Electronic Colloquium on Computational Complexity, Report No. 139 (2018)

Hardness Magnification for Natural Problems

Igor C. Oliveira
University of Oxford

Rahul Santhanam
University of Oxford

August 10, 2018

Abstract

We show that for several natural problems of interest, complexity lower bounds that are barely non-trivial imply super-polynomial or even exponential lower bounds in strong computational models. We term this phenomenon "hardness magnification". Our examples of hardness magnification include:

1. Let $\mathsf{MCSP}[s]$ be the decision problem whose YES instances are truth tables of functions with circuit complexity at most $s(n)$. We show that if $\mathsf{MCSP}[2^{\sqrt{n}}]$ cannot be solved on average with zero error by formulas of linear (or even sub-linear) size, then NP does not have polynomial-size formulas. In contrast, Hirahara and Santhanam [HS17] recently showed that $\mathsf{MCSP}[2^{\sqrt{n}}]$ cannot be solved in the worst case by formulas of nearly quadratic size.
2. If there is a $c > 0$ such that for each positive integer d there is an $\varepsilon > 0$ such that the problem of checking if an n-vertex graph in the adjacency matrix representation has a vertex cover of size $(\log n)^c$ cannot be solved by depth-d AC^0 circuits of size $m^{1+\varepsilon}$, where $m = \Theta(n^2)$, then NP does not have polynomial-size formulas.
3. Let (α, β)-$\mathsf{MCSP}[s]$ be the promise problem whose YES instances are truth tables of functions that are α-approximable by a circuit of size $s(n)$, and whose NO instances are truth tables of functions that are not β-approximable by a circuit of size $s(n)$. We show that for arbitrary $1/2 < \beta < \alpha \leq 1$, if (α, β)-$\mathsf{MCSP}[2^{\sqrt{n}}]$ cannot be solved by randomized algorithms with random access to the input running in sublinear time, then $\mathsf{NP} \not\subseteq \mathsf{BPP}$.
4. If for each probabilistic quasi-linear time machine M using poly-logarithmic many random bits that is claimed to solve Satisfiability, there is a deterministic polynomial-time machine that on infinitely many input lengths n either identifies a satisfiable instance of bitlength n on which M does not accept with high probability or an unsatisfiable instance of bitlength n on which M does not reject with high probability, then $\mathsf{NEXP} \neq \mathsf{BPP}$.
5. Given functions $s, c \colon \mathbb{N} \to \mathbb{N}$ where $s \geq c$, let $\mathsf{MKtP}[c, s]$ be the promise problem whose YES instances are strings of Kt complexity [Lev84] at most $c(N)$ and NO instances are strings of Kt complexity greater than $s(N)$. We show that if there is a $\delta > 0$ such that for each $\varepsilon > 0$, $\mathsf{MKtP}[N^{\varepsilon}, N^{\varepsilon} + 5\log(N)]$ requires Boolean circuits of size $N^{1+\delta}$, then $\mathsf{EXP} \not\subseteq \mathsf{SIZE}(\mathrm{poly})$.

For each of the cases of magnification above, we observe that standard hardness assumptions imply much stronger lower bounds for these problems than we require for magnification.

We further explore magnification as an avenue to proving strong lower bounds, and argue that magnification circumvents the "natural proofs" barrier of Razborov and Rudich [RR97]. Examining some standard proof techniques, we find that they fall just short of proving lower bounds via magnification. As one of our main open problems, we ask whether there are other meta-mathematical barriers to proving lower bounds that rule out approaches combining magnification with known techniques.

ISSN 1433-8092

Contents

1 Introduction

1.1 Overview

It is a truth universally acknowledged that the state of the art in complexity lower bounds is very primitive indeed. The main open problems in complexity theory, such as NP vs. P and EXP vs. SIZE(poly) ask about *super-polynomial* lower bounds against *strong* computational models. In contrast, the lower bounds we can actually show for explicit problems are either for weak models such as constant-depth circuits [Ajt83, FSS84] and monotone circuits [Raz85], or are weak in magnitude, such as the $5n$ circuit size lower bound [IM02] or sub-cubic formula size lower bound for Andreev's function [Hås98]. Despite decades of effort, the frontier of complexity lower bounds tends to advance very slowly or not at all. Advances often require substantially new ideas, such as the algorithmic paradigm introduced by Williams [Wil13] and used to show [Wil14] that NEXP does not have polynomial-size constant-depth circuits with composite modular gates.

Is the lack of progress on lower bounds due to our own lack of imagination, or is there an inherent difficulty? Some evidence for the inherent difficulty of the problem is given by meta-mathematical barriers such as the relativization barrier [BGS75], natural proofs barrier [RR97] and algebrization barrier [AW09]. These barriers seek to understand and explain the limits of various classes of techniques for proving lower bounds.

The natural proofs barrier, in particular, suggests a certain dichotomy between "weak" circuit classes such as AC^0 circuits of sub-exponential size, $\mathsf{AC}^0[p]$ (for prime p) circuits of sub-exponential size, branching programs of sub-quadratic size and Boolean formulas of sub-cubic size on the one hand,[1] and "strong" circuit classes such as Boolean circuits of polynomial size or Boolean formulas of polynomial size which are believed to be able to implement pseudo-random functions (see e.g. [BR17]). For weak circuit classes, we already have lower bounds via current techniques using "naturalizing" proofs. For strong circuit classes, which are the ones that primarily interest us, lower bounds are believed to be far out of our reach, as there are not even good candidate techniques for proving such lower bounds. There are certain liminal classes such as ACC^0 and TC^0 circuits of depth 2 which don't yet fit clearly into this dichotomy, but by and large the dichotomy seems to capture current beliefs about the hardness of proving lower bounds.

We argue that this gap between weak and strong classes is somewhat illusory, or at the very least depends on the specific problems for which we try to show lower bounds. For various natural problems of interest, and for a range of computational models, we show that lower bounds against weak circuit classes *imply* lower bounds for strong circuit classes. We term this phenomenon "hardness magnification".

Hardness magnification in *computational complexity* is related to the somewhat recent work of Allender and Koucký [AK10], who showed how to amplify lower bounds for NC^1 against constant-depth circuit classes from size $n^{1+\varepsilon}$ for fixed $\varepsilon > 0$ to super-polynomial size using self-reducibility. However, while their work is focused on lower bounds for "strongly self-reducible" problems in NC, our results are much broader, addressing problems such as SAT for which their techniques do not seem to yield interesting consequences, and also addressing a greater variety of computational models such as general Boolean circuits, Boolean formulas, branching programs, sub-linear time algorithms, etc. Moreover, while the lower bounds they "amplify" are not known to hold for any explicit problems, the lower bounds from which we show magnification are in many cases known for

[1] We refer to a textbook such as [Juk12] for an exposition of these circuit classes and corresponding lower bounds.

explicit problems – just not for the specific problems we consider.[2] (We discuss additional related work in Section 1.3.)

On the other hand, a form of hardness magnification has been recently observed by Müller and Pich [MP17, Proposition 4.14] in the context of *proof complexity*. In more detail, their argument establishes that barely non-trivial lower bounds for bounded-depth Frege systems for certain tautologies of interest imply super-polynomial lower bounds for Frege systems.[3] The interesting aspect of this result compared to [AK10] is that lower bounds against bounded-depth Frege systems *are* known (but not for the same tautologies). Moreover, the tautologies from [MP17] have been studied in the past, and lower bounds in weaker proof systems have been obtained for the same class of tautologies. While our results are incomparable to their work in proof complexity, it served as a source of inspiration for our investigations.

The general template for our results is as follows. We consider various natural problems Q that are believed to be hard. For each such problem Q, we show:

(a) An implication from a weak lower bound for Q (i.e., a lower bound that is weak in magnitude and/or against a weak model) to a strong lower bound for a problem in NP or EXP.

(b) Under a standard hardness hypothesis, the weak lower bound for Q (and indeed a stronger lower bound) does hold. Thus magnification is a sound approach to proving strong lower bounds under standard hardness hypotheses.

(c) The weak lower bound is "barely non-trivial", in that a trivial lower bound that is only slightly smaller can be shown to hold. Moreover, in several cases, the weak bound is known for some explicit problem, just not for the problem Q we consider.

The details of the magnification results depend on the problems Q we consider and on the involved computational models. To establish magnification as a fairly general phenomenon, we consider a range of such problems:

- Variants of the Minimum Circuit Size Problem (MCSP), where the input is the truth table of a Boolean function and the question is whether the function has small circuits.
- Variants of the Minimum Kt Complexity Problem (MKtP), where the input is a string and the question is if the string has low Kt complexity, where Kt complexity is Levin's notion of Kolmogorov time bounded complexity.
- The Vertex Cover problem.
- Satisfiability (3-SAT).

Our most interesting magnification results are for "meta-computational" problems such as MCSP, MKtP and SAT, where the instance of the problem itself encodes a computation. This re-inforces the message of several works in complexity theory (e.g., [NW94, RR97, Wil13]) that understanding the complexity of meta-computational problems is closely associated with making progress on complexity lower bounds.

[2] Note that some researchers [Wil18] do not see compelling evidence that the weak lower bounds required by Allender and Koucký [AK10] hold.

[3] The result is not stated in this way, but the proof shows that weak lower bounds are sufficient.

Our results can be interpreted in different ways. For an optimist, they might give hope that strong lower bounds are achievable. First, the weak lower bounds from which we magnify are in several cases already known for explicit problems, so it's "merely" a question of showing similar lower bounds for the problems we consider. Second, approaches via magnification appear to avoid the natural proofs barrier to proving lower bounds. Recall that the natural proofs barrier [RR97] rules out circuit lower bounds given by dense and constructible properties that are useful against the circuit class, modulo standard cryptographic assumptions. Even if the weak lower bound that is required for magnification is shown via a natural property, the magnification step seems to destroy the density of the corresponding property, as magnification only seems to hold for certain special structured problems and not for generic ones. The argument here is similar to the argument of Allender and Koucký [AK10, Section 8] that "amplifying lower bounds via self-reducibility" evades the natural proofs barrier.

For the pessimist, magnification might simply be an indication that natural proofs and other barriers do not capture all obstacles to proving circuit lower bounds. In this view, magnification is simply an invitation to refine and extend our meta-mathematical understanding of circuit lower bounds so that we have compelling explanations of why lower bounds via magnification would be hard to achieve.

We see this as a win-win situation: either strong lower bounds can be shown via magnification, or magnification and similar phenomena will motivate us to gain a better understanding of the limitations of lower bound techniques.

1.2 Results and techniques

This section describes in more detail our main results and techniques. We often use a concrete choice of parameters to simplify the exposition. More general versions of these statements appear in the main body of the paper.

A magnification phenomenon around quasi-linear size lower bounds. We say that a function $f\colon \{0,1\}^n \to \{0,1\}$ is γ-approximable by a boolean circuit C if $\Pr_x[f(x) = C(x)] \geq \gamma$, where $x \sim \{0,1\}^n$. Let (α,β)-$\mathsf{MCSP}[s]$ be the promise problem whose YES instances are truth tables of functions that are α-approximable by a circuit of size $s(n)$, and whose NO instances are truth tables of functions that are not β-approximable by a circuit of size $s(n)$. We use $N = 2^n$ to denote the input length of (α,β)-$\mathsf{MCSP}[s]$, and consider the problem with $\alpha = 1$ and $\beta = 1-\delta$, where $\delta = \delta(n) > 0$ is a parameter that measures the gap between YES and NO instances of $(1,1-\delta)$-$\mathsf{MCSP}[s]$.

We use $\mathsf{Formula}[t]$ to denote the set of functions that can be computed by Boolean formulas of size at most t. Our first result can be stated as follows.

Theorem 1. *Let $\delta\colon \mathbb{N} \to \mathbb{R}$ and consider the problem $(1,1-\delta)$-$\mathsf{MCSP}[s]$. The following results hold.*

- *(i) Let $s(n) = n^k$ and $\delta(n) = n^{-k}$, where $k \in \mathbb{N}$. If $(1,1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{Formula}[N \cdot (\log N)^{O(1)}]$ then there is $L \in \mathsf{NP}$ over m-bit inputs such that $L \notin \mathsf{Formula}[\mathsf{poly}(m)]$.*
- *(ii) For the same choice of parameters, if $(1,1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{Formula}[N^{1+\varepsilon}]$ for some $\varepsilon > 0$, then there is $L \in \mathsf{NP}$ over m-bit inputs and $\delta > 0$ such that $L \notin \mathsf{Formula}[2^{m^\delta}]$.*

(*iii*) *Let* $s(n) = 2^{o(n)}$ *and* $\delta(n) = 2^{-o(n)}$. *if* $(1, 1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{Formula}[N^{1+\varepsilon}]$ *for some* $\varepsilon > 0$, *then there is* $L \in \mathsf{NP}$ *over* m*-bit inputs such that* $L \notin \mathsf{Formula}[\mathsf{poly}(m)]$.

Theorem 1 is a consequence of a more general result presented in Section 3.1. It magnifies super-linear size formula lower bounds between $N \cdot \mathsf{poly}(\log N)$ and $N^{1+\varepsilon}$ to much stronger lower bounds against formulas for a function in NP. As a consequence, minor improvements in lower bounds for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ around the linear-size regime would have major implications in our understanding of formula lower bounds for explicit problems.

Note that Theorem 1 asks for barely super-linear formula size lower bounds. We observe that there are no known algorithms for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ over N-bit inputs running in time $2^{o(s(n))}$. In particular, for $\delta > 1/2$ it is not known how to solve this problem in time polynomial over the input length N for any $n \ll s(n) \ll 2^{o(n)}$. Under standard cryptographic assumptions, it is possible to use the theory of natural proofs [RR97] to prove that $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ cannot be computed by circuits of polynomial size if $s(n) = n^k$ and k is sufficiently large (Proposition 19).

Let $\mathsf{AC}^0_d[t]$ denote depth-d AC^0 circuits of size t. We establish the following additional hardness magnification result.

Theorem 2. *Suppose there exists* $k \geq 1$ *such that for every* $d \geq 1$ *there is* $\varepsilon_d > 0$ *such that* $(1, (1-\delta))$-$\mathsf{MCSP}[s] \notin \mathsf{AC}^0_d[N^{1+\varepsilon_d}]$, *where* $s(n) = n^k$ *and* $\delta(n) = n^{-k}$. *Then* $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.

This result can be seen as an analogue in circuit complexity of the hardness magnification phenomenon observed in proof complexity by Müller and Pich [MP17, Proposition 4.14].

Perhaps intriguingly, much stronger formula size lower bounds are known for problems considerably simpler than $(1, 1-\delta)$-$\mathsf{MCSP}[s]$. For instance, it is well-known that the parity function over N input variables requires formulas of size $\Omega(N^2)$, and there are several techniques able to prove strong lower bounds against AC^0 circuits. Can we use these existing approaches to establish lower bounds for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$?

We sketch in Appendix Section C how known results and techniques imply the following lower bounds. Let $1/2 \leq \gamma < 1$ be an arbitrary constant. Then, for some choice of $s(n) = 2^{\Theta(n^\gamma)}$ and $\delta(n) = 2^{-n+\Theta(n^\gamma)}$, we have $(1, 1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{Formula}[N^{2-o(1)}]$. Note that this lower bound is not good enough for hardness magnification (i.e., Theorem 1) because the parameter $\delta(n)$ is not large enough. On the other hand, we note that if $s(n) = n^{\omega(1)}$ and $\delta < 1/2$ is fixed, then for every $d \geq 1$ and every $\ell \geq 1$ we have $(1, 1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{AC}^0_d[N^\ell]$. Observe that this lower bound is not good enough for hardness magnification (i.e., Theorem 2) because the parameter $s(n)$ is too large.

We briefly discuss the techniques behind the proof of Theorem 1, which is not technically involved but requires certain crucial insights. The argument is by contrapositive, so we assume that every problem in NP on m input bits has formulas of bounded size. In order to solve $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ by almost-linear size formulas, we proceed as follows. We take a projection of the input truth table on a certain number of random locations, and define a "compressed" language L in NP over m input bits, where $m \approx s(n)/\delta(n) \ll N = 2^n$, which captures a succinct version of the initial problem. Using the gap parameter δ, it is possible to show that if the input truth table admits small circuits, so does its random projection. On the other hand, a probabilistic argument proves that if the input truth table cannot be approximated by small circuits, then with high probability the projected truth table does not admit small approximating circuits. This allows us to employ small formulas for L to solve the original problem. However, this argument is probabilistic, and the reduction sketched before is randomized. In order to get almost-linear size formulas for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$, we derandomize this construction in an elementary but careful way using non-uniform

formulas. While the formulas for L have small size as a function of m when compared to N, the non-constructive derandomization argument is responsible for the almost-linear size bounds (in N) for the formulas obtained for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$.

We observe that the argument is inspired in part by Occam's Razor, an idea from learning theory (see e.g. [KV94]). The details are in Section 3.1, where the proof of Theorem 2 is also presented.

Results similar to Theorem 1 can be proved with respect to boolean circuits and indeed for a variety of boolean devices. The main advantage of presenting these ideas in the context of formula complexity is because for formulas there are non-trivial polynomial lower bounds for several explicit problems, and a large number of techniques have been developed for establishing such lower bounds (cf. the discussion in Section 5).[4]

Kt Complexity and lower bounds for EXP. Note that the previously considered problem concerns the *average-case* complexity of strings, viewed as truth tables. We show next a magnification result for a computational problem related to the *worst-case* complexity of strings.

Let U be a fixed universal machine. For a non-empty string $x \in \{0,1\}^*$, $Kt(x)$ is the minimum over $|p| + \log(t)$ such that $U(p)$ outputs x within t steps. Given functions $s, c \colon \mathbb{N} \to \mathbb{N}$ where $s \geq c$, let $\mathsf{MKtP}[c, s]$ be the promise problem over N-bit inputs whose YES instances are strings of Kt complexity [Lev84] at most $c(N)$ and NO instances are strings of Kt complexity greater than $s(N)$.

