$ is a YES instance of MCSP[s], whereas a string w

chosen uniformly at random is a NO instance of MCSP[s] with high probability. This means that any $\mathfrak{C}$-algorithm that computes MCSP[s] distinguishes the pseudorandom distribution $G(z)$ from the uniform distribution w, and hence the existence of $\mathfrak{C}$-algorithm for MCSP[s] implies that there exists no local hitting set generator secure against $\mathfrak{C}$. This approach has been used in several previous works, e.g., [35, 1, 20, 11]. In fact, it is worthy of note that, in some sense, this is *the only approach* – at least for a general polynomial-size circuit class $\mathfrak{C} = \mathsf{P/poly}$, because Hirahara [18] showed that a lower bound for an approximation version of MCSP is equivalent to the existence of a local HSG.

At the core of our results is the recent breakthrough result of Forbes and Kelley [13], who constructed the first pseudorandom generator with $\mathsf{polylog}(n)$ seed length that fools unknown-order read-once oblivious branching programs. Viola [38] used their construction to obtain a pseudorandom generator that fools *deterministic* Turing machines (DTMs). Herein, we generalize his result to the case of *randomized* Turing machine (RTMs), and the case of *two-sided-error* randomized Turing machine ($\mathsf{BPTIME}_1[t(N)]$).[3] At a high level, our crucial idea is that Viola's proof does not exploit the uniformity of Turing machines, and hence a good coin flip sequence of a randomized oracle algorithm and all of its [small enough] oracle queries and corresponding answers can be fixed as non-uniformity (Lemma 22). In addition, by a careful examination of the Forbes-Kelley PRG, we show that their PRG is local; this gives rise to a local PRG that fools $\mathsf{BPTIME}_1[t(N)]$, which will complete a proof of our main result (Theorem 3).

We note that the proof above implicitly shows an exponential-size lower bound for MCSP against read-once oblivious branching programs, which was previously not known. Corollary 7 generalizes this lower bound to the case of co-non-deterministic read-once (not necessarily oblivious) branching program. In order to prove this, we make use of PRGs that fool combinatorial rectangles (e.g., [13, 27]). We present a general transformation from a PRG for combinatorial rectangles into a HSG for non-deterministic read-once branching program, by using the proof technique of Borodin, Razborov, and Smolensky [7]; see Theorem 6.

1.2.2 Nečiporuk's method for MKTP lower bounds

In order to apply Nečiporuk's method to MKTP, we need to give a lower bound on the number of distinct subfunctions that can be obtained by fixing all but $O(\log n)$ bits.

The idea of counting distinct subfunctions of MKTP is to show that a random restriction which leaves $O(\log n)$ variables free induces different subfunctions with high probability. Specifically, partition the input variables $[n]$ into $m := n/O(\log n)$ blocks, pick $m-1$ strings $\rho := \rho_2 \cdots \rho_m \in (\{0,1\}^{O(\log n)})^{m-1}$ randomly, and consider the restricted function $f\lceil_\rho(\rho_1) := \mathrm{MKTP}(\rho_1\rho_2\cdots\rho_m, \theta)$ for some threshold function θ to be chosen later. Then, the string $\rho_i\rho_2\cdots\rho_m$ is compressible when $i \in \{2,\cdots,m\}$ whereas the string $\rho_1\rho_2\cdots\rho_m$ is not compressible when ρ_1 is chosen randomly. This holds as, in the former case, there exists a $k \in \{2,\ldots,m\}$ such that $\rho_i = \rho_k$ and this yields a description for the string $\rho_i\rho_2\cdots\rho_m$ that is shorter than most of its descriptions in the latter case. Let now θ be an upper bound on the KT complexity of $\rho_i\rho_2\cdots\rho_m$ in the case where $i \in \{2,\cdots,m\}$. Therefore, $f\lceil_\rho(\rho_i) = 1$ for any ρ and $i \in \{2,\cdots,m\}$, and $f\lceil_\rho(\rho_1) = 0$ with high probability over random ρ and ρ_1. This implies that, with high probability over the random restrictions ρ and ρ', it is the case that $f\lceil_\rho \not\equiv f\lceil_{\rho'}$. This is so as, for every $i \in \{2,\ldots,m\}$, the probability over the random

[3] We emphasize that the notion of PRGs secure against these three computational models is different. See Definition 11, Definition 13, and Lemma 15.

restrictions ρ and ρ' that the string ρ_i is such that $f{\restriction_{\rho'}}(\rho_i) = f{\restriction_{\rho}}(\rho_i)$ is small, by the fact that $f{\restriction_{\rho}}(\rho_i) = 1$ for any ρ and the fact that $f{\restriction_{\rho'}}(\rho_i) = 0$ with high probability over random ρ_i and ρ' [and therefore with high probability over random ρ and ρ' as well].

Unfortunately, the probability that $f{\restriction_{\rho}} \equiv f{\restriction_{\rho'}}$ holds may not be exponentially small. As a consequence, a lower bound on the number of distinct subfunctions that can be directly obtained from this fact may not be exponential. In contrast, we need to prove an exponential lower bound on the number of distinct subfunctions in order to obtain the state-of-the-art lower bound via Nečiporuk's method.

In order to make the argument work, we exploit symmetry of information for (resource-unbounded) Kolmogorov complexity and Kolmogorov-randomness. Instead of picking ρ and ρ' randomly, we keep a set P which contains restrictions ρ that induce distinct subfunctions. Starting from $P := \emptyset$, we add one Kolmogorov-random restriction ρ to P so that the property of P is preserved. By using symmetry of information for Kolmogorov complexity, we can argue that one can add a restriction to P until P becomes as large as $2^{\Omega(n)}$, which proves that the number of distinct subfunctions of MKTP is exponentially large. Details can be found in Section 4.

1.3 Related work

Chen, Jin, and Williams [10] generalized hardness magnification phenomena to arbitrary sparse languages in NP. Note that MCSP$[2^{\mu n}]$ is a *sparse* language in the sense that the number of YES instances of MCSP$[2^{\mu n}]$ is at most $2^{\widetilde{O}(2^{\mu n})}$, which is much smaller than the number 2^{2^n} of all the instances of length 2^n. Hirahara [19] proved that a super-linear-size lower bound on co-non-deterministic branching programs for computing an approximation and space-bounded variant of MKtP implies the existence of a hitting set generator secure against read-once branching programs (and, in particular, $\mathsf{RL} = \mathsf{L}$).

Regarding unconditional lower bounds for MCSP, Razborov and Rudich [35] showed that there exists no AC^0-natural property useful against $\mathsf{AC}^0[\oplus]$, which in particular implies that MCSP $\notin \mathsf{AC}^0$; otherwise, the complement of MCSP would yield an AC^0-natural property useful against $\mathsf{P/poly} \supseteq \mathsf{AC}^0[\oplus]$. Hirahara and Santhanam [20] proved that MCSP essentially requires quadratic-size de Morgan formulas. Cheraghchi, Kabanets, Lu, and Myrisiotis [11] proved that MCSP essentially requires cubic-size de Morgan formulas as well as quadratic-size (general, unconstrained) branching programs. Golovnev, Ilango, Impagliazzo, Kabanets, Kolokolova, and Tal [14] proved that, for any prime p, MCSP requires constant-depth circuits, that are augmented with MOD_p gates, of weakly-exponential size.

The state-of-the-art time lower bound against DTMs on inputs of size n is $\Omega\big(n^2\big)$, proved by Maass [28], for the Polydromes function (which is a generalization of Palindromes). Regarding the case when the considered DTMs have a two-way read-only input tape, Maass and Schorr [29] proved that there is some problem in $\Sigma_2\mathsf{TIME}[n]$ that requires $\Omega\big(n^{3/2}/\log^6 n\big)$ time to compute on such machines. As mentioned earlier, in Section 1.1, the state-of-the-art time lower bound against RTMs is due to Kalyanasundaram and Schnitger [26], who showed that Element Distinctness is not in $\mathsf{BPTIME}_1\big[o\big(N^2/\log N\big)\big]$.

Viola [38] gave a PRG that fools RTMs that run in time $n^{1+\Omega(1)}$; this also yields a $n^{1+\Omega(1)}$ time lower bound against such machines. To do this, Viola extended prior work [29, 37] on simulating any RTM by a sum of ROBPs [see Lemma 20] and then employed the PRG by Haramaty, Lee, and Viola [15] that fools ROBPs;[4] it is a straightforward observation [38],

[4] It should be noted that before Haramaty, Lee, and Viola [15] and Viola [38], the problem of designing PRGs of polynomial stretch that fool RTMs was wide open despite intense research efforts.

then, that the Forbes-Kelley PRG [13] [which appeared afterwards and was inspired by the PRG by Haramaty, Lee, and Viola] yields a PRG of nearly quadratic stretch that fools RTMs and, therefore, a nearly quadratic lower bound against the same model as well. Moreover, Viola [38] showed that there exists some problem in $\Sigma_3\mathsf{TIME}[n]$ that requires $n^{1+\Omega(1)}$ time to compute on any RTM that has the extra feature of a two-way read-only input tape; one of the ingredients of this result, is again the PRG by Haramaty, Lee, and Viola [15].

For the case of one-tape TMs with no extra tapes, Hennie [17] proved in 1965 that the Palindromes function requires $\Omega(n^2)$ time to compute. Van Melkebeek and Raz [37] observed fixed-polynomial time lower bounds for SAT against non-deterministic TMs with a d-dimensional read/write two-way work tape and a random access read-only input tape; these lower bounds depend on d.

1.4 Organization

In Section 2, we give the necessary background. We prove Theorem 3 in Section 3, and Theorem 4 in Section 4. The proofs of the rest of our results appear in the full version.

2 Preliminaries

2.1 Circuit complexity

Let $f : \{0,1\}^n \to \{0,1\}$. We define the *circuit complexity of* f, denoted by $\mathrm{CC}(f)$, to be equal to the size (i.e., the number of gates) of the smallest bounded fan-in unbounded fan-out Boolean circuit, over the $\{\mathrm{AND}, \mathrm{OR}, \mathrm{NOT}\} = \{\wedge, \vee, \neg\}$ basis, that, on input x, outputs $f(x)$. For a string $y \in \{0,1\}^{2^n}$, we denote by $\mathrm{CC}(y)$ the circuit complexity of the function $f_y : \{0,1\}^n \to \{0,1\}$ encoded by y; i.e., $f_y(x) = y_x$, for any $x \in \{0,1\}^n$.

A standard counting argument shows that a random function attains nearly maximum circuit complexity with high probability.

▶ **Proposition 8** ([36]). *For any function $s\colon \mathbb{N} \to \mathbb{N}$ with $s(n) = o(2^n/n)$, it holds that*

$$\Pr_{x \sim \{0,1\}^{2^n}}[\mathrm{CC}(x) \le s(n)] = o(1),$$

for all large $n \in \mathbb{N}$.

▶ **Definition 9** (Minimum Circuit Size Problem [25]). *We define* MCSP *as*

$$\mathrm{MCSP} := \left\{(x, \theta) \in \{0,1\}^{2^n} \times \{0,1\}^n \mid \mathrm{CC}(x) \le \theta\right\}_{n \in \mathbb{N}},$$

and its parameterized version *as*

$$\mathrm{MCSP}[s(n)] := \left\{x \in \{0,1\}^{2^n} \mid \mathrm{CC}(x) \le s(n)\right\}_{n \in \mathbb{N}},$$

for a size parameter $s\colon \mathbb{N} \to \mathbb{N}$.

2.2 Turing machines

Throughout this paper, we consider a Turing machine that has one work tape and a one-way input tape. In this context, "one-way" means that the tape-head may move only from left to right.

A *deterministic Turing machine (DTM)* is a Turing machine with two tapes: A two-way read/write work tape and a one-way read-only input tape. Let $x \in \{0,1\}^*$ and M be a DTM; we write $M(x)$ to denote the output of M when its input tape is initialized with x and its work tape is empty. Let $t\colon \mathbb{N} \to \mathbb{N}$ be time-constructible. The class of languages $L \subseteq \{0,1\}^*$ decided by some $O(1)$-state time-t DTM is denoted by $\mathsf{DTIME}_1[t]$.