Let $\mathsf{SIZE}[t]$ denote the class of boolean circuits of size at most t. We establish the following result.

Theorem 3. *If there is a fixed $\delta > 0$ such that for each $\varepsilon > 0$, $\mathsf{MKtP}[N^\varepsilon, N^\varepsilon + 5\log(N)]$ does not have circuits of size $N^{1+\delta}$, then $\mathsf{EXP} \not\subseteq \mathsf{SIZE}(\mathrm{poly})$.*

It is not hard to see that $\mathsf{MKtP}[N^\varepsilon, N^\varepsilon + 5\log(N)] \in \mathsf{EXP}$. Moreover, results from [ABK+06] show that this problem is hard for EXP under non-uniform reductions. In particular, if EXP does not have polynomial size circuits, neither does this problem. This allows us to get the following *equivalence* as a consequence.

Corollary 4. *$\mathsf{MKtP}[N^\varepsilon, N^\varepsilon + 5\log(N)]$ has polynomial-size circuits for each $\varepsilon > 0$ iff for all $\delta > 0$ there is an $\varepsilon > 0$ such that $\mathsf{MKtP}[N^\varepsilon, N^\varepsilon + 5\log(N)]$ has circuits of size $N^{1+\delta}$.*

In terms of techniques, the proof of Theorem 3 adapts the ideas behind Theorem 1 and combines them with a new ingredient: highly efficient error-correcting codes that can be encoded and decoded by algorithms of quasi-linear complexity [Spi96]. Such error-correcting codes can be used together with the notion of Kt complexity to reduce the proof of Theorem 3 to a scenario that is similar to the one in the proof of Theorem 1. An important distinction is that we reduce to a certain compressed language that is in EXP instead of NP, and the argument uses boolean circuits instead of boolean formulas.

Theorem 3 and Corollary 4 are proved in Section 3.2.

Magnification from sub-linear average-case lower bounds. The results discussed above require lower bounds that are super-linear in the number N of input bits. Our next theorem shows that in some settings hardness magnification also follows from *sub-linear* lower bounds.

[4] In some computational models, we can work with a two-sided version (α, β)-$\mathsf{MCSP}[s]$ instead of $(1, 1-\delta)$-$\mathsf{MCSP}[s]$. The only issue here is the complexity of the non-uniform derandomization.

Let $\mathsf{MCSP}[s]$ be the (standard) Minimum Circuit Size Problem over N-bit inputs whose YES instances are truth tables of functions with circuit complexity at most $s(n)$. We consider the most natural notion of average-case complexity for MCSP over the uniform distribution, as introduced by [HS17]. A (zero-error) average-case algorithm or device A for $\mathsf{MCSP}[s]$ is a deterministic procedure that always outputs a value in $\{0, 1, \text{“?”}\}$, and that satisfies the following conditions: (1) A is never incorrect; and (2) $\Pr_x[A(x) \neq \text{“?”}] \geq 1/2$. (The constant $1/2$ is arbitrary.) In other words, the procedure provides a $0/1$ answer for a non-trivial fraction of input truth tables, and never makes a mistake.

Theorem 5. *If $\mathsf{MCSP}[2^{\sqrt{n}}]$ on N-bit inputs cannot be solved on average with zero error by formulas of size $o(N)$, then NP does not have polynomial size formulas.*

It is not hard to show that $\mathsf{MCSP}[s]$ is not in $\mathsf{SIZE}[\mathsf{poly}(N)]$ on average under standard cryptographic assumptions. Indeed, as observed in [HS17], the existence of average-case algorithms for MCSP is equivalent to the existence of natural proofs.

This magnification result is in sharp contrast to a recent lower bound from [HS17] showing that $\mathsf{MCSP}[2^{\sqrt{n}}]$ cannot be solved in the worst case by formulas of nearly quadratic size. As a consequence, there is a dramatic distinction between establishing *worst-case super-linear* lower bounds and establishing *average-case sub-linear* lower bounds for MCSP. Also note that much stronger $N^{3-o(1)}$ average-case formula size lower bounds are known for explicit problems [KRT17].

The proof Theorem 5 is elementary, and is reminiscent of an idea used in the proof of the main result from [RR97]. Our new conceptual insight is in exploring this argument from the perspective of hardness magnification. This and other related results are discussed in Section 3.3, and we refer to that section for more details.

Sub-linear randomized lower bounds and NP vs. BPP. Our next result in the sub-linear regime is with respect to worst-case probabilistic computations. We say that a randomized algorithm A computes a function f if on every input x, $\Pr[A(x) = f(x)] \geq 2/3$. For the definition of (sub-linear) randomized computation, we take any uniform model of computation that allows random-access to the input string.

Recall the definition of the problem (α, β)-$\mathsf{MCSP}[s]$ introduced above. We show the following magnification result in the setting of sub-linear randomized computations.

Theorem 6. *Let $1/2 < \beta < \alpha \leq 1$ be constants, $c \in \mathbb{N}$, and $s(n) \leq 2^{n^\gamma}$, where $\gamma < 1$. If there is $\varepsilon > 0$ such that (α, β)-$\mathsf{MCSP}[s]$ on N-bit inputs cannot be computed by a (two-sided error) randomized algorithm running in time $2^{(\log N)^{1-\varepsilon}}$, then $\mathsf{NP} \not\subseteq \mathsf{BPP}$.*

We note that stronger lower bounds are known against randomized computations with random-access to the input string for other (explicit) computational problems. Indeed, any reasonable model of computation of this form can be simulated by (non-uniform) randomized branching programs. If the original randomized algorithm uses time T and memory S, so does the corresponding distribution of deterministic branching programs (i.e., the longest path has length $\leq T$ and there are $\leq 2^S$ nodes in any branching program in the support of the distribution). In [BSSV03, Corollary 6.7], it is proved that an explicit N-bit boolean function in P requires two-sided error randomized branching programs running in super-linear time $T(N) = \omega(N)$ if the number of nodes in the branching program is $\leq 2^{N^{1-\Omega(1)}}$. This lower bound is much stronger than the one required in Theorem 6.

The proof of Theorem 6 is along the lines of the random projection argument sketched for the proof of Theorem 1. We refer the reader to Section 3.4.

Vertex Cover and Magnification. The results discussed before concerned meta-computational problems such as MCSP and its variants. We discuss now a hardness magnification result for Vertex Cover, a well-known graph-theoretic problem in NP.

Recall that in the Vertex-Cover problem, the input is a pair (G, w), where $G \in \{0,1\}^{\binom{n}{2}}$ encodes the adjacency matrix of an undirected n-vertex graph $G = (V, E)$, and $w \in \{0,1\}^{\log n}$ encodes in binary an integer parameter $k \geq 0$. The pair (G, w) is a positive instance of Vertex-Cover if there exists $S \subseteq V$, $|S| \leq k$, such that every $e = \{u, v\} \in E$ intersects S. Similarly, for $k = k(n)$ we let k-Vertex-Cover be the same computational problem with the exception that the parameter k is fixed in advance and is not part of the input. We use $m = \Theta(n^2)$ to denote the total input length of an instance of k-Vertex-Cover on n-vertex graphs.

Theorem 7. *Let $k(n) = (\log n)^C$, where $C \in \mathbb{N}$ is arbitrary. If for every $d \geq 1$ there exists $\varepsilon > 0$ such that k-Vertex-Cover $\notin \mathsf{AC}^0_d[m^{1+\varepsilon}]$, then $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.*

Under ETH, k-Vertex-Cover cannot be solved in time $2^{o(k)} \cdot \mathsf{poly}(m)$ ([IPZ01]; see e.g. [DF13, Theorem 29.5.9]). Therefore, it is plausible that k-Vertex-Cover is not in P if $k = \omega(\log n)$. Moreover, using the NP-completeness of Vertex-Cover and a simple translation argument, for any $\varepsilon > 0$ there is a constant C such that if $k = (\log n)^C$, then k-Vertex-Cover is not in P unless CNF-SAT $\in \mathsf{DTIME}[2^{n^\varepsilon}]$. Consequently, while Theorem 7 asks for a barely non-trivial lower bound in a very weak circuit model, standard assumptions imply much stronger results.

For larger $k(n)$, we can establish a connection to time-space trade-off results (see e.g. [FLvMV05, Wil07]). The next statement assumes a model where the algorithm has random-access to the input string. This is compatible with existing time-space trade-offs. For concreteness, one can use Turing Machines with a read-only input tape, a constant number of working tapes, and a special tape used to address the input string. Recall that $\mathsf{DTISP}[t(n), s(n)]$ denotes the class of languages that can be decided by an algorithm that simultaneously runs in time $O(t(n))$ and space $O(s(n))$.

Theorem 8. *Let $k(n) = n^{o(1)}$. If there exists $\varepsilon > 0$ such that k-Vertex-Cover $\notin \mathsf{DTISP}[m^{1+\varepsilon}, m^{o(1)}]$, where the input is an n-vertex graph represented by an adjacency matrix of bit length $m = \Theta(n^2)$, then $\mathsf{P} \neq \mathsf{NP}$.*

The reader familiar with time-space trade-offs might recall that it is known unconditionally that the Vertex Cover problem is not in $\mathsf{DTISP}[m^{1.8}, m^{o(1)}]$ (see e.g. [Wil07]). However, such statement has two important differences in comparison to Theorem 8. First, the input encoding is different. Existing lower bounds employ a list of vertices followed by a list of edges. Secondly, the parameter $k(n)$ is much closer to n.

The techniques employed in the proof of these two results explore a connection to *kernelization*, a widely-investigated method from parameterized complexity (see e.g. [DF13]). Recall that a kernelizable (parameterized) problem can be reduced in polynomial time to a much smaller instance of itself, where the new input size m' can be upper bounded by a function of k. It is well-known that k-Vertex-Cover is kernelizable. We explore this idea from the point of view of *hardness magnification*. Indeed, while kernelizablity has been used mainly algorithmically, our insight here is that an extremely efficient implementation has consequences for lower bounds.

In order to explain the approach, we sketch some high-level ideas used in the proof of Theorem 7. The result is established in the contrapositive. In other words, under the assumption that $\mathsf{NP} \subseteq \mathsf{NC}^1$, we must compute k-Vertex-Cover using almost-linear size constant-depth circuits. Let T be a kernelization routine for k-Vertex-Cover which maps inputs represented by m bits and

with a parameter k to an instance of Vertex-Cover whose size is $\mathsf{poly}(k)$. Since by assumption NP is contained in NC^1, standard results imply that NP can be computed by sub-exponential size circuits of sufficiently large constant depth. Consequently, using that Vertex-Cover $\in$ NP and $k(n) = (\log n)^C$, we can solve the new instance obtained via kernelization by constant-depth circuits of size at most m (where $m = \Theta(n^2)$ is the original input length). Our argument would be complete if we could also implement the routine T using almost-linear size AC^0 circuits. This part of the argument requires low-level implementations, and relies on fairly efficient computations that can be done by such circuits with a sub-linear number of (unbounded fan-in) gates.

Theorems 7 and 8 are formally established in Section 4.1.

Nontrivial lower bounds for Satisfiability and NEXP vs. BPP. Our last hardness magnification example holds for the standard formulation of the 3-SAT problem. However, unlike the earlier results which magnify from worst-case or average-case lower bounds, the main result in this section magnifies from lower bounds against polynomial-time refuters, a notion defined by Kabanets [Kab01].

Intuitively, the theorem says that if any probabilistic quasi-linear time algorithm for SAT can be efficiently refuted in the sense that there is a polynomial-time algorithm that infinitely often produces either YES instances on which the algorithm does not accept with high probability or NO instances for which the algorithm does not reject with high probability, then NEXP is different from BPP. The antecedent here is a "mild" lower bound against quasi-linear time algorithms, while the consequent is a more significant lower bound against any polynomial-time algorithm, but for a larger class. While the requirement to prove a lower bound against refuters might appear much stronger than the requirement to prove a worst-case lower bound, the work of [GST07] (see also [Ats06]) shows that these requirements are *essentially equivalent* when the refuter has more resources than the algorithm, as in our case. The reason we are unable to relax our requirement to a worst-case lower bound is that it is important in our argument that the refuter is deterministic while the algorithm is allowed to be randomised.

We use $|\phi|$ to denote the bitlength complexity of a 3-SAT formula ϕ.

Theorem 9. *Suppose that for every probabilistic machine M that on inputs of length m halts in time $m \cdot \mathrm{polylog}(m)$ and uses $\mathrm{polylog}(m)$ many random bits, there is a deterministic polynomial-time machine N such that for infinitely many m, $N(1^m)$ outputs a 3-CNF formula ϕ_m where $|\phi_m| = m$ for which either ϕ_m is satisfiable and $M(\phi_m)$ rejects with probability $> 1/m$ or ϕ_m is unsatisfiable and $M(\phi_m)$ accepts with probability $> 1/m$. Then $\mathsf{NEXP} \not\subseteq \mathsf{BPP}$.*

Modulo the use of refuters, Theorem 9 provides a bridge between two frontiers in complexity theory: establishing non-trivial lower bounds for Satisfiability against unrestricted algorithms, and understanding the power of randomness in computation. Note that much stronger running time lower bounds are know for Satisfiability in the standard sense (i.e., without refuters), but they only hold under the assumption that the algorithm uses a sub-linear amount of memory (see e.g. [FLvMV05, Wil07]).

This is our most technically demanding proof. In terms of results, the argument relies on efficient versions of the PCP theorem where the reduction runs in quasi-linear time, and on the Easy Witness Lemma of [IKW02]. Other concepts that play a role are the notion of Kt complexity discussed above, a related notion of KT Kolmogorov complexity introduced by [All01], and a random projection of clauses that defines a randomized reduction to a certain compressed satisfiability problem. Instead of presenting a high-level exposition of the proof, we discuss some analogies with the proof of

Theorem 1, which motivated our main ideas. (The argument sketched below assumes background in complexity theory.)

The proof of Theorem 9 is also by contrapositive. Suppose that $\mathsf{NEXP} = \mathsf{BPP}$. Recall that for Theorem 1 we considered a version of the MCSP problem with a *gap* between positive and negative instances. One should think of each clause of the 3-SAT instance ϕ as an entry in the input truth table for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$. If we were able to create a gap between positive (satisfiable) and negative (unsatisfiable) instances of 3-SAT, we could try to exploit some of the ideas employed in the MCSP context. Under this analogy, this turns out to be precisely what is granted by the PCP theorem. (We use a very efficient version of the PCP theorem that does not affect our barely super-linear complexity requirements.) Assume from now on that we need to solve 3-SAT on instances with a gap: YES instances are satisfiable, while in any NO instance at most a $(1-\varepsilon)$-fraction of the clauses can be simultaneously satisfied.

For the next step of the analogy, the reader should relate circuits to assignments, and small circuits to assignments that are encoded by strings of low complexity. From this perspective, a random projection in the context of $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ corresponds here to a random projection of input clauses. Similarly to the proof of Theorem 1, we reduce the problem to a certain compressed language L over smaller input strings, and use the complexity collapse to solve L very efficiently. There is however a crucial difference: while in the $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ setting we only had to care about potential small circuits computing the input truth table, for 3-SAT we care about *all* possible satisfying assignments, and not only about those of "low complexity". In order to address this, the Easy Witness Lemma and the notion of refuters play a fundamental role.

The high-level idea is that the refuter either produces (1) a unsatisfiable formula; (2) a satisfiable formula that admits a satisfying assignment of low complexity; or (3) a satisfiable formula such that every satisfying assignment has high complexity. Since we are arguing in the contrapositive, we need to claim that no refuter succeeds. By carefully designing the compressed/sketched version of the 3-SAT problem used in our reduction, it is possible to rule out cases (1) and (2) using an analysis that is similar to the proof of Theorem 1. On the other hand, in the remaining case (3) we get that the (deterministic) refuter produces a sequence of infinitely many formulas that are satisfiable but do not admit satisfying assignments of low-complexity. By appropriately defining the notions of complexity involved in the argument, we are able to explore this refuter to define a language in NEXP that does not admit succinct witnesses in the sense of [IKW02]. By their Easy Witness Lemma, this implies (in particular) that $\mathsf{NEXP} \neq \mathsf{BPP}$, which contradicts our initial assumption.

A detailed exposition of the proof of Theorem 9 appears in Section 4.2.

We defer the discussion of some concrete directions and open problems to Section 5.

1.3 Further remarks and related work

Artificial Constructions. The magnification theorems presented in the previous section hold for several *natural* problems. We note that in the deterministic worst-case setting more dramatic magnification theorems can be established for problems that are defined in an artificial way. In particular, under plausible hardness assumptions, there are explicit problems over n-bit inputs that are not in P, but showing that they require circuits of size $(1+\varepsilon)n$ in the worst-case implies

$\mathsf{NP} \not\subseteq \mathsf{SIZE}[2^{n^{o(1)}}]$.[5] Unfortunately, this does not offer a better approach to lower bounds. We are not aware of a magnification result from $C \cdot n$ size lower bounds for problems of practical or theoretical interest, where C is a constant (in the deterministic worst-case setting).

Terminology. A few comments on the "magnification" terminology are in order. As opposed to other similar names considered in the literature, we use this notation to refer to results where a lower bound that is small in value with respect to a certain measure is magnified to a lower bound of larger value with respect to the same complexity measure. In some cases, our results also "escalate" to a more expressive computational model, such as in Theorems 2 and 7 (this phenomenon is observed in the so-called "lifting" theorems from communication complexity and proof complexity). Hardness magnification is in particular different from standard hardness "amplification" results such as the well-known XOR Lemma and its variants (where average-case hardness is amplified, but size bounds slightly decrease). Due to these distinctions, we believe that the term "magnification" describes our set of results in a more appropriate way.

Related work on magnification. In addition to the papers [AK10] and [MP17] mentioned above, we are aware of three other works showing magnification theorems in our sense.