We also consider a randomized variant of DTMs. A *randomized Turing machine (RTM)* is a Turing machine with three tapes: A two-way read/write work tape, a one-way read-only input tape, and a one-way read-only random tape. Let $x, r \in \{0,1\}^*$ and M be a RTM; we write $M(x,r)$ to denote the output of M when its input tape contains x, its work tape is empty, and its random tape contains r. Let $t\colon \mathbb{N} \to \mathbb{N}$ be time-constructible. For a language $L \subseteq \{0,1\}^*$ and a RTM M, we say that M *decides* L *with two-sided error* if $\Pr_r[M(x,r)=1] \geq \frac{2}{3}$ for every input $x \in L$ and $\Pr_r[M(x,r)=0] \geq \frac{2}{3}$ for every input $x \notin L$. The class of languages $L \subseteq \{0,1\}^*$ decided by some $O(1)$-state time-t RTM with two-sided error is denoted by $\mathsf{BPTIME}_1[t]$.

A *randomized oracle Turing machine (oracle RTM)* is a Turing machine with four tapes: A two-way read/write work tape, a one-way read-only input tape, a one-way read-only random tape, and an oracle tape. This model is identical to the randomized Turing machine model apart from the oracle tape, which is a standard oracle tape. The class of languages $L \subseteq \{0,1\}^*$ decided by some $O(1)$-state time-t oracle RTM, with access to some oracle $O \subseteq \{0,1\}^*$, with two-sided error is denoted by $\mathsf{BPTIME}_1^O[t]$.

2.3 Streaming algorithms

A *space-$s(n)$ streaming algorithm* with update time $u(n)$ on an input $x \in \{0,1\}^n$ has a working storage of $s(n)$ bits. At any point the algorithm can either choose to perform one operation on $O(1)$ bits in storage or it can choose to read the next bit from the input. The total time between two next-bit reads is at most $u(n)$ and the final outcome is reported in $O(u(n))$ time.

▶ **Lemma 10.** *Any one-pass streaming algorithm with $t(N)$ update time, on inputs of length N, can be simulated by a one-tape Turing machine with a one-way read-only input tape running in time $O(N \cdot \operatorname{poly}(t(N)))$.*

Proof. Recall that a streaming algorithm reads one bit of its input from left to right, and each consecutive read operation occurs within $t(N)$ time steps. Thus, it takes $N \cdot \operatorname{poly}(t(N))$ time-steps in total to finish the computation on inputs of length N in the standard multi-tape Turing machine model, as the size of the input is N and $\operatorname{poly}(t(N))$ time-steps suffice for some multi-tape Turing machine to perform an update [12]. For any time constructible function $T : \mathbb{N} \to \mathbb{N}$, a one-tape Turing machine can simulate a $T(n)$-time multi-tape Turing machine within $O(T(n)^2)$ steps. Thus, a streaming algorithm can be simulated in time $N \cdot (\operatorname{poly}(t(N)))^2 = N \cdot \operatorname{poly}(t(N))$ by a one-tape Turing machine. ◀

2.4 Branching programs

A *branching program (BP)* is a directed acyclic graph with three special vertices: a start vertex s (the *source*) and two finish vertices, namely an accepting vertex h_1 and a rejecting vertex h_0 (the *sinks*).

On input $x \in \{0,1\}^n$, the computation starts at s and follows a directed path from s to some h_b, with $b \in \{0,1\}$. On this occasion, the output of the computation is b. In each step, the computation queries some input x_i, for $i \in [n]$, and then visits some other node, depending on the value of the variable just queried, namely 0 or 1, through an edge with label "$x_i = 0$" or "$x_i = 1$," respectively.

A branching program P *decides a language* $L \subseteq \{0,1\}^*$ in the natural way, i.e., $x \in L$ if and only if, on input x, the computation path that P follows starts at s and finishes at h_1. If the branching program is layered and the variable queried within each layer is the same, then the branching program is called *oblivious*. If the branching program queries each variable at most once, then the branching program is called a *read-once branching program (ROBP)*. If the branching program is oblivious and always queries the variables in some known order, where it is known beforehand which variable is queried at each layer, then the branching program is called *known-order*, else it is called *unknown-order*.

A branching program is called *non-deterministic* if some of its vertices have an arbitrary number of outgoing edges (i.e., if this number is not 2) or if some of its vertices have edges that do not refer to the same input variable. Non-deterministic branching programs may also have *unlabelled* edges, as well. Due to the nature of a non-deterministic branching program, it is possible that a computation never reaches either h_0 or h_1 as there can be some node with edges that their labels are all false according to the input at hand; in this case, we assume that the computation halts in a rejecting state.

A *non-deterministic branching program computes a function* $f\colon \{0,1\}^n \to \{0,1\}$ if, for every $x \in \{0,1\}^n$ such that $f(x) = 1$, there is some s-h_1 path and for every $x \in \{0,1\}^n$ such that $f(x) = 0$, all computations end in a rejecting state.

A *co-non-deterministic branching program computes a function* $f\colon \{0,1\}^n \to \{0,1\}$ if, for every $x \in \{0,1\}^n$ such that $f(x) = 1$, all source-to-sink paths are s-h_1 paths and for every $x \in \{0,1\}^n$ such that $f(x) = 0$, there exists some rejecting computation.

A *parity branching program* is a branching program that has counting semantics. That is, a parity branching program computes a function $f\colon \{0,1\}^n \to \{0,1\}$ if, for every $x \in \{0,1\}^n$ such that $f(x) = 1$, there is an odd number of s-h_1 paths and for every $x \in \{0,1\}^n$ such that $f(x) = 0$, there is an even number of s-h_1 paths.

We define the *size* of a branching program to be the number of its labelled edges.

2.5 Pseudorandom generators and hitting set generators

We recall the standard notions of pseudorandom generators and hitting set generators.

► **Definition 11.** *Let* $s\colon \mathbb{N} \to \mathbb{N}$ *be a function,* $\mathfrak{C}$ *be a circuit class, and* $0 < \varepsilon < 1$*. A* pseudorandom generator (PRG) that ε-fools $\mathfrak{C}$ *is a function* $G\colon \{0,1\}^{s(n)} \to \{0,1\}^n$ *such that*

$$\left| \mathop{\mathbf{Exp}}_{x \sim \{0,1\}^n}[f(x)] - \mathop{\mathbf{Exp}}_{y \sim \{0,1\}^{s(n)}}[f(G(y))] \right| \le \varepsilon,$$

for any circuit $C \in \mathfrak{C}$*. The value* $s(n)$ *is referred to as the* seed length *of* G*.*

► **Definition 12.** *Let* $s\colon \mathbb{N} \to \mathbb{N}$ *be a function,* $\mathfrak{C}$ *be a circuit class, and* $0 < \varepsilon < 1$*. A* hitting set generator (HSG) ε-secure against $\mathfrak{C}$ *is a function* $G\colon \{0,1\}^{s(n)} \to \{0,1\}^n$ *such that*

$$\mathop{\mathbf{Pr}}_{x \sim \{0,1\}^n}[C(x) = 1] \ge \varepsilon \quad \Longrightarrow \quad C(H(y)) = 1 \text{ for some } y \in \{0,1\}^{s(n)},$$

for any circuit $C \in \mathfrak{C}$*. By default, we choose* $\varepsilon := 1/2$*.*

For our purpose, it is useful to extend the notion of PRG to a pseudorandom generator that fools *randomized* algorithms.

▶ **Definition 13.** *For a function* $s\colon \mathbb{N} \to \mathbb{N}$ *and a parameter* $0 < \varepsilon < 1$*, a function* $G\colon \{0,1\}^{s(n)} \to \{0,1\}^n$ *is said to be a pseudorandom generator that* ε-fools q-state time-t RTMs *if*

$$\left| \mathop{\mathbf{Exp}}_{\substack{x\sim\{0,1\}^n,\\ r\sim\{0,1\}^t}}[M(x,r)] - \mathop{\mathbf{Exp}}_{\substack{y\sim\{0,1\}^{s(n)},\\ r\sim\{0,1\}^t}}[M(G(y),r)] \right| \le \varepsilon,$$

for any q*-state time-*t *RTM* M*.*

2.6 MCSP lower bounds from local HSGs

For a function $G\colon \{0,1\}^s \to \{0,1\}^n$, we say that G is *local* [11] if $\mathrm{CC}(G(z)) \le s$ for every string $z \in \{0,1\}^s$. We make use of the following standard fact.

▶ **Lemma 14.** *Let* $s\colon \mathbb{N} \to \mathbb{N}$ *be a function such that* $s(n) = o(2^n/n)$*, and* $N := 2^n$*. Suppose that there exists a local hitting set generator* $H\colon \{0,1\}^{s(n)} \to \{0,1\}^N$ *for a circuit class* $\mathfrak{C}$*. Then,* $\mathrm{MCSP}[s(n)] \notin \mathsf{co}\mathfrak{C}$*.*

Proof. We prove the contrapositive. Let $C \in \mathsf{co}\mathfrak{C}$ be a circuit that computes $\mathrm{MCSP}[s(n)]$. Since $\mathrm{CC}(H(z)) \le s(n)$, we have $H(z) \in \mathrm{MCSP}[s(n)]$; thus $C(H(z)) = 1$, for every $z \in \{0,1\}^{s(n)}$. For a random $w \sim \{0,1\}^N$, it follows from Proposition 8 that $w \notin \mathrm{MCSP}[s(n)]$ with probability $1 - o(1)$; hence $C(w) = 0$ for most w. Therefore, $\neg C \in \mathfrak{C}$ accepts at least a half of $\{0,1\}^N$ but rejects every string in the range of H, which contradicts the security of the hitting set generator H. ◀

We observe that a local pseudorandom generator for time-t RTMs also "fools" $\mathsf{BPTIME}_1[t(N)]$ in the following sense.

▶ **Lemma 15.** *Let* $s, t\colon \mathbb{N} \to \mathbb{N}$ *be functions, such that* $s(n) = o(2^n/n)$*, and* $N := 2^n$*. Suppose that there is a family of local pseudorandom generators* $G = \{G_n : \{0,1\}^{s(n)} \to \{0,1\}^N\}_{n\in\mathbb{N}}$ *such that, for every* $n \in \mathbb{N}$*,* G_n $(1/6)$*-fools time-*$t(N)$ *RTMs. Then,* $\mathrm{MCSP}[s(n)]$ *is not in* $\mathsf{BPTIME}_1[t(N)]$*.*

Proof. We prove the contrapositive. Let M be a time-t RTM that decides $\mathrm{MCSP}[s(n)]$. Fix any $n \in \mathbb{N}$. For any seed $z \in \{0,1\}^{s(n)}$, we have $G_n(z) \in \mathrm{MCSP}[s(n)]$ since G_n is local. Thus, $\Pr_r[M(G_n(z), r) = 1] \ge 2/3$. On the other hand, pick a string $w \in \{0,1\}^N$ chosen uniformly at random. By the counting argument of Proposition 8, we get $\Pr_w[w \notin \mathrm{MCSP}[s(n)]] \ge 1 - o(1)$. Thus, we have $\Pr_{w,r}[M(w,r) = 1] \le o(1) + 1/3 < 1/2$. Therefore,

$$\Pr_{z,r}[M(G_n(z),r) = 1] - \Pr_{w,r}[M(w,r) = 1] > \frac{2}{3} - \frac{1}{2} = \frac{1}{6},$$

which means that G_n does not fool RTMs. ◀

3 MCSP lower bounds against one-tape oracle RTMs

In this section, we present a proof of our main result.

▶ **Theorem 16** (Theorem 3, restated)**.** *Let* $O \subseteq \{0,1\}^*$ *be any language. Then, for every constant* $1/2 < \mu < 1$*,* $\mathrm{MCSP}[2^{\mu\cdot n}]$ *on truth tables of size* $N := 2^n$ *is not in* $\mathsf{BPTIME}_1^O\left[N^{2\cdot(\mu' - o(1))}\right]$ *for all* $1/2 < \mu' < \mu$*, where all of the strings queried to* O *are of length* $N^{o(1)}$*.*

3.1 Connections to hardness magnification

As discussed in Section 1.1.1, Theorem 16 implies that establishing hardness magnification phenomena for MCSP, when the circuit size threshold parameter is $2^{0.51n}$, would require the development of new techniques; see Remark 19. To explain why this is true, we shall first require the following result by McKay, Murray, and Williams [30] that gives an oracle streaming algorithm for MCSP.