Srinivasan [Sri03] considered the problem of $n^{1-o(1)}$-approximating the size of the maximum clique in a graph, and showed that barely super-linear lower bounds against randomized algorithms imply separations such as $\mathsf{NP} \not\subseteq \mathsf{BPP}$. We refer to his work for several variants of this result. In terms of techniques, Srinivasan's proof also uses random projections, though the technical details are different compared for instance to our Theorem 1. A drawback of his magnification theorem is that it seems to be very hard to prove unconditional lower bounds for this approximating problem, not to mention the difficulty of analysing the corresponding computational model. To our knowledge, [Sri03] was the first to show a magnification phenomenon for a natural problem previously considered in the literature.

Motivated by [AK10], Lipton and Williams [LW13] established a magnification result for the Circuit Evaluation Problem in the context of almost-linear time and sub-linear space algorithms (similar in spirit to Theorem 8, but with different parameters). In contrast, their result offers an approach to separating P from NC. The proof uses different ideas compared to our Theorem 8, and we refer to their work for more details and related results.

More recently, [CILM18] (see also [HWY11]) established a magnification theorem in non-commutative arithmetic circuit complexity. Interestingly, their results show the existence of a generic magnification phenomenon in this setting, where any explicit lower bound can be magnified. The argument seems to explore non-commutativity in a fundamental way.

2 Preliminaries

2.1 Basic definitions

Let $\mathcal{F}_n$ be the family of all boolean functions $f\colon \{0,1\}^n \to \{0,1\}$, and $\mathfrak{F} = \{\mathcal{F}_n\}_n$. We can represent Boolean functions as strings via the truth table mapping. Given a Boolean function $f \in \mathcal{F}_n$, $\mathtt{tt}(f)$ is the 2^n-bit string which represents the truth table of f in the standard way, and conversely, given a string $y \in \{0,1\}^{2^n}$, $\mathtt{fn}(y)$ is the Boolean function in $\mathcal{F}_n$ whose truth

[5] This can be shown, for instance, using a padding argument and assuming ETH (Exponential Time Hypothesis).

table is represented by y. For convenience, a boolean function f is often identified with the set $f^{-1}(1) \subseteq \{0,1\}^n$.

Let $\mathfrak{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a class of Boolean functions, where each $\mathcal{C}_n \subseteq \mathcal{F}_n$. Given a language $L \subseteq \{0,1\}^*$, we write $L \in \mathfrak{C}$ if for every large enough n we have that $L_n \stackrel{\text{def}}{=} \{0,1\}^n \cap L$ is in $\mathcal{C}_n$. Often we will abuse notation and view $\mathfrak{C}$ as a class of Boolean circuits. We use number of gates to measure circuit size. We denote by $\mathfrak{C}_d[s]$ the set of $\mathfrak{C}$-circuits of size at most $s = s(n)$ and depth at most $d = d(n)$. As usual, we say that a uniform complexity class Γ is contained in $\mathfrak{C}[\mathsf{poly}(n)]$ if for every $L \in \Gamma$ there exists $k \geq 1$ such that $L \in \mathfrak{C}[n^k]$. We say that $\mathfrak{C}$ is *typical* if $\mathfrak{C} \in \{\mathsf{AC}^0, \mathsf{AC}^0[p], \mathsf{ACC}^0, \mathsf{TC}^0, \mathsf{NC}^1, \mathsf{Formula}, \mathsf{Circuit}\}$. We sometimes use $\mathsf{SIZE}[s]$ to refer to boolean circuits of size at most s.

We will use a few different variants of the Minimum Circuit Size Problem in our results. We refer to Section A for definitions.

2.2 Notions of Kolmogorov Complexity

For the next definitions, we fix a computational model with random-access to the input string. Indeed, whenever we consider almost linear time computations we refer to algorithms that can access their input bits in an efficient way. The particular details of the computational model are not so relevant, since we won't distinguish polylogarithmic factors in the context of uniform computations.

Definition 10. *Let U be a universal machine. For any string $x \in \{0,1\}^*$, $Kt(x)$ is the minimum over $|p| + \log(t)$ such that $U(p)$ outputs x within t steps.*

Definition 11. *Let U be a universal machine. For any string $x \in \{0,1\}^*$, $KT(x)$ is the minimum over $|p|+t$ such that $U(p,i) = x_i$ in at most t steps for each $i, 1 \leq i \leq |x|$, and moreover $U(p,i) = \perp$ for all $i > |x|$.*

The following inequality is Proposition 1 in [All01], where KT-complexity was introduced.

Proposition 12 ([All01])**.** *For all $x \in \{0,1\}^*$, $Kt(x) \leq 3KT(x)$.*

2.3 Technical results

We state below some technical results that will be used in different proofs.

Lemma 13 (Concentration Bound; cf. [JŁR00, Theorem 2.1])**.** *Let $X \sim \mathsf{Bin}(m,p)$ and $\lambda = mp$. For any $t \geq 0$,*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \exp\left(-\frac{t^2}{2(\lambda + t/3)}\right).$$

Lemma 14 (Folklore; cf. [AHM+08, Lemma 8.1])**.** *Let $\{C_n\}, n \in \mathbb{N}$ be a sequence of fan-in two Boolean circuits over n input variables and of depth $D = O(\log n)$, and let $d \geq 2$ be a constant. There is a sequence $\{D_n\}$ of depth-d AC^0 circuits of size $2^{n^{O(1/(d-1))}}$ such that for each $n \in \mathbb{N}$, D_n computes the same Boolean function as C_n.*

3 Hardness Magnification for MCSP and Related Problems

3.1 Non-uniform formula lower bounds

We use number of leaves to measure formula size, and let $\mathsf{Formula}[s]$ denote the set of functions computed by formulas of size at most s. We assume negations appear at the input leaves only, and that constants 0 and 1 are available in addition to the input literals. We will need the following standard result.

Proposition 15 (Bound on the Number of Circuits)**.** *Let $s \geq n$ and $n \geq c$, where c is a large enough constant. There are at most $2^{3s \log s}$ different circuits on n input variables of size at most s.*

Proof. Since $s \geq n$, each (fan-in two) circuit of this size can be represented by a boolean string of length $\leq 3s \log s$ that lists its collection of wires and gate types. □

Recall that for string $w \in \{0,1\}^N$ and $N = 2^n$, we use $\mathsf{fn}(w)\colon \{0,1\}^n \to \{0,1\}$ to denote the boolean function encoded by w.

Lemma 16 (Magnification Lemma for Formulas)**.** *Let $s\colon \mathbb{N} \to \mathbb{N}$ and $\delta\colon \mathbb{N} \to [0, 1/2)$ be constructive functions, where $s(n) \geq n$. Consider the promise problem $\Pi[s, \delta] = (1, 1-\delta)$-$\mathsf{MCSP}[s]$, where for every $N = 2^n$,*

$$\begin{aligned} \Pi_N^{\mathsf{yes}} &= \{y \in \{0,1\}^N \mid \exists \text{ circuit of size } \leq s(n) \text{ that computes } \mathtt{fn}(y)\} \\ \Pi_N^{\mathsf{no}} &= \{y \in \{0,1\}^N \mid \nexists \text{ circuit of size } \leq s(n) \text{ that } (1-\delta(n))\text{-approximates } \mathtt{fn}(y)\}. \end{aligned}$$

Then, for every function $\ell\colon \mathbb{N} \to \mathbb{N}$, if $\Pi[s, \delta]$ cannot be computed by formulas of size $N \cdot \ell(n)$, then there is a sequence $\{f_{m(n)}\}_{n \geq 1}$ of $m(n)$-bit functions in NP that cannot be computed by formulas of size $\leq \ell(n)$, where $m(n) = \Theta((n \cdot s(n) \cdot \log s(n))/\delta(n))$.

Proof. We consider the computational problem $\mathtt{Succinct}$-$\Pi[s, t]$, defined next. This is essentially $\mathsf{Succinct}$-MCSP (Section A), but in order to be formal we make the input encoding fully explicit. The input instances are of the form $\langle 1^n, 1^{s(n)}, (x_1, b_1), \ldots, (x_{t(n)}, b_{t(n)})\rangle$, where $x_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$, $i \in [t(n)]$. Any instance of this form can be encoded by a string of length exactly $m(n) = n + 1 + s(n) + 1 + t(n) \cdot (n+1)$. The function $\mathtt{Succinct}$-$\Pi[s, t]\colon \{0,1\}^* \to \{0,1\}$ evaluates to 1 on an input string if and only if it is of the form above and there exists a circuit C over n input variables and of size at most s such that $C(x_i) = b_i$ for all $i \in [t(n)]$. Note that the problem is in NP as a function of its total input length m, provided that s and t are constructive functions.

From now on, let $t(n) \stackrel{\text{def}}{=} (3s(n) \log s(n))/\delta(n)$. Assume there exists a sequence $\{F_{m(n)}\}_{n \geq 1}$ of formulas of size $\ell(n)$ computing $\mathtt{Succinct}$-$\Pi[s, t]$, where $F_{m(n)}\colon \{0,1\}^{m(n)} \to \{0,1\}$. Recall that $N(n) = 2^n$. We construct randomized formulas E_N that separate Π_N^{yes} and Π_N^{no}. More precisely, each formula $E_N\colon \{0,1\}^N \times \{0,1\}^{n \cdot t(n)} \to \{0,1\}$ will satisfy the following conditions:

$$\text{If } w \in \Pi_N^{\mathsf{yes}} \implies \Pr_{\boldsymbol{y}}[E_N(w, \boldsymbol{y}) = 1] = 1, \tag{1}$$

$$\text{If } w \in \Pi_N^{\mathsf{no}} \implies \Pr_{\boldsymbol{y}}[E_N(w, \boldsymbol{y}) = 1] < 1/2. \tag{2}$$

Each formula E_N computes as follows. It interprets its random input y as a sequence of $t(n)$ strings $y_1, \ldots, y_{t(n)} \in \{0,1\}^n$. From such strings and the input string $w \in \{0,1\}^N$, E_N produces an instance $z_{w,y} \in \{0,1\}^{m(n)}$ of $\mathtt{Succinct}$-$\Pi[s, t]$ given by

$$z_{w,y} \stackrel{\text{def}}{=} \langle 1^n, 1^{s(n)}, (y_1, (\mathsf{fn}(w))(y_1)), \ldots, (y_{t(n)}, (\mathsf{fn}(w))(y_{t(n)}))\rangle,$$

where $(\mathsf{fn}(w))(y_i)$ is the value of the function represented by w on the input y_i, i.e., the bit of w indexed by y_1. Finally, the formula E_N outputs $F_{m(n)}(z_{w,y})$.

We argue next that conditions (1) and (2) are satisfied by E_N. If $w \in \Pi_N^{\mathsf{yes}}$, then w is computed by some circuit C of size at most $s(n)$. Consequently, for every choice y of the random string $\boldsymbol{y}$, C is also consistent with the input pairs encoded by $z_{w,y}$. It follows that $E_N(w,y) = F_{m(n)}(z_{w,y}) = 1$. On the other hand, let $w \in \Pi_N^{\mathsf{no}}$ be a negative instance, and fix an arbitrary circuit C on n input variables and of size at most $s(n)$. Since $\Pr_{\boldsymbol{x}}[C(\boldsymbol{x}) \neq (\mathsf{fn}(w))(\boldsymbol{x})] \geq \delta(n)$, it follows that the probability that C agrees with $\mathsf{fn}(w)$ on $t(n)$ independent random inputs $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t(n)}$ is less than $(1-\delta(n))^{t(n)} \leq e^{-\delta(n)t(n)} \leq e^{-3s(n)\log(s(n))}$, using our choice of $t(n)$. Therefore, by a union bound on the number of circuits of size at most $s(n)$ (Proposition 15), it follows that condition (2) is also satisfied.

Our next step is a derandomization of the formulas E_N using a standard argument. Let G_N be the formula that computes the conjunction of N independent copies of E_N. In other words,

$$G_N \colon \{0,1\}^N \times (\{0,1\}^{n\cdot t(n)})^N \to \{0,1\} \text{ and } G_N(w, y^{(1)}, \ldots, y^{(N)}) \stackrel{\text{def}}{=} \bigwedge_{i=1}^{N} E_N(w, y^{(i)}).$$

Clearly, $G_N(w, \vec{\boldsymbol{y}})$ accepts a string $w \in \Pi_N^{\mathsf{yes}}$ with probability 1. On the other hand, it accepts a string $w \in \Pi_N^{\mathsf{no}}$ with probability strictly less than 2^{-N}. Fix $\alpha \in \{0,1\}^{N\cdot(n\cdot t(n))}$ to be a string that correctly derandomizes G_N on inputs from $\Pi_N^{\mathsf{yes}} \cup \Pi_N^{\mathsf{no}} \subseteq \{0,1\}^N$, and let $G_N^\star$ be a formula computing the corresponding function $G_N^\star \colon \{0,1\}^N \to \{0,1\}$.

In order to complete the proof, it is enough to upper bound the formula size of $G_N^\star$. Recall that $F_{m(n)}$ has formula size $\ell(n)$. Each component $E_N(w, y^{(i)})$ of $G_N^\star$ has been fixed to $E_N(w, \alpha^{(i)})$, where $\alpha^{(i)} \in \{0,1\}^{n\cdot t(n)}$ denotes the i-th section of the string α. Therefore, it is enough to replace the leaves of the formula $F_{m(n)}$ by constants and appropriate literals formed from w in order to compute $E_N(w, \alpha^{(i)})$. It follows that each $E_N(w, \alpha^{(i)})$ can be computed by a formula of size at most $\ell(n)$. Consequently, $G_N^\star$ can be computed by a formula containing at most $N \cdot \ell(n)$ leaves. □

Note that in the proof above we used in a crucial way that $G_N^\star$ does not need to compute addressing functions to perform the projection. This would result in larger formulas.

Below we instantiate Lemma 16 with some representative settings of parameters.

Theorem 17 (Magnification of Weak Formula Lower Bounds)**.** *For convenience, let* $\Pi[s,\delta] = (1, 1-\delta)\text{-}\mathsf{MCSP}[s]$*. The following implications hold.*

(*i*) *Let* $s(n) = n^k$ *and* $\delta(n) = n^{-k}$*, where* $k \in \mathbb{N}$*. If* $\Pi[s,\delta] \notin \mathsf{Formula}[N \cdot (\log N)^{O(1)}]$ *then there is* $L \in \mathsf{NP}$ *over* m*-bit inputs such that* $L \notin \mathsf{Formula}[\mathsf{poly}(m)]$*.*

(*ii*) *For the same choice of parameters, if* $\Pi[s,\delta] \notin \mathsf{Formula}[N^{1+\varepsilon}]$ *for some* $\varepsilon > 0$*, then there is* $L \in \mathsf{NP}$ *over* m*-bit inputs and* $\delta > 0$ *such that* $L \notin \mathsf{Formula}[2^{m^\delta}]$*.*

(*iii*) *Let* $s(n) = 2^{o(n)}$ *and* $\delta(n) = 2^{-o(n)}$*. if* $\Pi[s,\delta] \notin \mathsf{Formula}[N^{1+\varepsilon}]$ *for some* $\varepsilon > 0$*, then there is* $L \in \mathsf{NP}$ *over* m*-bit inputs such that* $L \notin \mathsf{Formula}[\mathsf{poly}(m)]$*.*

Proof. Immediate from Lemma 16. □

Let $\mathsf{AC}^0_d[M]$ denote the class of unbounded fan-in depth-d size-M circuits. The next result shows that even very weak bounded-depth circuit lower bounds imply super-polynomial formula-size lower bounds.

Theorem 18 (Formula Lower Bounds from Weak AC^0 Lower Bounds)**.** *Suppose there exists $k \geq 1$ such that for every $d \geq 1$ there is $\varepsilon_d > 0$ such that $(1, 1-\delta)$-$\mathsf{MCSP}[s] \notin \mathsf{AC}^0_d[N^{1+\varepsilon_d}]$, where $s(n) = n^k$ and $\delta(n) = n^{-k}$. Then $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.*

Proof Sketch. We employ the same approach and notation of Lemma 16. Suppose that $\mathsf{NP} \subseteq \mathsf{NC}^1$, and let $\{F_m\}$ be a family of polynomial size formulas for the corresponding problem $\texttt{Succinct-}\Pi[s,t]$. Take $d' \in \mathbb{N}$ large enough so that each F_m can be computed by a depth-d' circuit of size, say, $2^{O(\sqrt{n})}$ (Lemma 14). Note that a fixed d' independent of n with this property must exist, since for our choice of parameters each formula F_m has $\mathsf{poly}(n)$ input variables. Proceeding exactly as in the proof of Lemma 16 but using bounded-depth circuits instead of formulas, we get that for some $d \in \mathbb{N}$, $\Pi[s,\delta]$ can be computed by depth-d circuits of size $N^{1+o(1)}$. This completes the proof. □

We can show under standard cryptographic assumptions and an appropriate choice of parameters that Π is hard even for polynomial size circuits. The next proposition follows ideas and notation from [RR97].

Proposition 19 (Conditional Hardness of Π)**.** *Suppose there exists a polynomial time one-way function $f\colon \{0,1\}^* \to \{0,1\}^*$ secure against circuits of size 2^{n^ε}, where $\varepsilon > 0$. Then, if $k \in \mathbb{N}$ is large enough, $\Pi[n^k, 1/3] = (1, 2/3)$-$\mathsf{MCSP}[n^k]$ cannot be computed by circuits of size $N^{O(1)}$.*

Proof Sketch. We use an idea from [RR97]. Under the existence of exponentially hard one-way functions, there exists a family $\{F_w\}_{w \in \{0,1\}^{\mathsf{poly}(n)}}$ of pseudorandom functions with $F_w\colon \{0,1\}^n \to \{0,1\}$ computable by circuits of size n^c that is secure against non-uniform oracle circuits of size 2^{n^γ}, for some $c \in \mathbb{N}$ and $\gamma > 0$. However, if $\Pi[n^k, 1/3]$ can be computed by circuits of size $N^{O(1)}$ for every $k \in \mathbb{N}$, it is possible to distinguish $\{F_w\}$ from random functions, as described below.