▶ **Lemma 17** ([30, Theorem 1.2]). *Let $s : \mathbb{N} \to \mathbb{N}$ be a size function, with $s(n) \geq n$ for all n, and $N := 2^n$. Then, there is a one-pass streaming algorithm for $\mathrm{MCSP}[s(n)]$ on N-bit inputs running in $N \cdot \widetilde{O}(s(n))$ time with $\widetilde{O}\Big(s(n)^2\Big)$ update time and $\widetilde{O}(s(n))$ space, using an oracle for $\Sigma_3\mathrm{SAT}$ with queries of length $\widetilde{O}(s(n))$.*

A corollary of Lemma 17 and Lemma 10 is the following.

▶ **Corollary 18** (Consequences of hardness magnification from currently known techniques). *Let $s : \mathbb{N} \to \mathbb{N}$ be a size function. Then, $\mathrm{MCSP}[s(n)]$ on truth tables of length $N := 2^n$ is in $\mathsf{DTIME}_1^O[N \cdot \mathrm{poly}(s(n))]$, for some $O \in \Sigma_3^{\mathsf{P}}$, where all of the strings queried to O are of length at most $\mathrm{poly}(s(n))$.*

The following remark summarizes the main idea of this subsection.

▶ Remark 19. By Corollary 18, we see that if $s(n) = 2^{\mu \cdot n}$, for $\mu = o(1)$, then $\mathrm{MCSP}[s(n)]$ is in $\mathsf{DTIME}_1^O\left[N^{1+o(1)}\right]$, where all of the strings queried to O are of length $N^{o(1)}$. In light of this observation, Theorem 16 is important for the following reason. As $\mathsf{DTIME}_1^O\left[N^{1+o(1)}\right]$ is a subset of $\mathsf{BPTIME}_1^O\left[N^{2\cdot\left(\mu'-o(1)\right)}\right]$ for all $1/2 < \mu' < 1$ and all languages $O \subseteq \{0,1\}^*$, Theorem 16 shows that establishing hardness magnification phenomena for $\mathrm{MCSP}[s(n)]$ like that of Theorem 1, when $s(n) = 2^{\mu \cdot n}$ for any constant $1/2 < \mu < 1$, would require the development of techniques that do not rely on designing oracle algorithms that make short oracle queries.

3.1.1 Comparison with the locality barrier

Chen, Hirahara, Oliveira, Pich, Rajgopal, and Santhanam [9] introduced the "locality barrier" to explain why it will be difficult to acquire a major complexity breakthrough through the lens of hardness magnification. Their reasoning goes as follows:

> Existing magnification theorems unconditionally show that problems, against which some circuit lower bound implies a complexity-theoretic breakthrough, admit highly efficient small fan-in oracle circuits, while lower bound techniques against weak circuit models quite often easily extend to circuits containing such oracles.

Our Remark 19, therefore, is close in spirit to the results of Chen et al. [9]: We make use of a lower bound (Theorem 16) to motivate the development of new techniques for proving hardness magnification phenomena while Chen et al. make use of hardness magnification phenomena to motivate the development of new techniques for acquiring lower bounds; a notable difference is that we consider one-tape Turing machines while they consider Boolean circuits.

3.2 Proof of Theorem 16

In order to prove Theorem 16, our goal is to construct a local pseudorandom generator that fools oracle RTMs and then apply Lemma 15. Viola [38] constructed a pseudorandom generator that fools the one-tape Turing machine model (DTM).[5] We will show that, in fact, the same construction fools oracle RTMs as well. In order to do so, we recall the idea of Viola [38]. The idea is that, in order to fool DTMs, it is sufficient to use a PRG that ε-fools ROBPs for an exponentially small ε. This is because time-t DTMs can be written as the sum of an exponential number of ROBPs.

▶ **Lemma 20** (Viola [38]). *Let $n \in \mathbb{N}$ and M be a q-state time-t DTM. Then, there is a family $\{P_\alpha\}_{\alpha \in A}$ of n-input ROBPs of width $\exp\big(O\big(\sqrt{t} \cdot \log(tq)\big)\big)$ such that, for any $x \in \{0,1\}^n$,*

$$M(x) = \sum_{\alpha \in A} P_\alpha(x),$$

where $|A| \leq (tq)^{O(\sqrt{t})}$.

By a simple calculation, any pseudorandom generator that $\varepsilon/|A|$-fools ROBPs also ε-fools DTMs. Viola [38] then used the pseudorandom generator of Forbes and Kelley [13] that fools ROBPs. By a careful examination, we will show that the Forbes-Kelley pseudorandom generator is local; see the full version.

▶ **Theorem 21** (Forbes-Kelley PRG is local). *There exists a local pseudorandom generator with seed length $\widetilde{O}\big((\sqrt{t} + \log(1/\varepsilon)) \cdot \log q\big)$ that ε-fools q-state time-t n-input DTMs for any $t \geq n$.*

Our main idea for obtaining an oracle randomized Turing machine lower bound is that Viola's reduction can be applied to *non-uniform computational models*, i.e., q-state Turing machines where q can become large as the input length becomes large. More specifically, it is possible to incorporate all possible oracle queries [along with their answers] and any good coin flip sequence r into the internal states of DTMs.

▶ **Lemma 22.** *For an input length $n \in \mathbb{N}$, for any q-state time-t oracle RTM M, that only queries strings of length at most ℓ to its oracle O, and a coin flip sequence $r \in \{0,1\}^t$, there exists some $\big(q \cdot 2^\ell \cdot t\big)$-state time-$t$ DTM M' such that $M'(x) = M^O(x,r)$ for every input $x \in \{0,1\}^n$.*

Proof. Let Q_M denote the set of the states of M. We define the set of the states of M' as

$$Q_{M'} := \Big\{(q,s,b,i) \in Q_M \times \{0,1\}^\ell \times \{0,1\} \times [t] \mid O(s) = b\Big\}.$$

The transition from the state $(q,s,b,i) \in Q_{M'}$ can be defined in a natural way, by using the i-th bit of r, namely r_i, the state q, and the fact that $O(s) = b$. ◀

▶ **Corollary 23.** *There exists a local pseudorandom generator with seed length $\sigma(t,q,\varepsilon) = \widetilde{O}\big((\sqrt{t} + \log(1/\varepsilon)) \cdot \log(q \cdot 2^\ell \cdot t)\big)$ that ε-fools q-state time-t n-input oracle RTMs that may only query strings of length at most ℓ to their oracle, for any $t \geq n$.*

[5] We note that our definition of PRG is different from that of [38] in that a random tape is not regarded as an input tape.