Let $\{D_{N,k}\}_N$ be a sequence of circuits $D_{N,k}\colon \{0,1\}^N \to \{0,1\}$ of size $\leq N^\ell$ computing $\Pi[n^k, 1/3]$, where $N(n) = 2^n$. We construct an oracle distinguisher against $\{F_w\}$. Let $m = n^{\gamma/2}$, and consider $D_{M,k}$ for $M = 2^m$ and a large enough k. We view $D_{M,k}$ as an oracle circuit $\widetilde{D}_{M,k}$ for functions in $\{0,1\}^n \to \{0,1\}$ by replacing its M-bit input string by oracle calls corresponding to the oracle input locations $x1^{n-n^{\gamma/2}}$, where $x \in \{0,1\}^{n^{\gamma/2}}$. Observe that, for large n, $\widetilde{D}_{M,k}$ has size at most 2^{n^γ}, since $D_{M,k}$ has size $M^\ell \leq 2^{\ell \cdot n^{\gamma/2}}$.

We claim that $\widetilde{D}_{M,k}$ distinguishes $\{F_w\}$ from random with constant probability. First, note that if $g\colon \{0,1\}^n \to \{0,1\}$ is a random function, then $g'\colon \{0,1\}^{n^{\gamma/2}} \to \{0,1\}$ given by $g'(x) = g(x1^{n-n^{\gamma/2}})$ is also a random function. Since a random function cannot be non-trivially approximated by polynomial size circuits, it follows that $g' \in \Pi^{\mathsf{no}}_M[m^k, 1/3]$. In particular, $\widetilde{D}_{M,k}$ outputs 0 on g with probability $\geq 1 - o(1)$, given that this oracle circuit was constructed from a circuit $D_{M,k}$ that computes $\Pi[m^k, 1/3]$ over M-bit inputs. On the other hand, for an oracle function coming from $\{F_w\}$, observe that the restriction of any function $F_w\colon \{0,1\}^n \to \{0,1\}$ to $F'_w(x) = F_w(x1^{n-n^{\gamma/2}})$ is a function over m input variables and of circuit complexity at most $m^{O(c/\gamma)}$. Thus, for a large enough k, $F'_w \in \Pi^{\mathsf{yes}}_M[m^k, 1/3]$. In turn, we get that $\widetilde{D}_{M,k}$ outputs 1 on F_w with probability 1. This contradicts the security of $\{F_w\}$, and completes the proof. □

Is $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ harder than natural properties? While Proposition 19 provides evidence of the hardness of $(1, 1-\delta)$-$\mathsf{MCSP}[s]$ via natural properties, we give in Section B an example of a consequence of the easiness of $(1, 1-\delta)$-MCSP-$\mathfrak{C}[s]$ that does not seem to follow from the existence of a natural property against $\mathfrak{C}$-circuits of size s.

We refer to Section C for some lower bounds related to the magnification results discussed in this section.

3.2 Non-uniform circuit lower bounds

In this section, we describe results showing that slightly better than super-linear circuit lower bounds imply separations such as $\mathsf{P} \neq \mathsf{NP}$. Indeed, substantially stronger separations can be obtained by improving circuit lower bounds from $n \cdot (\log n)^k$ to $n^{1+\varepsilon}$, where n is the number of input bits.

In the case of boolean circuits, we can employ a two-sided variant of the Approximate MCSP problem (see Appendix A).[6] The main advantage is that its hardness might be easier to establish. For simplicity, we state the result for a particular choice of parameters.

Theorem 20 (Magnification of Weak Circuit Lower Bounds in NP)**.** *Let $s\colon \mathbb{N} \to \mathbb{N}$, where $s(n) \geq n$, and let $1/2 < \beta < \alpha \leq 1$ be arbitrary constants. Further, let $\Pi[s, \alpha, \beta] = (\alpha, \beta)$-$\mathsf{MCSP}[s]$. Then,*

(i) Suppose there exists sub-exponentially secure one-way functions. Then there is $k^\star \geq 1$ such that for $s(n) = n^{k^\star}$, Π on N-bit inputs requires circuits of size $N^{\omega(1)}$. Unconditionally, Π requires circuits of size $\Omega(N)$.

(ii) Let $s(n) = n^{k^\star}$. If Π on inputs of length N cannot be computed by circuits of size $N \cdot (\log N)^{O(1)}$, then $\mathsf{NP} \not\subseteq \mathsf{SIZE}[\mathsf{poly}(n)]$.

(iii) Similarly, if Π is not computable by circuits of size $N \cdot 2^{(\log N)^{o(1)}}$, then $\mathsf{NP} \not\subseteq \mathsf{SIZE}[2^{n^{o(1)}}]$.

Proof Sketch. The argument in (i) showing that Π requires super-polynomial size circuits under a cryptographic assumption is similar to the proof of Proposition 19. On the other hand, in order to show that Π unconditionally requires circuits of size $\Omega(N)$, it is enough to use an adversarial argument. Indeed, a circuit of size s depends on at most $2s$ input variables. One can then fix these inputs and apply the idea from the proof of Lemma 51.

For the proof of (ii), the strategy is analogous to the proof of Lemma 16. A crucial difference is that we reduce $\Pi[s, \alpha, \beta]$ to Approximate Succinct MCSP (Appendix A). In terms of parameters, we consider the gap $\gamma = \alpha - \beta$, and use (α', β')-$\mathsf{Succinct\text{-}MCSP}[t]$, where $t = \widetilde{\Theta}(n^{k^\star})$, $\alpha' = \beta + \gamma/2 + \gamma/10$, and $\beta' = \beta + \gamma/2 - \gamma/10$. The analysis of the correctness of the reduction in Lemma 16 is replaced here by a similar analysis that relies on a concentration bound (Lemma 13).[7] Finally, for the derandomization step, we use a threshold gate instead of a conjunction. The crucial point is that circuits, as opposed to formulas, can compute majority gates over M input variables using $O(M)$ gates (cf. [Weg87]).

Finally, the proof of (iii) is entirely analogous, and the only difference is over the choice of parameters. □

Note that a generalization of Theorem 20 in the sense of Lemma 16 also holds. Consequently, understanding the circuit complexity of (α, β)-MCSP around the almost-linear-size complexity regime would have major consequences for complexity theory.

Our next results show that weak circuit lower bounds for the problem of approximating the Kt complexity of a string to within a small additive term imply super-polynomial circuit size lower bounds for EXP. While these results are for a larger class, they have the advantage they hold for the problem of approximating the Kt complexity of a string to within a small additive term. This is

[6] In the case of formulas, we were not able to use this formulation because it is unclear to us whether an approximate majority function on n input bits can be computed by formulas of size $\widetilde{O}(n)$.

[7] For more details, see the proof of Theorem 31 in Section 3.4.

a problem about the *worst-case* complexity of a string rather than its average-case complexity, and moreover the problem we consider is known [ABK+06] to be hard for EXP under polynomial-size reductions.

We will use Spielman's construction [Spi96] of error-correcting codes of constant rate and distance that are very efficiently encodable and decodable.[8]

Theorem 21 ([Spi96])**.** *There is a family of error correcting codes of constant rate and relative distance that can be encoded, decoded and inverted by algorithms running in quasi-linear time. Moreover, encoding can be done by circuits of linear size.*

Now we are ready to state and prove our theorem.

Theorem 22 (Magnification of Weak Circuit Lower Bounds in EXP)**.** *If there is a fixed $\delta > 0$ such that for each $\epsilon > 0$, $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ does not have circuits of size $N^{1+\delta}$, then $\mathsf{EXP} \not\subseteq \mathsf{SIZE}(\mathrm{poly})$.*

Proof. We show the contrapositive. Assume $\mathsf{EXP} \subseteq \mathsf{SIZE}(\mathrm{poly})$. We show that for all $\delta > 0$, there is $\epsilon > 0$ such that $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ has circuits of size $N^{1+\delta}$.

Let E be the linear-time encodable and decodable error correcting code given by Theorem 21. Let $C > 1$ and $\gamma > 0$ be constants such that E maps N bits to CN bits and has relative distance 2γ.

We show how to construct circuits of size $N^{1+\delta}$ for the problem $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$, for any ϵ small enough.

Given an input w for the problem $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$, we first apply the error-correcting code E to get $z = E(w)$. Note that this can be implemented by a circuit of linear size, by the assumption on complexity of the encoder.

We claim that for large enough N, if $Kt(w) \leq N^\epsilon$, then $Kt(z) \leq N^\epsilon + 2\log(N)$, and if $Kt(w) > N^\epsilon + 5\log(N)$, then $Kt(z') > N^\epsilon + 3\log(N)$ for any z' such that $|z'| = |z|$ and $\Delta(z, z') < \gamma$, where $\Delta(z, z')$ is the relative Hamming distance between z and z'.

To see the first part of the claim, let p be a program and t a time bound such that $U(p) = w$ in at most t steps, and such that $|p| + \log(t) \leq N^\epsilon$. Consider the program p' that first runs p and then runs the quasilinear-time algorithm for E on the output of p. We have that $|p'| = |p| + O(1)$, and $U(p') = z$ in at most t' steps, where $t' = t + O(N\,\mathrm{polylog}(N))$. Hence we have $\log(t') = \log(t) + (1 + o(1))\log(N)$, and hence $Kt(z) \leq |p'| + \log(t') = |p| + \log(t) + (1 + o(1))\log(N) \leq Kt(w) + 2\log(N)$ for N large enough.

To see the second part of the claim, suppose to the contrary that there is z' such that $Kt(z') \leq N^\epsilon + 3\log(N)$, and moreover $\Delta(z, z') < \gamma$. Let p be a program and t be a time bound such that $U(p) = z'$ in at most t steps, and such that $|p| + \log(t) \leq N^\epsilon + 3\log(N)$. We define a new program p' and time bound t' such that $U(p') = w$ in at most t' steps and $|p'| + \log(t') \leq N^\epsilon + 5\log(N)$ for large enough N, which yields a contradiction to the assumption on Kt complexity of w. The program p' first runs the program p to generate z', and then runs the quasilinear-time decoding procedure to generate the nearest codeword, which is z since $\Delta(z, z') < \gamma$. It then runs the quasilinear-time

[8] Technically, Spielman's result requires a polynomial-time pre-processing computation that produces the nearly-linear time encoding and decoding routines. This does not affect the size of the (non-uniform) circuit used for encoding. In any case, the use of uniform polynomial-time routines for encoding and decoding in the proof of Theorem 22 only affects the constant $C = 5$ appearing in the statement, and we observe that more recent (folklore) constructions can be used that do not require such polynomial time pre-processing (Venkatesan Guruswami, private communication).

inversion procedure on z to produce w. We have that $|p'| = |p| + O(1)$, and moreover the time taken by $U(p')$ is at most $t + O(N \operatorname{polylog}(N))$. Hence, using the same calculation as in the previous para, we have that $Kt(w) \leq Kt(z') + 2\log(N)$. which implies $Kt(w) \leq N^\epsilon + 5\log(N)$ – a contradiction.

Define the auxiliary language L as follows. An input $y = < a, 1^b, (i_1, b_1) \ldots (i_r, b_r) >$ belongs to L (where a and b are positive integers) if each $i_j, 1 \leq j \leq r$ is of length $\lceil \log(a) \rceil$, and if there is a string z of length a such that $Kt(z) \leq b$ and for each $j, 1 \leq j \leq r$, $z_{i_j} = b_j$. L is decidable in exponential time as we can exhaustively search all strings of length a with Kt complexity at most b and check if there is one which has the specified values at the corresponding bit positions in time $\mathsf{poly}(2^b, a, r)$. Since $\mathsf{EXP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ by assumption, L has polynomial-size circuits. Assume wlog that L has circuits of size $O(n^k)$ for some constant k, where n is the total input length.

We now give a low-complexity reduction from $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ to L by computing z from w and then applying the following sampling procedure. We sample uniformly and independently $r = N^{2\epsilon}$ indices $i_1 \ldots i_r \in [CN]$. We form the string $y = < CN, 1^{N^\epsilon + 2\log(N)}, (i_1, z_{i_1}) \ldots (i_r, z_{i_r}) >$. We claim that with high probability over the randomness of the reduction procedure, if w is a YES instance of $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$, then $y \in L$, and if w is a NO instance, then $y \notin L$.

The claim about YES instances can be seen to hold with probability 1 as follows. If w is a YES instance, we have that $Kt(z) \leq N^\epsilon + 2\log(N)$. In this case, z is a string of length CN with $Kt(z) \leq N^\epsilon + 2\log(N)$ such that z has the specified values at the specified bit positions, regardless of the positions that are sampled by the reduction.

For the claim about NO instances, as previously established, we have that $Kt(z') > N^\epsilon + 3\log(N)$ for any z' such that $|z'| = |z|$ and $\Delta(z, z') < \gamma$. Now consider any string z'' of length CN such that $Kt(z'') \leq N^\epsilon + 2\log(N)$. For such a string z'', for each $j, 1 \leq j \leq r$, the probability that $z''_{i_j} = z_{i_j}$ is at most $1 - \gamma$. Hence the probability that z'' agrees with z at all the specified bit positions is at most $(1-\gamma)^r \leq 2^{-\Omega(2N^\epsilon)}$. By a union bound over z'' with $Kt(z'') \leq N^\epsilon + 2\log(N)$, the probability that there exists a string z'' with Kt complexity at most $N^\epsilon + 2\log(N)$ which is consistent with the values at the specified bit positions is exponentially small in N^ϵ. Hence with high probability, y is a NO instance.

We construct small circuits for $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ as follows. Our construction uses randomness but we then show how to eliminate the randomness. We first apply the error-correcting code E to the input w to obtain a string z – this part of the computation can be done in quasi-linear size using the guarantee from Theorem 21. We then compute the instance y for the auxiliary language L by random sampling and simulate the $O(n^k)$ size circuit for L on y, accepting iff the circuit accepts. The random sampling step can be implemented in quasi-linear size, and the input length n of y is $O(N^{2\epsilon} \operatorname{polylog}(N))$, and hence for $\epsilon < \delta/3k$, the total size of the randomised circuit for $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ is at most $N^{1+\delta}$ for N large enough.

A naive derandomisation of these randomised circuits using Adleman's trick would incur an additional factor of N in the circuit size, but we can do better by taking advantage of the structure of our reduction. The trick is to use carefully chosen *random* circuits (where the randomness is in the structure of the circuit rather than in auxiliary inputs to it) rather than randomised circuits. Note that the random sampling step can be implemented by a random circuit which projects r randomly chosen co-ordinates of z together with their associated indices – this can be done in size $O(N^{2\epsilon} \cdot \operatorname{polylog}(N))$. Now consider such a random projection done $10N$ times independently to produce instances $y_1, y_2 \ldots y_{10N}$ of L. The circuit for L is simulated on each such input and the AND of all these results is output as the final answer. This random circuit has size at most $N^{1+\delta}$ for ϵ chosen small enough, and the error on any instance is less than 2^{-N}, hence by a union bound,

there exists a way of doing these $10N$ projections that yields a deterministic circuit solving the problem correctly on all instances, and having the same size.

□

Theorem 22 shows that even barely super-linear circuit lower bounds for the problem about Kt complexity we consider would have significant implications, i.e., a super-polynomial circuit lower bound for EXP. On the other hand, under the standard hardness assumption that EXP does not have polynomial size circuits, our problem does not have polynomial size circuits either. This follows from the hardness of the problem under polynomial-size reductions for each $\epsilon > 0$ [ABK+06]. We get the following equivalence as a consequence.

Corollary 23. $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ *has polynomial-size circuits for each* $\epsilon > 0$ *iff for all* $\delta > 0$ *there is an* $\epsilon > 0$ *such that* $\mathsf{MKtP}[N^\epsilon, N^\epsilon + 5\log(N)]$ *has circuits of size* $N^{1+\delta}$.

Theorem 22 magnifies a weak circuit lower bound to a strong one. We can additionally show that, in some cases, a *weak slightly non-uniform* lower bound can be magnified to a *strong non-uniform* lower bound, i.e., hardness magnification is both with respect to the size of the bound and the uniformity of the algorithm. We choose a variant of the MKtP problem with a slightly different parameter setting to emphasize the magnification with respect to uniformity.

Theorem 24. *If for each* $d > 0$, $\mathsf{MKtP}[\log(N)^d, \log(N)^d + 5\log(N)]$ *is not in* $\mathsf{BPTIME}(N \cdot \mathrm{polylog}(N))/\mathrm{polylog}(N)$, *then* $\mathsf{EXP} \not\subseteq \mathsf{SIZE}(\mathrm{poly})$.

Proof Sketch. We implement the same proof strategy as in the proof of Theorem 22, except for the following differences. Our Kt complexity parameter is smaller and hence we compress to a much smaller instance y of the auxiliary language L, of size $\mathrm{polylog}(N)$. Rather than solving y using a circuit, we supply a poly-size circuit y for inputs of the appropriate polylogarithmic input length as advice to the probabilistic quasi-linear time procedure which does the compression. The advice is used to solve the compressed instance.

Using our assumption that $\mathsf{EXP} \subseteq \mathsf{SIZE}(\mathrm{poly})$, poly-size circuits exist for the auxiliary language L, and hence there is a correct advice string for the probabilistic quasi-linear algorithm sketched above. □

We note that using a scaled-down version of the EXP-hardness result [ABK+06] for MKtP, if EXP does not have sub-exponential size circuits, then $\mathsf{MKtP}[\log(N)^d, \log(N)^d + 5\log(N)]$ does not have polynomial-size circuits for large enough d. The assumption which we magnify in Theorem 24 is much weaker, in that it only assumes hardness for slightly non-uniform probabilistic quasi-linear time algorithms with bounded error.