Proof. We hard-code the oracle queries and their answers in the internal states and, moreover, we use an averaging argument to fix one good coin flip sequence r. Let M be any q-state time-t oracle RTM that may query to its oracle O strings of length at most ℓ. Let G be a PRG from Theorem 21. We have that

$$\begin{aligned}
&\left|\mathop{\mathbf{Exp}}_{r\sim\{0,1\}^t}\left[\mathop{\mathbf{Exp}}_{x\sim\{0,1\}^n}\left[M^O(x,r)\right]\right] - \mathop{\mathbf{Exp}}_{r\sim\{0,1\}^t}\left[\mathop{\mathbf{Exp}}_{y\sim\{0,1\}^{\sigma(t,q,\varepsilon)}}\left[M^O(G(y),r)\right]\right]\right| \\
&\quad = \left|\mathop{\mathbf{Exp}}_{r}\left[\mathop{\mathbf{Exp}}_{x}\left[M^O(x,r)\right] - \mathop{\mathbf{Exp}}_{y}\left[M^O(G(y),r)\right]\right]\right| \\
&\quad \le \mathop{\mathbf{Exp}}_{r}\left[\left|\mathop{\mathbf{Exp}}_{x}\left[M^O(x,r)\right] - \mathop{\mathbf{Exp}}_{y}\left[M^O(G(y),r)\right]\right|\right] \\
&\quad \le \left|\mathop{\mathbf{Exp}}_{x}\left[M^O(x,r^*)\right] - \mathop{\mathbf{Exp}}_{y}\left[M^O(G(y),r^*)\right]\right|,
\end{aligned}$$

for some $r^* \in \{0,1\}^t$, by an averaging argument. By applying Lemma 22, for M^O, O, and r^*, we obtain an equivalent $\left(q \cdot 2^\ell \cdot t\right)$-state time-$t$ DTM M'. The result now follows from Theorem 21. Specifically,

$$\left|\mathop{\mathbf{Exp}}_{x}\left[M^O(x,r^*)\right] - \mathop{\mathbf{Exp}}_{y}\left[M^O(G(y),r^*)\right]\right| = \left|\mathop{\mathbf{Exp}}_{x}[M'(x)] - \mathop{\mathbf{Exp}}_{y}[M'(G(y))]\right| \le \varepsilon. \qquad \blacktriangleleft$$

Proof of Theorem 16. Take the local pseudorandom generator G of Corollary 23 with parameter $\varepsilon := 1/6$. Let $1/2 < \mu' < \mu < 1$ be arbitrary constants. Let $t, s, \ell \colon \mathbb{N} \to \mathbb{N}$ be functions such that $t(N) = N^{2\cdot\left(\mu'-o(1)\right)}$, $s(n) = 2^{\mu\cdot n}$, and $\ell(n) = 2^{o(n)}$. Then, the seed length of G is at most

$$\widetilde{O}\left(\sqrt{t(N)} \cdot (\log q + \ell(n))\right) \le \widetilde{O}(N^{\mu'-o(1)+o(1)}) \le s(n),$$

where $N = 2^n$. Since $s(n) = o(2^n/n)$, by Lemma 15, we obtain that $\mathrm{MCSP}[s(n)] \notin \mathsf{BPTIME}_1^O[t(N)]$, where all of the strings queried to O are of length $N^{o(1)}$. ◀

4 MKTP lower bounds against branching programs

In this section, we develop a proof technique for applying Nečiporuk's method to MKTP and prove Theorem 4. The KT-complexity is formally defined as follows.

▶ **Definition 24.** *Let U be an efficient universal Turing machine. For a string $x \in \{0,1\}^*$, the* KT-complexity *of x is defined as follows.*

$$\mathrm{KT}(x) := \min\{|d| + t \mid U^d(i) \textit{ outputs } x_i \textit{ in time } t \textit{ for every } i \in [|x|+1]\}.$$

Here we define x_i as the ith bit of x if $i \le |x|$ and $\bot$ otherwise.

For a threshold $\theta \colon \mathbb{N} \to \mathbb{N}$, we denote by $\mathrm{MKTP}[\theta]$ the problem of deciding whether $\mathrm{KT}(x) \le \theta(|x|)$ given a string $x \in \{0,1\}^*$ as input.

Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function and $\rho \in \{0,1,*\}^n$ a restriction. The *ρ-restricted version of f* is a function, denoted by $f\restriction_\rho$, such that for any $x \in \{0,1\}^n$ it is the case that $f\restriction_\rho(x) := f(y)$ where $y \in \{0,1\}^n$ and, for all $1 \le i \le n$, $y_i := \rho(i)$ if $\rho(i) \in \{0,1\}$, else $y_i := x_i$.

For a function $f\colon \{0,1\}^n \to \{0,1\}$, we partition the input variables $[n]$ into disjoint blocks $V_1, \cdots, V_m$, where $|V_i| = v$ for each $i \in [m]$ and $n = vm$. ($v = O(\log n)$ will be chosen later.) The idea of the Nečiporuk's method is to lower-bound the number of subfunctions. For each $i \in [m]$, we define $c_i(f)$ to be the number of distinct functions $f\restriction_\rho$ such that $\rho\colon [n] \to \{0,1,*\}$ is a restriction with $\rho^{-1}(*) = V_i$.

The Nečiporuk method can be then summarized as follows.

▶ **Theorem 25** (Nečiporuk [32]; cf. [24, Theorem 15.1])**.** *The size of a branching program computing f is at least $\Omega\left(\sum_{i=1}^m \log c_i(f)/\log\log c_i(f)\right)$. The size of a non-deterministic branching program or a parity branching program computing f is at least $\Omega\left(\sum_{i=1}^m \sqrt{\log c_i(f)}\right)$.*

Our main technical result of this section is the following.

▶ **Theorem 26.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be $\mathrm{MKTP}[\theta]$ on n-bit inputs for $\theta := n - 3c\log n - 4$, where $c > 0$ is a universal constant. Then, for every $i \in [m]$, it holds that $c_i(f) = 2^{\Omega(n)}$.*

The lower bounds for branching programs (Theorem 4) immediately follow from Theorem 26 and Theorem 25.

In our proof of Theorem 26, we only need the following two properties of KT-complexity.