3.3 Magnification from sub-linear average-case lower bounds

In this section we will prove Theorem 5 and other related statements. Indeed, Theorem 5 is just a particular instantiation of the results presented in this section. We start with the following observations.

Proposition 25. *For every choice of the parameter functions* s *and* t, $\mathsf{Succinct\text{-}MCSP}\text{-}\mathfrak{C}[s,t] \in \mathsf{NP}$.

Proof. First, assume that $1^{s(n)}$ is also part of the input for this problem. Then a circuit of size $\leq s$ can be guessed and verified in time $\mathsf{poly}(n, s(n), t(n))$, which is polynomial in the total input length. However, in the definition of $\mathsf{Succinct\text{-}MCSP}[s,t]$, the parameters s and t are fixed and not part of the input. If $s(n) \gg t(n) \cdot n$, then by a brute-force construction there is always a circuit of size at most $O(t(n) \cdot n)$ that correctly labels the input pairs. Therefore, the only non-trivial case is when $s(n) = O(t(n) \cdot n)$. But in this case a circuit can be guessed and verified in nondeterministic polynomial time. It follows that $\mathsf{Succinct\text{-}MCSP}\text{-}\mathfrak{C}[s,t] \in \mathsf{NP}$ for every choice of s and t. □

Lemma 26. *Let $n \leq s(n) \leq 2^{o(n)}$ and $t(n) = 4s(n)\log s(n)$. Let $\boldsymbol{I}$ be a random collection of $t(n)$ pairs of the form $(x_i, \boldsymbol{b}_i)$, where each $\boldsymbol{b}_i \sim \{0,1\}$, and all x_i are fixed but distinct elements of $\{0,1\}^n$. Then, for every n sufficiently large,*

$$\Pr_{\boldsymbol{I}}[\textit{ There is a circuit of size } \leq s(n) \textit{ consistent with } \boldsymbol{I}\,] \;<\; 2^{-n}.$$

Proof. Using that $s(n) \geq n$, there are at most $2^{3s(n)\log s(n)}$ circuits of size at most $s(n)$ over n input variables (Proposition 15). For each circuit D of this form, the probability that D agrees with every pair in $\boldsymbol{I}$ is precisely $2^{-t(n)} = 2^{-4s(n)\log s(n)}$. Consequently, by a union bound and using that $s(n) \geq n$, the aforementioned probability is at most $2^{3s(n)\log s(n))-t(n)} < 2^{-n}$. □

Recall the definition of solving MCSP on average informally discussed in Section 1.2, and formally defined in Section A.

Theorem 27 (Magnification from sub-linear average-case lower bounds for MCSP)**.** *Let $\mathfrak{C}$ be a typical circuit class, and consider $\mathsf{MCSP}[s]$ for a size parameter $n \leq s(n) \leq 2^{o(n)}$. Assume that there exists $\delta > 0$ such that solving $\mathsf{MCSP}[s]$ on average on N-bit inputs requires $\mathfrak{C}$-circuits of size $\Omega(N^{\delta})$. Then there is a language L in NP such that L on inputs of length $m = \Theta(n \cdot s(n) \cdot \log s(n))$ requires $\mathfrak{C}$-circuits of size $\ell(m) = 2^{\Omega(n)}$.*

Proof. We take L as $\mathsf{Succinct\text{-}MCSP}[s,t]$, where $t = B \cdot s \log s$ and B is a sufficiently large constant. By Proposition 25, L is in NP. Suppose towards a contradiction that L on inputs of total length m (as in the statement) can be computed by $\mathfrak{C}$-circuits of size $2^{o(n)} = N^{o(1)}$. Let $\{D_m\}_m$ be a sequence of such circuits. We argue below that it is possible to use these circuits to solve $\mathsf{MCSP}[s]$ on average.

In order to achieve that, we describe a family $\{C_n\}_n$ of $\mathfrak{C}$-circuits that compute as follows. Let $N = 2^n$ be the input length for an instance of $\mathsf{MCSP}[s]$. Given $y \in \{0,1\}^{2^n}$, C_n on input y produces a string that encodes n and a list of pairs (w_i, b_i), where $i \in [t(n)]$, w_i is the i-th n-bit string in lexicographic order, and b_i is the corresponding entry in the truth table y. C_n then feeds these inputs to the corresponding circuit D_m, and outputs 0 if D_m outputs 0, and outputs "?" otherwise. This completes the description of each circuit C_n.

First, we claim that C_n never makes a mistake. Indeed, C_n always outputs a value in $\{0, \text{"?"}\}$, and if C_n outputs 0, by construction it follows that a projection of the input truth table does not admit circuits of size s. Obviously, in this case the truth table y does not admit circuits of size s.

Finally, we argue that C_n outputs "?" with a very small probability. Indeed, since $t(n) \geq 4s(n)\log s(n)$ and $n \leq s(n) \leq 2^{o(n)}$, over a random input $y \sim \{0,1\}^{2^n}$ Lemma 26 shows that most collections $\{(w_i, b_i) \mid i \in [t(n)]\}$ generated by C_n during the reduction will not be consistent with any circuit of size at most $s(n)$. By construction, C_n outputs 0 on such input truth tables. This completes the proof of Theorem 27. □

We can derive an immediate consequence from this result.

Corollary 28. *If there exists $c \geq 1$ such that $\mathsf{MCSP}[n^c]$ is hard on average for circuits of size N^γ, where $\gamma > 0$, then $\mathsf{NP} \not\subseteq \mathsf{SIZE}[2^{n^{o(1)}}]$.*

The next result shows that even sub-linear AC^0 lower bounds for a certain range of the size parameter s would have major consequences in complexity theory.

Corollary 29. *Assume there is $c \geq 1$ and $\gamma > 0$ such that $\mathsf{MCSP}[n^c]$ is not computable by $\mathsf{AC}^0[N^\gamma]$ circuits on average. Then $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.*

Proof. It is enough to observe that the reduction employed in the proof of Theorem 27 can be easily computed in AC^0, and that by Lemma 14, if $\mathsf{NP} \subseteq \mathsf{NC}^1$ then for every language $L \in \mathsf{NP}$ and $\varepsilon > 0$ there exists $d \in \mathbb{N}$ such that L can be computed by depth-d circuits of size at most $O(2^{n^\varepsilon})$. □

These results further motivate the investigation of the complexity of MCSP in very weak computational models. For instance, for formula size and under a different regime of parameters, the same technique yields the following connection.

Corollary 30. *Let $s(n) = 2^{n^\gamma}$, where $0 < \gamma < 1$. If $\mathsf{MCSP}[s(n)]$ on N-bit inputs cannot be computed on average by formulas of size $N^{o(1)}$, then there is a language $L \in \mathsf{NP}$ that over m-bit inputs requires formulas of size $2^{\Omega((\log m)^{1/\gamma})}$ for infinitely many values of m.*

Proof. The result is also immediate from Theorem 27 using that $m = \Theta(2^{n^\gamma} \cdot n^{1+\gamma})$ for the choice of $s(n)$, which gives $n = \Theta((\log m)^{1/\gamma})$. □

3.4 Magnification from sub-linear randomized lower bounds

We fix a uniform randomized model of computation which allows random-access to the input string. In other words, in one time unit it is possible to retrieve the bit stored in the i-th input cell, assuming the corresponding address is written on an appropriate index tape or register. The particular details of the model are not important for our result, and any computational model that allows random-access to the input and probabilistic computations is sufficient.[9] For concreteness, we say that a *randomized algorithm* $\mathbf{A}$ computes a function f if on every input x, $\Pr[\mathbf{A}(x) = f(x)] \geq 2/3$.

Theorem 31 (Magnification of sub-linear randomized lower bounds)**.** *Let $1/2 < \beta < \alpha \leq 1$ be constants, $c \in \mathbb{N}$, and $s(n) \leq 2^{n^\gamma}$, where $\gamma < 1$. If there is $\varepsilon > 0$ such that (α, β)-$\mathsf{MCSP}[s]$ on N-bit inputs cannot be computed by a (two-sided error) randomized algorithm running in time $2^{(\log N)^{1-\varepsilon}}$, then $\mathsf{NP} \not\subseteq \mathsf{BPP}$.*

Proof Sketch. We establish the contrapositive. Let $f \in \{0,1\}^N$ be a valid input to (α, β)-$\mathsf{MCSP}[s]$. We view this N-bit input string as the function $f\colon \{0,1\}^n \to \{0,1\}$, where $N = 2^n$. Our randomized algorithm $\boldsymbol{A}$ projects f at $t = \Theta_{\alpha,\beta}(s \log s)$ random input locations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t \in \{0,1\}^n$, producing an instance of (α', β')-$\mathsf{Succinct\text{-}MCSP}[s]$, where $\alpha' = \beta + \gamma/2 + \gamma/10$, $\beta' = \beta + \gamma/2 - \gamma/10$, and $\gamma = \alpha - \beta$. If $\mathsf{NP} \subseteq \mathsf{BPP}$, there is a randomized algorithm $\boldsymbol{B}$ that runs in polynomial time and correctly solves (α', β')-$\mathsf{Succinct\text{-}MCSP}[s]$. Since $t \leq 2^{(\log N)^{1-\Omega(1)}}$ due to the upper bound on $s(n)$

[9] For definiteness, assume that the random string is stored on a (read-once) one-way tape.

and $N = 2^n$, $\boldsymbol{B}$ runs in time $\leq 2^{(\log N)^{1-\Omega(1)}}$. $\boldsymbol{A}$ simply outputs the answer of $\boldsymbol{B}$ on the instance produced by the random projection.

Clearly, $\boldsymbol{A}$ runs in randomized time upper bounded by $\leq 2^{(\log N)^{1-\Omega(1)}}$, since we assume a computational model with random-access to the input string. If there is a circuit C of size s that agrees with f on a α fraction of the inputs, then by a concentration bound (Lemma 13) a projection of f over t random inputs admits a circuit (i.e. C) of size s that agrees with f on at least a $\alpha' < \alpha$ fraction of the samples. Therefore, with high probability $\boldsymbol{A}$ is correct on f, since $\boldsymbol{B}$ is a randomized algorithm that correctly computes (α', β')-Succinct-MCSP$[s]$. On the other hand, assume that no circuit C of size s agrees with f on more than a β fraction of the input strings. Then, for any fixed circuit D of size s, the probability over the random projection that D agrees with more than a $\beta' > \beta$ fraction of the t samples is $\leq \exp(-\Theta(t))$. By a union bound over the number of size-s circuits (Proposition 15) and by our choice of t as a function of s, α, and β, it follows that with high probability over the random projection, no such circuit agrees with more than a β' fraction of the samples. Using once again the correctness of $\boldsymbol{B}$, it follows that $\boldsymbol{A}$ is correct on f. □

4 Magnification for NP-complete Problems

4.1 Magnification for Vertex Cover

In this Section, we explore a connection between hardness magnification and kernelization, a well-know technique from parameterized complexity (see e.g. [DF13]). We restrict our results to one representative example.

Definition 32 (The Vertex Cover Problem)**.** *In the* Vertex-Cover *problem, the input is a pair* (G, w)*, where* $G \in \{0,1\}^{\binom{n}{2}}$ *encodes the adjacency matrix of an undirected* n*-vertex graph* $G = (V, E)$*, and* $w \in \{0,1\}^{\log n}$ *encodes in binary an integer parameter* $k \geq 0$*. The pair* (G, w) *is a positive instance of* Vertex-Cover *if there exists* $S \subseteq V$*,* $|S| \leq k$*, such that every* $e = \{u, v\} \in E$ *intersects* S*. Similarly, for* $k = k(n)$ *we let* k-Vertex-Cover *be the same computational problem with the exception that the parameter* k *is fixed in advance and is not part of the input.*

We use $m = \Theta(n^2)$ to denote the total input length of an instance of k-Vertex-Cover or Vertex-Cover on n-vertex graphs.

We briefly recall some known results about these problems. It is known that k-Vertex-Cover can be solved in time $c^k \cdot \mathsf{poly}(m)$, where $c < 2$ (see e.g. [CKX10]). Moreover, under ETH this problem cannot be solved in time $2^{o(k)} \cdot \mathsf{poly}(m)$ ([IPZ01]; see e.g. [DF13, Theorem 29.5.9]). Therefore, it is plausible that k-Vertex-Cover is not in P if $k = \omega(\log n)$. On the other hand, using the NP-completeness of Vertex-Cover and a simple translation argument, if $k = (\log n)^C$ for a large constant $C \geq 1$, then k-Vertex-Cover is not in P unless CNF-SAT $\in$ DTIME$[2^{n^\varepsilon}]$, where $\varepsilon = \varepsilon(C)$.

Theorem 33 (Hardness Magnification via Kernelization)**.** *Let* $k(n) = (\log n)^C$*, where* $C \in \mathbb{N}$ *is arbitrary. If for every* $d \geq 1$ *there exists* $\varepsilon > 0$ *such that* k-Vertex-Cover $\notin \mathsf{AC}^0_d[m^{1+\varepsilon}]$*, then* NP $\not\subseteq$ NC^1*.*

In order to prove this result, we need the following construction. For a boolean string x, let $|x|_1$ be the number of 1s in x.

Lemma 34 (AC^0 circuits for small thresholds [HWWY94, Theorem 7])**.** *Let* $T^n_\ell \colon \{0,1\}^n \to \{0,1\}$ *be the function where* $T^n_\ell(x) = 1$ *if and only if* $|x|_1 \leq \ell$*. Let* $\ell \leq (\log n)^d$*, where* $d \in \mathbb{N}$*. Then* T^n_ℓ *can be computed by an* AC^0 *circuit of depth* $d + O(1)$ *that contains* $n^{o(1)}$ *gates.*

Proof of Theorem 33. First, we recall a standard kernelization/decision procedure for k-Vertex-Cover that takes an input G and either produces the correct answer, or outputs a pair (G', k'), where the number of vertices of the new graph G' is upper bounded by $2k^2$, $k' \leq k$, and such that G admits a cover set $S \subseteq V(G)$ of size k if and only if G' admits a cover set $S' \subseteq V(G')$ of size k'. Note that while the input graph G is an instance of k-Vertex-Cover, the output is either the correct answer or a pair of strings describing an instance of Vertex-Cover.

This procedure computes as follows. Let $T = \{v \in G \mid \mathsf{deg}_G(v) > k\}$. If $|T| > k$, reject. Otherwise, let $V^- = V \setminus T$, and $G^- = G[V^-]$, the subgraph of G induced by V^-. Now let G' be the graph obtained by deleting all isolated vertices in G^-, and let $k' = k - |T|$. If $|V(G')| > 2k^2$, reject. Otherwise, output (G', k'). This completes the description of the procedure.

It is not hard to prove that this procedure never rejects a graph G that admits a set cover of size k, and that for every input graph G that is not rejected, G admits a set cover of size k if and only if G' admits a set cover of size k'. Indeed, if a set cover for G of size k exists, then every vertex in T must be in the cover. On the other hand, since no vertex in G' is isolated, it must contain at least $V(G')/2$ edges. At the same time, each vertex of G' has degree at most k, which implies that a subset $S' \subseteq V(G')$ of size k' can cover at most $kk' \leq k^2$ edges. Consequently, if $|V(G')| > 2k^2$ the input graph G can be safely rejected. It follows that whenever the input instance G is not rejected by the procedure, it is correctly reduced to an instance of Vertex-Cover whose size (number of vertices) is at most $2k^2$ and whose associated parameter $k' \leq k$. (For related references and for a less concise exposition, we refer to any standard textbook in parameterized complexity.)

In order to prove the result, we will use Lemma 34 to implement this reduction via bounded-depth circuits of size $m^{1+o(1)}$. We then employ the assumption that $\mathsf{NP} \subseteq \mathsf{NC}^1$ and an additional trick (Lemma 14) to solve Vertex-Cover on graphs of size $\leq 2k^2$ in a very efficient way (as a function of m, the initial input size). Altogether, this will provide us with a bounded-depth circuit D of size $m^{1+o(1)}$ for k-Vertex-Cover, which completes the proof of the contrapositive formulation of the theorem. More details follow.

Our circuit D computes as follows. Let G be an input graph on n vertices represented by its adjacency matrix, and $k = (\log n)^C$ be fixed. Recall that the total input length is $m = \Theta(n^2)$.

1. First, D computes a vector $\alpha \in \{0,1\}^n$ where $\alpha_i = 1$ if and only if $\mathsf{deg}_G(v_i) > k$. If $|\alpha|_1 > k$, D outputs 0. (α corresponds to the set T in the discussion above.)

Note that each α_i can be computed in parallel by a constant-depth circuit of size $n^{o(1)}$ whose existence follows from Lemma 34. Similarly, $|\alpha|_1$ can be tested by such circuits.

2. D computes a vector $\beta \in \{0,1\}^n$ where $\beta_i = 1$ if and only if $\mathsf{deg}_{G^-}(v_i) = 0$, where $G^- = G[V^-]$ and $V^- = V \setminus T$. (β captures the set of isolated vertices in the discussion above.)

Clearly, this vector can be computed by a multi-output AC^0 circuit of size $O(m)$ using the input graph G and α.

3. Let $\gamma \in \{0,1\}^n$ be given by $\gamma_i = \neg(\alpha_i \vee \beta_i)$. (This vector encodes $V(G')$ in the discussion above.) D outputs 0 if $|\gamma|_1 > 2k^2$. Otherwise, D computes $k' = k - |\alpha|_1$, represented in binary.

Again, $|\gamma|_1$ can be tested in constant-depth using $n^{o(1)}$ gates via Lemma 34 thanks to the value of k. In order to compute k', observe that this is a string on $O(\log \log n)$ bits, since it takes an integer value in the interval $[0, (\log n)^C]$. Since k is fixed and the value of $|\alpha|_1$ is at most k by Item 1, D can guess and check all possible values of $|\alpha|_1$ in parallel (via Lemma 34), then produce a

correct representation of k'. Therefore, it is clear that this step can also be done in constant-depth and by a circuit size at most $O(m)$.