1. The resource-unbounded Kolmogorov complexity[6] provides a lower bound on the KT-complexity. That is, $\mathrm{K}(x) \le \mathrm{KT}(x)$ for any $x \in \{0,1\}^*$.
2. For any strings $\rho_1, \cdots, \rho_m \in \{0,1\}^v$ such that there exist distinct indices $i \ne j \in [m]$ such that $\rho_i = \rho_j$, we have $\mathrm{KT}(\rho_1 \cdots \rho_m) \le (m-1)\cdot v + O(\log n)$. This is because each bit of the string $\rho_1 \cdots \rho_m$ can be described by the strings $\{\rho_1, \cdots, \rho_m\} \setminus \{\rho_j\}$ and the index $j \in [m]$ in time $O(\log n)$.[7]

For simplicity, we focus on the case when $i = 1$; the other case can be proved similarly. The idea of the proof is the following. Imagine that we pick $\rho \in \{*\}^{V_1} \times \{0,1\}^{V_2 \cup \cdots \cup V_m}$ uniformly at random. (Here we identify a restriction with a string in $\{0,1,*\}^{[n]}$.) We denote by $\rho_2 \in \{0,1\}^{V_2}, \cdots, \rho_m \in \{0,1\}^{V_m}$ the random bits such that $\rho = *^{V_1}\rho_2 \cdots \rho_m$. We will sometimes identify $\rho_2 \cdots \rho_m$ with ρ.

Consider the function $f\restriction_\rho\colon \{0,1\}^{V_1} \to \{0,1\}$ obtained by restricting f by ρ. Then, we expect that $f\restriction_\rho(\rho_i) = 1$ for any $i \in \{2, \cdots, m\}$ since $\mathrm{KT}(\rho_i\rho_2 \cdots \rho_m)$ is small, whereas $f\restriction_\rho(U) = 0$ for a random $U \sim \{0,1\}^{V_1}$ with high probability. Thus, the function $f\restriction_\rho$ is likely to be distinct for a randomly chosen ρ.

In order to make the argument formal, we proceed as follows. Pick ρ randomly. Then we add it to a set P while keeping the promise that the map $\rho \in P \mapsto f\restriction_\rho$ is injective. We will show that one can keep adding ρ until the size of P becomes exponentially large.

We will make use of symmetry of information of (resource-unbounded) Kolmogorov complexity.

▶ **Lemma 27.** *There exists a constant $c > 0$ such that, for any strings $x, y \in \{0,1\}^*$,*

$$\mathrm{K}(xy) \ge \mathrm{K}(x) + \mathrm{K}(y \mid x) - c\log \mathrm{K}(xy)\,.$$

---

[6] Let U be an efficient universal Turing machine. For a string $x \in \{0,1\}^*$, the *resource-unbounded Kolmogorov complexity of x* is defined as $\mathrm{K}(x) := \min\{|d| \mid U^d(i) \text{ outputs } x_i \text{ for every } i \in [|x|+1]\}$.

[7] Here we assume that the universal Turing machine is efficient. If the universal Turing machine is slower and the time is $\mathrm{polylog}(n)$, we obtain a branching program size lower bound of $n^2/\mathrm{polylog}(n)$.

We focus on restrictions ρ such that ρ is Kolmogorov-random. To this end, define

$$R := \{\rho \in \{0,1\}^{V_2 \cup \cdots \cup V_m} \mid \mathrm{K}(\rho) \geq |\rho| - 1\}$$

as the set of Kolmogorov-random restrictions ρ. By the standard counting argument, we have

$$\Pr_\rho[\rho \notin R] \leq \sum_{i=1}^{|\rho|-2} 2^i/2^{|\rho|} \leq \frac{1}{2}.$$

The following lemma is the key for counting the number of distinct subfunctions.

▶ **Lemma 28.** *Let $\rho' \in R$ be an arbitrary restriction and define $\theta := n - v + c\log n$. If $f\restriction_\rho \equiv f\restriction_{\rho'}$, then $\mathrm{K}(\rho_i \mid \rho') \leq 2c\log n + 1$ for any $i \in \{2, \cdots, m\}$.*

Proof. For each $i \in [m] \setminus \{1\}$,

$$\mathrm{KT}(\rho_i\rho_2 \cdots \rho_m) \leq |\rho_2| + \cdots + |\rho_m| + O(\log n) \leq (m-1) \cdot v + c\log n \leq \theta.$$

This means that $\rho_i\rho_2 \cdots \rho_m$ is a YES instance of MKTP[θ]. Therefore, we have $1 = f\restriction_\rho(\rho_i) = f\restriction_{\rho'}(\rho_i)$, which implies that $\mathrm{KT}(\rho_i\rho'_2 \cdots \rho'_m) \leq \theta$. By the symmetry of information,

$$\theta \geq \mathrm{KT}(\rho_i\rho'_2 \cdots \rho'_m) \geq \mathrm{K}(\rho_i\rho'_2 \cdots \rho'_m) \geq \mathrm{K}(\rho'_2 \cdots \rho'_m) + \mathrm{K}(\rho_i \mid \rho'_2 \cdots \rho'_m) - c\log n.$$

Since $\rho' \in R$, we have $\mathrm{K}(\rho'_2 \cdots \rho'_m) \geq v(m-1) - 1 = n - v - 1$. Therefore,

$$\mathrm{K}(\rho_i \mid \rho'_2 \cdots \rho'_m) \leq \theta + c\log n - (n - v - 1) = 2c\log n + 1.$$ ◀

Now we set $v := 4c\log n + 4$. Then, for any $\rho' \in R$,

$$\begin{aligned}\Pr_\rho[f\restriction_\rho \equiv f\restriction_{\rho'}] &\leq \Pr[\forall i \in [m] \setminus \{1\},\ \mathrm{K}(\rho_i \mid \rho') \leq v/2 - 1] \\ &\leq (2^{v/2}/2^v)^{m-1} \\ &= 2^{-n/2+v/2} \\ &\leq 2^{-n/3}.\end{aligned}$$

In particular, for any $P \subseteq R$, by the union bound, we obtain

$$\Pr_\rho[\exists \rho' \in P,\ f\restriction_\rho \equiv f\restriction_{\rho'}] \leq |P| \cdot 2^{-n/3}.$$

Therefore,

$$\Pr_\rho[\rho \notin R \text{ or } \exists \rho' \in P,\ f\restriction_\rho \equiv f\restriction_{\rho'}] \leq 1/2 + |P| \cdot 2^{-n/3},$$

which is strictly less than 1 if $|P| < 2^{n/3-1}$. To summarize, we established the following property.