4. D uses γ and the input graph G to compute the adjacency matrix of G', a graph that contains exactly $2k^2$ vertices. Together with Item 4, this produces an output pair (G', k') (This is similar to the discussion above, except that for convenience we pad G' to a graph on $2k^2$ vertices.)

This can be done as follows. The extra vertices of G' will have degree zero. In order to compute $G'[i,j]$, where $i \neq j \in [1, 2k^2]$, D guesses in parallel for every $a, b \in [n]$ whether v_a is the i-th non-zero entry of γ and v_b is the j-th non-zero entry of γ. For every fixed choice of a and b, this is done by appropriate circuits derived from Lemma 34. If there is a match, D inspects the input graph G and outputs 1 for $G'[i,j]$ if and only if $\{v_a, v_b\} \in E(G)$. As a consequence, for every choice of i and j, the bit $G'[i,j]$ can be computed by a constant-depth circuit of size at most $n^2 \cdot n^{o(1)} \leq m^{2+o(1)}$. Since there are $2k^2 \leq (\log n)^{O(1)}$ such pairs, and each bit of G' can be computed in parallel, the entire computation can be done in constant-depth and using at most $m^{2+o(1)}$ gates.

5. Assume $\mathsf{NP} \subseteq \mathsf{NC}^1$, and let E_t be a circuit of depth $O(\log t)$ that solves Vertex-Cover over t-vertex graphs. D simulates E_ℓ in constant-depth and outputs $E_\ell(G', k')$, where $\ell = \Theta(k^2)$.

We claim that this simulation can be computed by a constant-depth circuit of size $O(m^2)$. Indeed, by Lemma 14, E_ℓ can be computed by depth-d circuits of size at most $\exp(\ell^{O(1/(d-1))})$. Since $\ell = O(k^2) = O((\log n)^{2C})$, by taking d to be a constant sufficiently larger than $2C$, we get that E_ℓ can be simulated in constant-depth by a circuit of size at most $O(n) \leq O(m^2)$.

Overall, it follows that if $\mathsf{NP} \subseteq \mathsf{NC}^1$ then k-Vertex-Cover can be computed by constant-depth circuits of size $m^{1+o(1)}$ whenever $k \leq (\log n)^{O(1)}$. This completes the proof of Theorem 33. □

For larger $k(n)$, we can establish a connection to time-space trade-off results. We assume a computational model where the algorithm has random-access to the input string. (For concreteness, one can use Turing Machines with a read-only input tape, a constant number of working tapes, and a special tape used to address the input string.) Recall that $\mathsf{DTISP}[t(n), s(n)]$ denotes the class of languages that can be decided by an algorithm that simultaneously run in time $O(t(n))$ and space $O(s(n))$.

Theorem 35 (Magnification from Time-Space Trade-offs)**.** *Let $k(n) = n^{o(1)}$. If there exists $\varepsilon > 0$ such that k-Vertex-Cover $\notin \mathsf{DTISP}[m^{1+\varepsilon}, m^{o(1)}]$, where the input is an n-vertex graph represented by an adjacency matrix of bit length $m = \Theta(n^2)$, then $\mathsf{P} \neq \mathsf{NP}$.*

Proof Sketch. Again, we establish the contrapositive. First, we verify that the reduction behind the proof of Theorem 33 can be implemented by an algorithm running in time $m^{1+\varepsilon}$ and space $m^{o(1)}$, for $k(n) = n^{o(1)}$. Then, we solve the resulting instance of Vertex-Cover of size $\ell = \mathsf{poly}(k)$ in polynomial time under the assumption that $\mathsf{NP} \subseteq \mathsf{P}$. Since an algorithm running in time polynomial in ℓ also uses space polynomial in ℓ, and $\ell = m^{o(1)}$, it follows that k-Vertex-Cover $\in \mathsf{DTISP}[m^{1+\varepsilon}, m^{o(1)}]$.

Recall that $m = \Theta(n^2)$. To show that the reduction can be computed in $\mathsf{DTISP}[m^{1+\varepsilon}, m^{o(1)}]$, we need to proceed in a slightly different way. In particular, the algorithm does not have enough space to store the vectors α, β, and γ. (However, it has enough space to store the adjacency matrix produced by the reduction.) We follow in part the notation of the proof of Theorem 33, highlighting the relevant modifications.

Instead of computing these vectors as in steps **1**, **2**, and **3**, the algorithm produces a list of $\mathsf{poly}(k)$ vertices encoding γ (the sparsity of γ is a function of k). This list is stored on a fixed working tape of the algorithm, and it is not hard to see that it can be computed within the complexity bounds using random-access to the input string. (As opposed to AC^0, we can count in logarithmic space and almost linear time.) From such a list it is possible to output on a different tape the adjacency matrix of the graph G' from step **4**. This completes the proof. □

4.2 Magnification for SAT

In this section, we show a magnification result for SAT. In contrast to other results, here we magnify from lower bounds against polynomial-time refuters, a notion defined by Kabanets [Kab01]. We assume a computational model with random access to the input string.

The proof relies on efficient versions of the PCP theorem where the reduction runs in quasi-linear time.

Theorem 36 ([BGH+06, BS08, Din07])**.** *For some constant $\epsilon > 0$, there is a function f computable in time $O(m \operatorname{polylog}(m))$ and a function g computable in polynomial time such that:*

1. *If ϕ is a satisfiable 3-CNF, $f(\phi)$ is a satisfiable 3-CNF.*
2. *If ϕ is an unsatisfiable 3-CNF, then at most a $(1-\epsilon)$ fraction of clauses of $f(\phi)$ are simultaneously satisfiable.*
3. *If ϕ is a satisfiable 3-CNF and w is a satisfying assignment to ϕ, then $g(\phi, w)$ is a satisfying assignment to $f(\phi)$.*

We also need the Easy Witness Lemma of [IKW02]. The lemma is usually stated in terms of witnesses of YES instances having small circuit complexity when interpreted as truth tables, but we find it more convenient to use an equivalent formulation in terms of KT complexity.

Lemma 37 ([IKW02])**.** *Let $L \in \mathsf{NEXP}$ be any language, and let R be a polyomial-time computable relation such that $x \in L$ iff there is a string $y, |y| = 2^{|x|^k}$ such that $R(x, y) = 1$, where k is a fixed constant. If $\mathsf{NEXP} \subseteq \mathsf{SIZE}(poly)$, then for each $x \in L$, there is $y, |y| = 2^{|x|^k}$ such that $R(x, y) = 1$ and $KT(y) = \operatorname{poly}(|x|)$.*

We are now ready to state and prove the theorem.

Theorem 38 (Magnification for SAT)**.** *Suppose that for every probabilistic machine M halting in time $m \operatorname{polylog}(m)$ and using $\operatorname{polylog}(m)$ many random bits, there is a deterministic polynomial-time machine N such that for infinitely many m, $N(1^m)$ outputs a 3-CNF formula $\phi_m, |\phi_m| = m$ for which either ϕ_m is satisfiable and $M(\phi_m)$ rejects with probability $> 1/m$ or ϕ_m is unsatisfiable and $M(\phi_m)$ accepts with probability $> 1/m$. Then $\mathsf{NEXP} \neq \mathsf{BPP}$.*

Proof. Assume, for the sake of contradiction, that $\mathsf{NEXP} = \mathsf{BPP}$. Let c be a constant such that $\mathsf{NTIME}(2^n) \subseteq \mathsf{BPTIME}(n^c)$ – the existence of such a constant c follows from $\mathsf{NEXP} = \mathsf{BPP}$ using the fact that there is a language complete for $\mathsf{NTIME}(2^n)$ under deterministic linear-time reductions.

We describe a probabilistic machine M that attempts to solve 3-SAT, and show that any deterministic polynomial-time refuter for M yields a contradiction.

Let ϕ be a 3-CNF formula with $|\phi| = m$ (we always use $|\phi|$ to denote the bitlength of ϕ in a standard representation) and q variables. Let $k = \log(m)^a$, where a is a constant to be determined later. M behaves as follows on ϕ. It applies the reduction f from Theorem 36 to ϕ to produce a 3-CNF formula ϕ'. Since f runs in time $O(m \operatorname{polylog}(m))$, we have that $|\phi'| = O(m \operatorname{polylog}(m))$. M then samples k^2 clauses from ϕ' independently and uniformly at random, and forms their conjunction ϕ''. Note that not all variables in ϕ' will appear in ϕ'' if k^2 is much smaller than q – the variables retain their original indices in ϕ''.

Define an auxiliary language L' as follows. Consider an input $w = (\psi, 1^\ell, q')$, where ψ is a 3-CNF formula such that all indices of variables appearing in ψ belong to $[q']$ and ℓ and q' are positive integers with $\ell \leq q' \leq 2^\ell$. This string belongs to L' if and only if there is a string $w' \in \{0,1\}^{q'}$ such that $Kt(w') \leq \ell$ and w' satisfies ψ when interpreted as a Boolean assignment to variables with indices in $[q']$. L' can be solved in deterministic time $2^{O(\ell)}|\psi| \operatorname{polylog}(|\psi|)$ simply by enumerating all strings of length q' with Kt complexity at most ℓ, which can be done in time $2^{O(\ell)}$, and checking for each one if it is a satisfying assignment of ψ. Each such check can be done in time $|\psi| \operatorname{polylog}(|\psi|)$. In particular, $L' \in \mathsf{DTIME}(2^{O(n)})$, where n is the total length of the input to L', and hence by assumption $L' \in \mathsf{BPTIME}(n^c)$. Let M' be a probabilistic machine deciding L' with error 2^{-n}.

M simulates M' on input $(\phi'', 1^{2k}, q')$, where q' is the number of variables in ϕ', accepting if M' accepts and rejecting if M' rejects. The total time taken by M is $O(m \operatorname{polylog}(m))$, since applying the reduction f takes time $m \operatorname{polylog}(m)$ and the simulation of the machine M' takes time $\operatorname{poly}(k) = \operatorname{polylog}(m)$. Thus M is a probabilistic machine running in time $m \operatorname{polylog}(m)$ that attempts to solve 3-SAT. In addition, it is easy to see that M employs at most poly-logarithmic many random bits, given our choice of k and the input length on which the machine M' is invoked. We would like to show that any deterministic polynomial-time refuter for M, i.e., any deterministic polynomial-time machine that infinitely often outputs an instance ϕ on which M does not give the correct answer with high probability, can be used to derive a contradiction.

Indeed, suppose there is such a polynomial-time machine N, that for infinitely many m on input 1^m, outputs a formula ϕ_m such that either ϕ_m is satisfiable and M rejects ϕ_m with probability $> 1/m$, or ϕ_m is unsatisfiable and M accepts ϕ_m with probability $> 1/m$.

Note that each formula ϕ_m output by N on input 1^m has Kt complexity $O(\log(m))$ (when interpreted as a string). The reason is that the code for the machine N together with the standard $\log(m)$-bit description of the number m can be used as a program p to generate ϕ_m; moreover the generation is accomplished in time $t = \operatorname{poly}(m)$. Recall that the Kt complexity of the string ϕ_m is the minimum over $|p| + \log(t)$ such that the universal machine U generates ϕ_m from p within t steps. In our case, we have $|p| = O(\log(m))$ and $t = \operatorname{poly}(m)$, hence the Kt complexity of ϕ_m is $O(\log(m))$.

We will argue that for any input $\phi_m, |\phi_m| = m$ such that $Kt(\phi_m) = O(\log(m))$, the machine M solves ϕ_m correctly if ϕ_m is unsatisfiable, and also solves ϕ_m correctly if ϕ_m is satisfiable and has a satisfying assignment w such that $Kt(w) \leq k$. We will then use these two facts to derive a contradiction to the existence of the refuter N.

We first argue that M solves ϕ_m correctly if ϕ_m is unsatisfiable, when $|\phi_m| = m$ and $Kt(\phi_m) = O(\log(m))$. Let ϕ' be the 3-CNF produced by M from applying the reduction f to ϕ_m and ϕ'' be the 3-CNF produced by then randomly sampling k^2 clauses and taking their conjunction. We argue that with overwhelmingly high probability, ϕ'' does not have a satisfying assignment w' such that $Kt(w') \leq 2k$.

Indeed, we have that for any fixed assignment y to the variables of ϕ', y satisfies at most a $(1-\epsilon)$

fraction of the clauses of ϕ', where ϵ is the constant given by Theorem 36. Thus the probability that y satisfies a randomly sampled clause is at most $(1-\epsilon)$. It follows that the probability that y satisfies all k^2 randomly sampled clauses occurring in ϕ'' is at most $(1-\epsilon)^{k^2} = 2^{-\Omega(k^2)}$, since the clauses are sampled independently and uniformly at random.

Now, by a union bound over all assignments w' such that $Kt(w') \leq 2k$, we have that the probability that there exists such a w' satisfying all clauses in ϕ'' is at most $2^{2k}2^{-\Omega(k^2)} = 2^{-\Omega(k^2)}$. Hence with probability at least $1 - 2^{-\Omega(k^2)}$ over the internal randomness of the machine M, the formula ϕ'' has no satisfying assignment w' such that $Kt(w') \leq 2k$. Note that for every such ϕ'', by correctness of the machine M', we have that M' rejects with probability at least $1-2^{-n}$, where n is the length of the input to M'. This probability is at least $1-2^{-k}$, hence by using Bayes' theorem, we have that with probability at least $1 - 2^{-\Omega(k)} > 1 - 1/m$ (for a chosen sufficiently large), M rejects as claimed.

We next argue that M solves ϕ_m correctly if ϕ_m is satisfiable and has a satisfying assignment w such that $Kt(w) \leq k$. Consider such a formula ϕ_m. The ordered pair $< \phi_m, w >$ has Kt complexity at most $k + O(\log(m))$ by sub-additivity of Kt complexity, which is at most $1.5k$ for large enough m when a is chosen to be greater than 1.

Now consider the formula ϕ' produced by applying the reduction f to ϕ_m. By Theorem 36, $g(\phi_m, w)$ is a satisfying assignment to ϕ'. Since (ϕ_m, w) has Kt complexity at most $1.5k$, and since g runs in poly(m) time, we have that the Kt complexity of $w' = g(\phi_m, w)$ is at most $2k$, for large enough m and when a is chosen to be greater than 1. Note that since w' satisfies ϕ', it also satisfies each sub-sampled formula ϕ''. Hence M' is always simulated on a positive instance of L', regardless of the internal randomness of L, and consequently M accepts with probability greater than $1-1/m$, using the correctness of M'.

Thus the only case where M does not solve ϕ_m correctly with error $< 1/m$ is when ϕ_m is satisfiable and does not have a satisying assignment of Kt complexity at most k. We will show that this case leads to a contradiction. We sketch the definition of a relation $R(u, y)$ as follows. The first input u is viewed as a pair (D, x), where D is a machine, and x is a number. We assume that the sizes of D and x are roughly $\log m$, where $\widetilde{\Theta}(m)$ is the total input length expected by R on a valid input pair (u, y). The relation R accepts its input pair (u, y) iff $D(1^x)$ when executed for $m^{O(\log m)}$ steps outputs a formula ϕ_x such that $\phi_x(y) = 1$. Note that R is an NEXP-verifier. Consequently, by Lemma 37 and using the assumption that $\mathsf{NEXP} = \mathsf{BPP} \subseteq \mathsf{SIZE}(\mathsf{poly})$, there is a constant b such that for every $u = (D, x)$, if $D(1^x)$ outputs a satisfiable formula ϕ_x during its computation then there is a satisfying assignment y for ϕ_x such that $KT(y) \leq \log(m)^b$. But now using Proposition 12 and choosing $a > b$, if fix the input D to be the code of the refuter, we have a contradiction to the assumption that there are infinitely many m for which ϕ_m is satisfiable and does not have a satisfying assignment of Kt complexity at most k. □

By adjusting parameters in the proof, we can get quasi-polynomial or even stronger lower bounds in the consequent, but there is an inherent tradeoff which prevents us from ruling out the possibility that NEXP is contained in probabilistic sub-exponential time.

5 Directions and open problems

We list below some questions and directions motivated by our results:

1. Formula lower bounds. There are several techniques known to yield super-linear size formula lower bounds (e.g. [Tal17], [DM16, GMWW17], [Juk12, Chapter 6], [Dun88, Chapter 4], [Weg87, Chapter 8]), and an obvious direction (see Theorem 17 (*iii*)) is to try to adapt them to $(1, 1-\delta)$-$\mathsf{MCSP}[s]$. A necessary step is to use the technique to prove lower bounds against the standard $\mathsf{MCSP}[s]$ problem, as done by [HS17]. Similarly, one can try to prove super-linear bounded-depth lower bounds for $(1, 1-\delta)$-$\mathsf{MCSP}[s]$, a direction motivated by Theorem 18.[10] While there are techniques tailored to such results (cf. Section C for three examples), it seems unlikely that they can be combined with magnification without substantial new ideas.

Perhaps a more reasonable goal is to prove super-linear lower bounds that, while not strong enough for hardness magnification, are at least compatible with the parameters $s(n)$ and $\delta(n)$ required for magnification. In particular, with respect to formulas and bounded-depth circuits, we pose the challenge of establishing a $\omega(N)$ lower bound. (Note that we have established even stronger bounds in Sections C.1 and C.2, but for values of $s(n)$ and $\delta(n)$ that do not seem to be enough for magnification.)

2. Barriers for magnification-based lower bounds? Another pressing question is whether there are barriers such as natural proofs [RR97] for this technique. A related discussion appears in [AK10, Section 8]. It seems that the magnification approach in its general form is not naturalizable.

3. Relations between MCSP and Approximate MCSP. Is there a reduction from MCSP to Approximate MCSP (see Section A for definitions) for an interesting regime of parameters? This is a relevant direction due to existing lower bounds for MCSP (cf. Section C.1).

4. The complexity of k-Vertex-Cover. Investigate the computational complexity of this problem in weak circuit models in the regime where $\log n \leq k \leq \mathsf{poly}(\log n)$, as suggested by Theorem 33. (Note that there has been progress in understanding the circuit complexity of other parameterized graph problems such as k-Clique [Ros10], when k is small.) Another related direction is to improve existing time-space trade-offs, in connection to Theorem 35.

5. Magnification of existing lower bounds? Our results follow a general strategy of reducing a problem of size n to a smaller instance of another explicit problem. The reduction is computed in a very efficient way, measured according to the computational model. Can we follow this strategy to magnify existing lower bounds in computational models such as formulas and bounded-depth circuits? Note that compression lower bounds in the sense of [CS12, OS15] can be interpreted as "barriers" to magnification. But in boolean devices such as formulas, monotone circuits, and circuits of linear size compression is less understood, and magnification of existing lower bounds might perhaps be feasible.

Acknowledgements

We are grateful to Jan Krajíček, Ján Pich and Ninad Rajgopal for helpful discussions and comments.

[10] Note however that the regime of s is different in the corresponding magnification results.

References

[ABK+06] Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006.

[AHM+08] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008.

[Ajt83] Miklós Ajtai. $\sum_1^1$-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.

[AK10] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010.

[All01] Eric Allender. When worlds collide: Derandomization, lower bounds, and Kolmogorov complexity. In *Foundations of Software Technology and Theoretical Computer Science* FSTTCS, pages 1–15, 2001.

[Ats06] Albert Atserias. Distinguishing SAT from polynomial-size circuits, through black-box queries. In *Conference on Computational Complexity* (CCC), pages 88–95, 2006.

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *Transactions on Computation Theory*, 1(1), 2009.

[BGH+06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput.*, 36(4):889–974, 2006.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. Relativizatons of the $\mathsf{P} =? \mathsf{NP}$ question. *SIAM J. Comput.*, 4(4):431–442, 1975.

[BR17] Andrej Bogdanov and Alon Rosen. Pseudorandom functions: Three decades later. In *Tutorials on the Foundations of Cryptography*, pages 79–158. 2017.

[BS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput.*, 38(2):551–607, 2008.

[BSSV03] Paul Beame, Michael E. Saks, Xiaodong Sun, and Erik Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003.

[CIKK16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Algorithms from natural lower bounds. *Electronic Colloquium on Computational Complexity* (ECCC), 23:8, 2016.

[CILM18] Marco L. Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness amplification for non-commutative arithmetic circuits. In *Computational Complexity Conference* (CCC), pages 12:1–12:16, 2018.

[CKX10] Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theor. Comput. Sci.*, 411(40-42):3736–3756, 2010.

[CR96] Shiva Chaudhuri and Jaikumar Radhakrishnan. Deterministic restrictions in circuit complexity. In *Symposium on Theory of Computing* (STOC), pages 30–36, 1996.

[CS12] Arkadev Chattopadhyay and Rahul Santhanam. Lower bounds on interactive compressibility by constant-depth circuits. In *Symposium on Foundations of Computer Science* (FOCS), pages 619–628, 2012.

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007.

[DM16] Irit Dinur and Or Meir. Toward the KRW composition conjecture: Cubic formula lower bounds via communication complexity. In *Conference on Computational Complexity* (CCC), pages 3:1–3:51, 2016.

[Dun88] Paul E. Dunne. *The Complexity of Boolean Networks*. Academic Press, 1988.

[FLvMV05] Lance Fortnow, Richard J. Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005.

[FSS84] Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

[GMWW17] Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: The composition of a function and a universal relation. *SIAM J. Comput.*, 46(1):114–131, 2017.

[GST07] Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. If NP languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007.

[Hås98] Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998.

[HS17] 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.

[HWWY94] Johan Håstad, Ingo Wegener, Norbert Wurm, and Sang-Zin Yi. Optimal depth, very small size circuits for symmetric functions in AC^0. *Inf. Comput.*, 108(2):200–211, 1994.

[HWY11] Pavel Hrubeš, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. *Journal of the American Mathematical Society*, 24(3):871–898, 2011.

[IKW02] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002.

[IM02] Kazuo Iwama and Hiroki Morizumi. An explicit lower bound of $5n - o(n)$ for Boolean circuits. In *Symposium on Mathematical Foundations of Computer Science* MFCS, pages 353–364, 2002.

[IMZ12] Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *Symposium on Foundations of Computer Science* (FOCS), pages 111–119, 2012.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[JŁR00] Svante Janson, Tomasz Łuczak, and Andrzej Ruciński. *Random graphs*. Wiley-Interscience, New York, 2000.

[Juk12] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*. Springer, 2012.

[Kab01] Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. *J. Comput. Syst. Sci.*, 63(2):236–252, 2001.

[KRT17] Ilan Komargodski, Ran Raz, and Avishay Tal. Improved average-case lower bounds for de Morgan formula size: Matching worst-case lower bound. *SIAM J. Comput.*, 46(1):37–57, 2017.

[KV94] M. Kearns and U. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994.

[Lev84] Leonid Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61:15–37, 1984.

[LW13] Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time, with applications. *Computational Complexity*, 22(2):311–343, 2013.

[MP17] Moritz Müller and Ján Pich. Feasibly constructive proofs of succinct weak circuit lower bounds. *Electronic Colloquium on Computational Complexity* (ECCC), 24:144, 2017.

[Nis91] Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[OS15] Igor C. Oliveira and Rahul Santhanam. Majority is incompressible by $\mathrm{AC}^0[p]$ circuits. In *Conference on Computational Complexity* (CCC), pages 124–157, 2015.

[Raz85] Alexander A. Razborov. Lower bounds on the monotone complexity of some Boolean functions. *Doklady Akademii Nauk SSSR*, 281:798–801, 1985. English translation in: Soviet Mathematics Doklady 31:354–357, 1985.

[Ros10] Benjamin Rossman. *Average-Case Complexity of Detecting Cliques*. PhD thesis, Massachusetts Institute of Technology, 2010.

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

[Spi96] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Information Theory*, 42(6):1723–1731, 1996.

[Sri03] Aravind Srinivasan. On the approximability of clique and related maximization problems. *J. Comput. Syst. Sci.*, 67(3):633–651, 2003.

[Tal17] Avishay Tal. Formula lower bounds via the quantum method. In *Symposium on Theory of Computing* (STOC), pages 1256–1268, 2017.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

[Weg87] Ingo Wegener. *The complexity of Boolean functions*. Wiley, 1987.

[Wil07] Ryan Williams. *Algorithms and Resource Requirements for Fundamental Problems*. PhD thesis, Carnegie Mellon University, 2007.

[Wil13] Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013.

[Wil14] Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014.

[Wil16] Ryan Williams. Natural proofs versus derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016.

[Wil18] Ryan Williams. Some estimated likelihoods for computational complexity. 2018.

A Variants of the Minimum Circuit Size Problem

We consider the following versions of the Minimum Circuit Size Problem (MCSP).

Definition 39 (Parameterized MCSP). *Let $\mathfrak{C}$ be a circuit class, and $s\colon \mathbb{N} \to \mathbb{N}$ be a function. The Minimimum Circuit Size Problem for $\mathfrak{C}$ with a fixed parameter s, abbreviated as $\mathsf{MCSP}\text{-}\mathfrak{C}[s]$, is defined as follows*:

- Input. *A string $y \in \{0,1\}^{2^n}$, where $n \in \mathbb{N}$ (inputs not of this form are rejected).*
- Question. *Does $\mathtt{fn}(y)$ have $\mathfrak{C}$-circuits of size at most $s(n)$?*

In our notation, $N = 2^n$ is always the input length of an instance of $\mathsf{MCSP}\text{-}\mathfrak{C}[s]$. Therefore, a polynomial time algorithm for this problem is an algorithm that runs in time $2^{O(n)}$.

Definition 40 (**Succinct MCSP**). *Let $s\colon \mathbb{N} \to \mathbb{N}$ and $t\colon \mathbb{N} \to \mathbb{N}$ be functions. The Succinct Minimum Circuit Size Problem for $\mathfrak{C}$ with fixed parameters s and t, abbreviated as* Succinct-MCSP-$\mathfrak{C}[s,t]$*, is defined as follows*:

- Input. *$n \in \mathbb{N}$, and a list of $t(n)$ pairs (w_i, b_i), where each $w_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$.*
- Question. *Is there a $\mathfrak{C}$-circuit D of size at most $s(n)$ such that $D(w_i) = b_i$ for each $i \in [t(n)]$?*

We also introduce *approximate* versions of MCSP and Succinct-MCSP. We say that a circuit ψ *γ-approximates* a function f if $\Pr_{\boldsymbol{x}}[\psi(\boldsymbol{x}) = f(\boldsymbol{x})] \geq \gamma$.

Definition 41 (**Approximate MCSP**). *Let $\mathfrak{C}$ be a circuit class, $s\colon \mathbb{N} \to \mathbb{N}$, and $\alpha, \beta\colon \mathbb{N} \to (1/2, 1]$, where $\alpha(n) > \beta(n)$ for all n. We let (α,β)-MCSP-$\mathfrak{C}[s]$ denote the following promise problem*:

- Yes instances. *A string $y \in \{0,1\}^{2^n}$ that can be α-approximated by a $\mathfrak{C}$-circuit of size $\leq s$.*
- No instances. *A string $y \in \{0,1\}^{2^n}$ that cannot be β-approximated by a $\mathfrak{C}$-circuit of size $\leq s$.*

Definition 42 (**Approximate Succinct MCSP**). *Let $\mathfrak{C}$ be a circuit class, $s\colon \mathbb{N} \to \mathbb{N}$, and $\alpha, \beta\colon \mathbb{N} \to (1/2, 1]$, where $\alpha(n) > \beta(n)$ for all n. We let (α,β)-Succinct-MCSP-$\mathfrak{C}[s]$ denote the following promise problem*:

- Yes instances. *$n \in \mathbb{N}$ and a list of $t(n)$ pairs (w_i, b_i), where each $w_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$. There exists a $\mathfrak{C}$-circuit of size $\leq s$ that correctly classifies $\geq \alpha \cdot t(n)$ of these pairs.*
- No instances. *$n \in \mathbb{N}$ and a list of $t(n)$ pairs (w_i, b_i), where each $w_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$. Every $\mathfrak{C}$-circuit of size $\leq s$ correctly classifies $\leq \beta \cdot t(n)$ of these pairs.*

Definition 43 (**Average-case MCSP** [HS17]). *We say that* MCSP-$\mathfrak{C}[s]$ *is computed on average by a device D (an algorithm or a family of circuits) if for every large enough n, the following holds:*

- *D* is never incorrect. *For each $y \in \{0,1\}^{2^n}$, $D(y) \in \{\text{MCSP-}\mathfrak{C}[s](y), \text{“?”}\}$.*
- *D* rarely outputs *“?”*. *$\Pr_{z \sim \{0,1\}^{2^n}}[D(z) \in \{0,1\}] \geq 1 - 1/n$.*

We say that MCSP-$\mathfrak{C}[s]$ *is average-case hard for a class of devices $\mathcal{A}$ if no $D \in \mathcal{A}$ computes* MCSP-$\mathfrak{C}[s]$ *on average.*

We assume that the parameter functions s, t, s', α, and β appearing in our statements are time-constructible. We often omit the circuit class $\mathfrak{C}$ when $\mathfrak{C} = \mathsf{Circuit}$, i.e., when the computational problem refers to general boolean circuits.

Definition 44 (**Gap MKtP**). *Given functions $s, c : \mathbb{N} \to \mathbb{N}$ where $s \geq c$,* MKtP$[c,s]$ *is the promise problem whose YES instances are strings of Kt complexity at most $c(N)$ and NO instances are strings of Kt complexity greater than $s(N)$, where N is the length of the input string.*

We say that $\mathfrak{R} = \{\mathcal{R}_n\}_{n \in \mathbb{N}}$ is a *combinatorial property* (of boolean functions) if $\mathcal{R}_n \subseteq \mathcal{F}_n$ for all n. We use $L_{\mathfrak{R}}$ to denote the language of truth-tables of functions in $\mathfrak{R}$. Formally, $L_{\mathfrak{R}} = \{y \mid y = \mathtt{tt}(f) \text{ for some } f \in \mathcal{R}_n \text{ and } n \in \mathbb{N}\}$.

Definition 45 (**Natural Properties** [RR97])**.** *Let $\mathfrak{R} = \{\mathcal{R}_n\}$ be a combinatorial property, $\mathfrak{C}$ a circuit class, and $\mathfrak{D}$ a (uniform or non-uniform) complexity class. We say that $\mathfrak{R}$ is a $\mathfrak{D}$-natural property useful against $\mathfrak{C}[s(n)]$ if there is $n_0 \in \mathbb{N}$ such that the following holds*:

(*i*) Constructivity. $L_{\mathfrak{R}} \in \mathfrak{D}$.

(*ii*) Density. *For every $n \geq n_0$,* $\Pr_{f \sim \mathcal{F}_n}[f \in \mathcal{R}_n] \geq 1/2$.

(*iii*) Usefulness. *For every $n \geq n_0$, we have $\mathcal{R}_n \cap \mathcal{C}_n[s(n)] = \emptyset$.*

It has been observed that the density parameter in the definition above can be amplified using a straightforward reduction (cf. [CIKK16, Lemma 2.7]). In addition, as noticed by [HS17], natural properties and average-case MCSP are equivalent notions.

B (α, β)-MCSP-$\mathfrak{C}[s]$ versus Natural Properties Against $\mathfrak{C}[s]$

We give an example of a consequence of the easiness of (α, β)-MCSP-$\mathfrak{C}[s]$ that does not seem to follow from a natural property with corresponding parameters. For concreteness, our parameters in the statement below correspond to $\alpha = 1$, $\beta = 9/10$, $s = 2^{n^{1/d}}$, and $\mathfrak{C} = \mathsf{AC}^0$ circuits of depth d. Under this choice, there are natural properties against $\mathfrak{C}$-circuits of size s [RR97]. On the other hand, we observe that:

Proposition 46. *Let $\delta = 1/10$. For every fixed $d \geq 2$, the following holds. If $(1, 1-\delta)$-*MCSP-$\mathsf{AC}^0_d[2^{n^{1/d}}] \in \mathsf{P}$*, then there exists a natural property against $\mathsf{AC}^0_d[2^{n/2}]$ computable in quasi-polynomial time.*

Proof. To define the natural property, first pad the input truth-table on n input bits to a truth-table over $m = (n/2)^d$ input bits, then invoke the corresponding $(1, 1-\delta)$-MCSP algorithm on truth-tables of size $M = 2^m$, negating the output. In a bit more detail, the function $f\colon \{0,1\}^n \to \{0,1\}$ is mapped to a function $g\colon \{0,1\}^m \to \{0,1\}$ such that $g(xy) = f(x)$, where $|x| = n$ and $|y| = m - n$. Running the initial algorithm on the function g takes time $\mathsf{poly}(M) = 2^{O(n^d)} = N^{\mathsf{poly}(\log N)}$.

If $f \in \mathsf{AC}^0_d[2^{n/2}]$, then the m-bit boolean function $g \in \mathsf{AC}^0_d[2^{m^{1/d}}]$. On the other hand, if f is a random function, with high probability any circuit of size $2^{(3/4)\cdot n}$ agrees with f on at most a $(1/2) + o(1)$ fraction of inputs. If the corresponding function g could be $(1-\delta)$-approximated by a depth-d circuit C of size $2^{m^{1/d}} = 2^{n/2}$, then some restriction of the y-variables of C must $(1-\delta)$-approximate f, which is impossible for a typical f. This implies that the algorithm described above computes a natural property against $\mathsf{AC}^0_d[2^{n/2}]$. □

Even for $d = 3$, the power of depth-d circuits of size $2^{n/2}$ remains largely unknown, and it is consistent with current knowledge that $\mathsf{NP} \subseteq \mathsf{AC}^0_d[2^{n/2}]$. For this reason, it seems unlikely that the natural property provided by Proposition 46 can be obtained from existing natural properties. (The issue with the reduction above lies on the relaxed density requirement of a natural property: most functions f might produce a function g outside the dense set.)

C Unconditional Lower Bounds

Motivated by Theorems 17 and 18 and Proposition 19, we describe some related lower bounds.

C.1 Lower bounds for good s but bad δ (with respect to Theorem 17)

For convenience, throughout this section we let $\Pi[s,\delta] = (1,1-\delta)\text{-}\mathsf{MCSP}[s]$. We recall a formula lower bound obtained by Hirahara and Santhanam [HS17]. Their result can be stated as follows.

Theorem 47 (A Near-Quadratic Formula Lower Bound [HS17])**.** *There exist functions $\delta(n) > 0$, $\varepsilon(n) \to 0$, and a constant $\lambda \in (0,1)$ for which the following holds:*

$$\Pi[s,\delta] \notin \mathsf{Formula}[N^{2-\varepsilon(n)}],$$

where $s(n) = 2^{n^\lambda}$.[11]

Proof. In [HS17] a near-quadratic formula size lower bound is established for the standard MCSP problem for an appropriate choice of the parameter $s(n)$. Note that MCSP is nothing else than $\Pi[s(n),\delta(n)]$ for $0 < \delta(n) < 2^{-n}$. □

In this section, we give the proof of a slightly stronger form of Theorem 47 with parameters $s(n) = 2^{O(n^\gamma)}$ and $\delta(n) = 2^{-n+\Theta(n^\gamma)}$, where $1/2 \le \gamma < 1$ is an arbitrary constant. (It is possible to establish a more general result with stronger parameters. The crucial point of our exposition is that one can prove a near-quadratic formula lower bound for $s(n) = 2^{o(n)}$.)