▶ **Corollary 29.** *For any $P \subseteq R$ such that $|P| < 2^{n/3-1}$, there exists a restriction ρ such that $\rho \in R$ and $f\restriction_\rho \not\equiv f\restriction_{\rho'}$ for any $\rho' \in P$.*

In light of this, we can construct a large set P such that the map $\rho \in P \mapsto f\restriction_\rho$ is injective as follows: Starting from $P := \emptyset$, add a restriction $\rho \in R$ such that $f\restriction_\rho \not\equiv f\restriction_{\rho'}$ for any $\rho' \in P$, whose existence is guaranteed by Corollary 29 if $|P| < 2^{n/3-1}$. In this way, we obtain a set P such that $|P| \geq 2^{n/3-1}$ and each $f\restriction_\rho$ is distinct for any $\rho \in P$. We conclude that $c_1(f) \geq |P| \geq 2^{n/3-1}$. This completes the proof of Theorem 26.

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 Shuichi Hirahara. New insights on the (non-)hardness of circuit minimization and related problems. In *Proceedings of the 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 54:1–54:14, 2017.

3 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010.

4 Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *J. Comput. Syst. Sci.*, 77(1):14–40, 2011.

5 Alexander E. Andreev, Juri L. Baskakov, Andrea E. F. Clementi, and José D. P. Rolim. Small pseudo-random sets yield hard functions: New tight explicit lower bounds for branching programs. In *Proceedings of the 26th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 179–189, 1999.

6 Paul Beame, Nathan Grosshans, Pierre McKenzie, and Luc Segoufin. Nondeterminism and an abstract formulation of Nečiporuk's lower bound method. *ACM Trans. Comput. Theory*, 9(1):5:1–5:34, 2016.

7 Allan Borodin, Alexander A. Razborov, and Roman Smolensky. On lower bounds for read-k-times branching programs. *Computational Complexity*, 3:1–18, 1993.

8 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Proceedings of the 31st Conference on Computational Complexity (CCC)*, pages 10:1–10:24, 2016.

9 Lijie Chen, Shuichi Hirahara, Igor Carboni Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. Beyond natural proofs: Hardness magnification and locality. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, pages 70:1–70:48, 2020.

10 Lijie Chen, Ce Jin, and R. Ryan Williams. Hardness magnification for all sparse NP languages. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 1240–1255, 2019.

11 Mahdi Cheraghchi, Valentine Kabanets, Zhenjian Lu, and Dimitrios Myrisiotis. Circuit lower bounds for MCSP from local pseudorandom generators. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 39:1–39:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

12 Stephen A. Cook and Robert A. Reckhow. Time-bounded random access machines. In Patrick C. Fischer, H. Paul Zeiger, Jeffrey D. Ullman, and Arnold L. Rosenberg, editors, *Proceedings of the 4th Annual ACM Symposium on Theory of Computing, May 1-3, 1972, Denver, Colorado, USA*, pages 73–80. ACM, 1972.

13 Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *Proceedings of the 59th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 946–955, 2018.

14 Alexander Golovnev, Rahul Ilango, Russell Impagliazzo, Valentine Kabanets, Antonina Kolokolova, and Avishay Tal. $\mathsf{AC}^0[p]$ lower bounds against MCSP via the coin problem. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 66:1–66:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

15 Elad Haramaty, Chin Ho Lee, and Emanuele Viola. Bounded independence plus noise fools products. *SIAM J. Comput.*, 47(2):493–523, 2018.

16 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

17 F. C. Hennie. One-tape, off-line turing machine computations. *Inf. Control.*, 8(6):553–578, 1965.

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

19 Shuichi Hirahara. Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions, 2020. Manuscript.

20 Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In *Proceedings of the 32nd Computational Complexity Conference (CCC)*, pages 7:1–7:20, 2017.

21 Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *Proceedings of the 31st Conference on Computational Complexity (CCC)*, pages 18:1–18:20, 2016.

22 John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *Proceedings of the 35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 236–245, 2015.

23 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from Shrinkage. *J. ACM*, 66(2):11:1–11:16, 2019.

24 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.

25 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.

26 Bala Kalyanasundaram and Georg Schnitger. Communication complexity and lower bounds for sequential computation. In *Informatik, Festschrift zum 60. Geburtstag von Günter Hotz*, pages 253–268. Teubner / Springer, 1992.

27 Chin Ho Lee. Fourier bounds and pseudorandom generators for product tests. In *Proceedings of the 34th Computational Complexity Conference (CCC)*, pages 7:1–7:25, 2019.

28 Wolfgang Maass. Quadratic lower bounds for deterministic and nondeterministic one-tape Turing machines (extended abstract). In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing (STOC)*, pages 401–408, 1984.

29 Wolfgang Maass and Amir Schorr. Speed-up of Turing machines with one work tape and a two-way input tape. *SIAM J. Comput.*, 16(1):195–202, 1987.

30 Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. Weak lower bounds on resource-bounded compression imply strong separations of complexity classes. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 1215–1225, 2019.

31 Cody D. Murray and Richard Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *Proceedings of the 30th Conference on Computational Complexity (CCC)*, pages 365–380, 2015.

32 E.I. Nečiporuk. On a Boolean function. *Doklady Akademii Nauk SSSR*, 169(4):765–766, 1966. English translation in Soviet Mathematics Doklady.

33 Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds. In *Proceedings of the 34th Computational Complexity Conference (CCC)*, pages 27:1–27:29, 2019.

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

35 Alexander A. Razborov and Steven Rudich. Natural proofs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, pages 204–213, 1994.

36 Claude E. Shannon. The synthesis of two-terminal switching circuits. *Bell Systems Technical Journal*, 28:59–98, 1949.

37 Dieter van Melkebeek and Ran Raz. A time lower bound for satisfiability. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings*, volume 3142 of *Lecture Notes in Computer Science*, pages 971–982. Springer, 2004.

38 Emanuele Viola. Pseudorandom bits and lower bounds for randomized Turing machines. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:51, 2019.

39 Osamu Watanabe. The time-precision tradeoff problem on on-line probabilistic Turing machines. *Theor. Comput. Sci.*, 24:105–117, 1983.