Let $\rho\colon [N] \to \{0,1,*\}$ be a *restriction*, and $\boldsymbol{\rho}$ be a *random restriction*, i.e., a distribution of restrictions. We say that $\boldsymbol{\rho}$ is *p-regular* if $\Pr[\boldsymbol{\rho}(i) = *] = p$ and $\Pr[\boldsymbol{\rho}(i) = 0] = \Pr[\boldsymbol{\rho}(i) = 1] = (1-p)/2$ for every $i \in [N]$. In addition, $\boldsymbol{\rho}$ is *k-wise independent* if any k coordinates of $\boldsymbol{\rho}$ are independent.

Proposition 48 (Succinct Random Restrictions (cf. [IMZ12, Vad12]))**.** *There exist explicit p-regular k-wise independent random restrictions $\boldsymbol{\rho}$ distributed over $\rho\colon [N] \to \{0,1,*\}$ samplable with $O(k\log(N)\log(1/p))$ bits. Furthermore, each coordinate of the random restriction can be computed in time polynomial in the number of random bits.*

Let $N = 2^n$. Given a function $F\colon \{0,1\}^N \to \{0,1\}$ and a restriction $\rho\colon [N] \to \{0,1,*\}$, we let $F\restriction_\rho$ be the function in $\{0,1\}^{\rho^{-1}(*)} \to \{0,1\}$ obtained in the natural way from F and ρ.

For convenience, we use $L(F)$ to denote the size (number of leaves) of the smallest formula that computes a function F.

The next result allows us to shrink the size of a formula using a pseudorandom restriction. This restriction can be obtained by a composition of restrictions, which reduces the amount of randomness and the corresponding circuit complexity of the final restriction.

Proposition 49 (Shrinkage for Pseudorandom Restrictions (cf. [IMZ12, HS17]))**.** *Let $F\colon \{0,1\}^N \to \{0,1\}$, $q = p^{1/r}$ for an integer $r \ge 1$, and $L(F)\cdot p^2 \ge 1$. Moreover, let $\mathcal{R}^r_{p,k}$ be a distribution obtained by the composition of r independent q-regular k-wise independent random restrictions supported over $[N] \to \{0,1,*\}$, where $k = q^{-2}$. Then,*

$$\mathbb{E}_{\boldsymbol{\sigma}\sim\mathcal{R}^r_{p,k}}[L(F\restriction_{\boldsymbol{\sigma}})] \le c^r p^2 L(F),$$

where $c \ge 1$ is an absolute constant.

We will also need the following results.

[11] The near-quadratic formula lower bound from [HS17] is stated with a different regime for the parameter $s(n)$.

Fact 50 (Average-Case Hardness)**.** *Let $A_n \subseteq \{0,1\}^n$ and $n \le s(n) \le 2^n/n^2$. There exists a constant $B \ge 1$ such that for every large enough n, if $|A_n| \ge B \cdot s(n) \log s(n)$, then there is a function $h_n \colon A_n \to \{0,1\}$ such that no circuit C of size at most $s(n)$ satisfies $\Pr_{\boldsymbol{a} \sim A_n}[C(\boldsymbol{a}) = h_n(\boldsymbol{a})] \ge 1/2 + 1/100$.*

Proof. Let $\boldsymbol{h} \colon A_n \to \{0,1\}$ be a random function. Using the size of A_n and a concentration bound, $\boldsymbol{h}$ is computed by a fixed circuit C with probability greater than $1/2 + 1/100$ over the choice of $\boldsymbol{a} \sim A$ with probability at most $\exp(-D \cdot s(n) \log s(n))$, for a large enough constant $D = D(B) \ge 4$ independent of n. The result now follows by a union bound using Proposition 15. □

Lemma 51 (Adaptation of Lemma 27 from [HS17])**.** *Let $\rho \colon [N] \to \{0,1,*\}$ be a restriction computed by a circuit C_ρ of size $t(n) \ge n$, where $N = 2^n$ and $C_\rho \colon \{0,1\}^n \to \{0,1\}^2$ (two bits are used to encode a value in $\{0,1,*\}$). Let $F \colon \{0,1\}^N \to \{0,1\}$, and suppose that $L(F\restriction_\rho) = M$. Finally, let*

$$V = \rho^{-1}(*), \quad s(n) = 10 \cdot t(n), \quad \delta(n) = (1/3)(|V| - M)2^{-n}.$$

There exists an absolute constant $B \ge 1$ such that if $|V| - M \ge B \cdot s(n) \log s(n)$, then F does not compute $\Pi[s(n), \delta(n)]$.

Proof. Under these assumptions, we define instances $w^y \in \Pi_N^{\mathsf{yes}}$ and $w^n \in \Pi_N^{\mathsf{no}}$ such that $F(w^y) = F(w^n)$. Below, we identify N and $\{0,1\}^n$. Since $L(F\restriction_\rho) = M$, $F\restriction_\rho$ depends on at most M input coordinates (indexed by elements in V). Let $W \subseteq V \subseteq [N]$ be this set of coordinates.

– $w^y \in \{0,1\}^N$ is obtained from C_ρ by additionally setting each $*$-output of this circuit (encoded as a two-bit string) to 0. Note that w^y viewed as a boolean function is computed by a circuit of size at most $10 \cdot t(n) \le s(n)$. Therefore, $w^y \in \Pi_N^{\mathsf{yes}}$.

– $w^n \in \{0,1\}^N$ is defined to be a function $f \colon \{0,1\}^n \to \{0,1\}$ that agrees with C_ρ over $\{0,1\}^n \setminus V$, evaluates to 0 on W, and otherwise computes as a fixed function $h \colon V \setminus W \to \{0,1\}$ with the following property: no circuit of size $s(n)$ computes h correctly on more than a $(1/2 + 1/100)$-fraction of the inputs in $V \setminus W$. Since $|V \setminus W| \ge |V| - M \ge B \cdot s(n) \log s(n)$, a function h with this property exists by Fact 50. The hardness of w^n over $V \setminus W$ and our choice of $\delta(n)$ imply that $w^n \in \Pi_N^{\mathsf{no}}$.

Finally, using (1) that F restricted to ρ depends only on variables from W, and (2) the definition of the inputs $w^y \in \Pi_N^{\mathsf{yes}}$ and $w^n \in \Pi_N^{\mathsf{no}}$, it follows that $F(w^y) = F(w^n)$. This completes the proof that F does not compute $\Pi[s(n), \delta(n)]$. □

We proceed with the proof of Theorem 47. Fix an arbitrary constant $1/2 \le \gamma < 1$. Now let

$$N \stackrel{\text{def}}{=} 2^n, \quad p \stackrel{\text{def}}{=} 2^{C_p n^\gamma}/N, \quad q \stackrel{\text{def}}{=} p^{1/r}, \quad k \stackrel{\text{def}}{=} q^{-2}, \quad r \stackrel{\text{def}}{=} C_r n^\gamma,$$

where $C_r \ge 1$ is a large enough constant, and $C_p > C_r$ is also a sufficiently large constant.

Lemma 52 (Concentration Bound for $|\boldsymbol{\sigma}^{-1}(*)|$)**.** *Let $k \ge 2$. Then, for $\boldsymbol{\sigma} \sim \mathcal{R}^r_{p,k}$ with parameters as above, we have $\Pr[|\boldsymbol{\sigma}^{-1}(*)| \ge pN/2] \ge 1/2$.*

Proof. Note that $\boldsymbol{\sigma}$ is p-regular and pairwise independent. The result follows from Chebyshev's using mean $\mu = pN$, variance $\sigma^2 = Np(1-p)$, and the value of p. □

Using Proposition 48, it is possible to sample a random restriction $\boldsymbol{\sigma}$ as in the statement of Proposition 49 where each $\sigma\colon [N] \to \{0,1,*\}$ in the support of $\boldsymbol{\sigma}$ has circuit complexity at most

$$t(n) \;\le\; \mathsf{poly}(r, k, \log N, \log(1/q)) \;\le\; \mathsf{poly}(n, k) \;\le\; 2^{n^{1-\gamma}} \;\le\; 2^{n^{\gamma}},$$

where in the last two inequalities we used the definition of k and the assumptions on the constants C_p, C_r, and γ.

Toward a contradiction, let $F\colon \{0,1\}^N \to \{0,1\}$ be a formula of size $L(f)$ that computes $\Pi[s(n), \delta(n)]$, where $L(F) \cdot p^2 = 1$ (note that $L(F) = N^{2-o(1)}$), and let

$$s(n) \stackrel{\text{def}}{=} 10 \cdot t(n), \quad \delta(n) \stackrel{\text{def}}{=} (1/3)(pN/2 - M)2^{-n}, \quad M \stackrel{\text{def}}{=} 10 \cdot c^r p^2 L(F),$$

for a constant $c \ge 1$ as in Proposition 49. Using Proposition 49 and Markov's inequality, Lemma 52, and a union bound, there is a fixed restriction $\rho\colon [N] \to \{0,1,*\}$ for which the following holds:

- For $V \stackrel{\text{def}}{=} \rho^{-1}(*)$, we have $|V| \;\ge\; pN/2$;
- ρ is computed by a circuit $C_\rho\colon \{0,1\}^n \to \{0,1\}^2$ of size at most $t(n) \le 2^{n^\gamma}$;
- $L(F\restriction_\rho) \;\le\; M$.

Using these parameters in the statement of Lemma 51, it is easy to check that $|V| - M \gg B \cdot s(n)\log(s(n))$. Consequently, Lemma 51 implies that $\Pi[s, \delta]$ requires formulas of size $N^{2-o(1)}$. Since our parameters satisfy $s(n) = 2^{O(n^\gamma)}$ and $\delta(n) = 2^{-n+\Theta(n^\gamma)}$, this completes the proof.

Limitations of the Hirahara-Santhanam Lower Bound Technique. Unfortunately, without further modifications, the approach described in this section is not sufficient for obtaining new formula lower bounds (via Lemma 16). Intuitively, the reason is that the probability parameter p must be small enough so that the initial formula simplifies after a pseudorandom restriction. At the same time, the parameter δ must be large enough so that there is a reasonable gap between positive and negative instances in Lemma 51, allowing Lemma 16 to be invoked with nontrivial parameters. However, since p and δ are connected, these are conflicting requirements.

We formalize this argument next. For convenience, we make the following simplifying assumptions.

– In Lemma 16, we get an $\ell(n)$ formula-size lower bound a function over $s(n)/\delta(n)$ input bits.
– The expectation in Proposition 49 is bounded by $p^2 L(F)$.
– In Lemma 51, $\delta(n) = (|V| - M)2^{-n}$, and the final condition is simply $|V| - M \ge s(n)\log s(n)$.
– Lemma 13: The composed restriction $\boldsymbol{\sigma}$ from Proposition 49 satisfies $|V| = |\boldsymbol{\sigma}^{-1}(*)| = pN$.

Under these assumptions, in order to prove a formula lower bound of the form m^d for an m-bit boolean function using Lemma 16, where $d > 1$, we need to have $\ell(n) \ge (s(n)/\delta(n))^d$. Since $|V| = pN = p2^n$, in order to prove a lower bound using Lemma 51, we must take $\delta(n) \le p$ (i.e., p controls the fraction of $*$'s, and if $\delta(n) > p$, there is not enough room to create a negative instance of $\Pi[\cdot, \delta]$ in the proof of Lemma 51). Now in the statement of Proposition 49, we need p small enough to shrink the initial formula F of size $N \cdot \ell(n)$ to a restricted formula $F\restriction_\rho$ such that $L(F\restriction_\rho) = M$ and $|V| - M \ge s(n)\log s(n)$ (Lemma 51). Consequently, $M \le |V| \le pN$. This requires $p^2 L(F) = p^2 N \ell(n)$ to be at most $M \le pN$, which in turn forces $p^2 N (s(n)/\delta(n))^d \le pN$,

using the lower bound on $\ell(n)$. We get from this inequality that $p \leq (\delta(n)/s(n))^d$. Now recall that $\delta(n) \leq p$ in order for the approach to work, which gives $\delta(n) \leq (\delta(n)/s(n))^d$, where $d > 1$ and of course $s(n) > 1$ (in order to simplify the statement of Lemma 16 and make it nontrivial, the stronger assumption $s(n) \geq n$ is employed there). Finally, it follows that the latter inequality is false, since $\delta(n) < 1$.

C.2 Lower bounds for good δ but bad s (with respect to Theorem 18)

For convenience, throughout this section we let $\Pi[s, \delta] = (1, 1-\delta)\text{-}\mathsf{MCSP}[s]$. We explore two approaches to lower bounds in connection with Theorem 18. First, we use the technique of deterministic restrictions [CR96] to prove the following result.

Theorem 53 (Small-Depth Lower Bounds for Large $s(n)$)**.** *Let $s(n) = 2^{(1-o(1))n}$ and $\delta = 1/2 - \lambda$, where $\lambda > 0$ is fixed but arbitrarily small. Then for every $d \geq 1$ there is $\varepsilon > 0$ such that $\Pi[s, \delta] \notin \mathsf{AC}^0_d[N^{1+\varepsilon}]$.*

We use the following lemma, which allows us to trivialize a very small constant-depth circuit by fixing only a few input variables.

Lemma 54 (Deterministic Restriction Lemma [CR96])**.** *Let C be a circuit over N input variables of size S and depth d, and let $\alpha_d = 4^{-d}$. Then there is a restriction $\rho\colon [N] \to \{0, 1, *\}$ that fixes at most $5 \cdot S^{1-\alpha_d}$ inputs such that $C\restriction_\rho$ is constant.*

Proof of Theorem 53. Suppose there is a sequence $\{D_N\}$ of depth-d circuits of size $N^{1+\varepsilon_d}$ that compute $\Pi[s, \delta]$, where ε_d is any small enough positive constant such that $(1+\varepsilon_d)(1-\alpha_d) < 1$. Let ρ be a restriction that fixes less than $N^{1-\Omega(1)}$ input variables such that $D_N\restriction_\rho$ is constant, as given by Lemma 54 and our choice of parameters. To complete the proof, we construct inputs $w^y, w^n \in \{0,1\}^N$ extending the restriction ρ such that $w^y \in \Pi^{\mathsf{yes}}_N$ and $w^n \in \Pi^{\mathsf{no}}_N$.

For w^y, we fix every $*$-coordinate of ρ to 0. Since $|\rho^{-1}(1)| \leq N^{1-\Omega(1)}$, it is clear that w^y encodes a function that can be computed by a circuit (or even a DNF) of size at most $N^{1-\Omega(1)} \cdot \mathsf{poly}(n) \leq s(n)$. Consequently, $w^y \in \Pi^{\mathsf{yes}}_N$.

On the other hand, let $\boldsymbol{w^n} \in \{0,1\}^N$ be a uniformly random completion of ρ. Then, since $|\rho^{-1}(*)| \geq (1-o(1))N$, a standard argument using a union bound and Lemma 13 shows that almost surely every circuit of size $s(n)$ computes $\mathsf{fn}(\boldsymbol{w^n})$ correctly on at most $(1/2 + o(1))2^n$ input strings. By fixing w^n to be any string with this property, we get $w^n \in \Pi^{\mathsf{no}}_N$. □

Note that, as opposed to Theorem 47, here δ is a (rather large) constant. Neither of these results can be used for hardness magnification, but the reason is different in each case. In Theorem 47, the relative distance δ is prohibitively small, while in Theorem 53 it is the size bound that is prohibitively large.

Succinct restrictions, uniformity, and unconditional lower bounds. We note that the lower bound described in Section C.1 and the proof of Theorem 53 follow the same idea of finding a restriction of bounded circuit complexity that simplifies a formula or a bounded-depth circuit, respectively. Is there a P-uniform (in 1^N) sequence of bounded-depth circuits C_N over N-input variables and of size $N^{1+\varepsilon}$ that cannot be made constant by a restriction $\rho_N\colon [N] \to \{0, 1, *\}$ of circuit complexity $s(n) \leq \mathsf{poly}(n)$ and with $|\rho^{-1}(*)| = \Omega(N)$?

By dropping the complexity requirement, the corresponding restrictions exist (Lemma 54). One can define from such a uniform sequence of circuits an NEXP-verifier (over inputs of the form 1^n) that does not admit succinct (polynomial-size) witnesses, in the formal sense of [IKW02, Wil16]. In particular, the existence of uniform circuits as above implies that $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$. Therefore, showing that the magnification approach via Theorem 18 *cannot* lead to new unconditional lower bounds is connected to the existence of *non-constructive* sequences of circuits that do not admit succinct restrictions.

It is possible to improve the parameters in Theorem 53 using a different approach. We rely on the construction of pseudorandom functions [Nis91] that can be computed by slightly super-polynomial size circuits and that are secure against AC^0 circuits of arbitrary fixed depth.[12] A related and perhaps more direct point of view is that the result follows from the fact that there are no AC^0-natural proofs against $\mathsf{AC}^0[2]$ circuits of super-polynomial size [RR97].

Theorem 55 (Small-Depth Lower Bounds for Medium $s(n)$)**.** *Let $s(n) = n^{\omega(1)}$ and $\delta = 1/2 - \lambda$, where $\lambda > 0$ is fixed but arbitrarily small. Then for every $d \geq 1$ and every $\ell \geq 1$ we have $\Pi[s, \delta] \notin \mathsf{AC}^0_d[N^\ell]$.*

Proof Sketch. It is enough to observe that any circuit computing a total function that agrees with $\Pi[s, \delta] = (1, 1 - \delta)\text{-}\mathsf{MCSP}[s]$ defines a natural property useful against size-s circuits. The result then follows immediately from [RR97, Theorem 4.3]. □

This result is much closer to the parameters in Theorem 18, but still insufficient for lower bounds.

[12] To be more precise, these are actually *implicitly* computable PRGs, and not PRFs in the usual cryptographic sense. The difference is that the corresponding distinguishers are allowed to run in time polynomial in the size of the output string, while in the (cryptographic) PRF setting a distinguisher typically has oracle access to the function and runs in sub-linear time in the size of the truth table.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il