Weak Lower Bounds on Resource-Bounded Compression Imply Strong Separations of Complexity Classes[*]

Dylan M. McKay
dmmckay@mit.edu
MIT CSAIL

Cody D. Murray[†]
codymurraycs@gmail.com

R. Ryan Williams[‡]
rrw@mit.edu
MIT CSAIL

ABSTRACT

The Minimum Circuit Size Problem (MCSP) asks to determine the minimum size of a circuit computing a given truth table. MCSP is a natural and powerful string compression problem whose **NP**-hardness remains open. Recently, Oliveira and Santhanam [FOCS 2018] and Oliveira, Pich, and Santhanam [ECCC 2018] demonstrated a "hardness magnification" phenomenon for MCSP in restricted settings. Letting MCSP$[s(n)]$ be the problem of deciding if a truth table of length 2^n has circuit complexity at most $s(n)$, they proved that small (fixed-polynomial) *average case* circuit/formula lower bounds for MCSP$[2^{\sqrt{n}}]$, or lower bounds for *approximating* MCSP$[2^{o(n)}]$, would imply major separations such as $\mathbf{NP} \not\subset \mathbf{BPP}$ and $\mathbf{NP} \not\subset \mathbf{P}/\text{poly}$.

We strengthen these results in several directions, obtaining magnification results from *worst-case* lower bounds on *exactly* computing the *search version* of generalizations of MCSP$[s(n)]$, which also extend to time-bounded Kolmogorov complexity. In particular, we show that search-MCSP$[s(n)]$ (where we must output a $s(n)$-size circuit when it exists) admits extremely efficient $\mathbf{AC}^0$ circuits and streaming algorithms using Σ_3SAT oracle gates of small fan-in (related to the size $s(n)$ we want to test).

For $A : \{0,1\}^\star \to \{0,1\}$, let search-MCSP$^A[s(n)]$ be the problem: *Given a truth table T of size $N = 2^n$, output a Boolean circuit for T of size at most $s(n)$ with AND, OR, NOT, and A-oracle gates (or report that no such circuit exists).* Some consequences of our results are:

- For reasonable $s(n) \geq n$ and $A \in \mathbf{PH}$, if search-MCSP$^A[s(n)]$ does not have a 1-pass deterministic poly$(s(n))$-space streaming algorithm with poly$(s(n))$ update time, then $\mathbf{P} \neq \mathbf{NP}$.

For example, proving that it is impossible to synthesize SAT-oracle circuits of size $2^{n/\log^\star n}$ with a streaming algorithm on truth tables of length $N = 2^n$ using N^ε update time and N^ε space on length-N inputs (for some $\varepsilon > 0$) would already separate **P** and **NP**. Note that some extremely simple functions, such as EQUALITY of two strings, already satisfy such lower bounds.

- If search-MCSP$[n^c]$ lacks $\tilde{O}(N)$-size, $\tilde{O}(1)$-depth circuits for a $c \geq 1$, then $\mathbf{NP} \not\subset \mathbf{P}/\text{poly}$.
- If search-MCSP$[s(n)]$ doesn't have circuits of $N \cdot \text{poly}(s(n))$ size and $O(\log N)$ depth, then $\mathbf{NP} \not\subset \mathbf{NC}^1$. It is known that MCSP$[2^{\sqrt{n}}]$ does not have *formulas* of $N^{1.99}$ size [Hirahara and Santhanam, CCC 2017].
- If there is an $\varepsilon > 0$ such that for all $c \geq 1$, search-MCSP$[2^{n/c}]$ does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth $\mathbf{ACC}^0$ circuits, then $\mathbf{NP} \not\subset \mathbf{ACC}^0$. Thus the amplification results of Allender and Koucký [JACM 2010] can extend to problems in **NP** and beyond.

Furthermore, if we substitute $\oplus$**P**, **PP**, **PSPACE**, or **EXP**-complete problems for the oracle A, we obtain separations for those corresponding complexity classes instead of **NP**. Analogues of the above results hold for time-bounded Kolmogorov complexity as well.

CCS CONCEPTS

• **Theory of computation** → **Complexity classes**; **Circuit complexity**; Streaming models.

KEYWORDS

hardness magnification, minimum circuit size problem, Kolmogorov complexity, streaming algorithms

ACM Reference Format:
Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. 2019. Weak Lower Bounds on Resource-Bounded Compression Imply Strong Separations of Complexity Classes. In *Proceedings of the 51st Annual ACM SIGACT Symposium on the Theory of Computing (STOC '19), June 23–26, 2019, Phoenix, AZ, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3313276.3316396

1 INTRODUCTION

In this paper, we show how extremely weak-looking lower bounds on solving canonical string compression problems would imply major separations in complexity theory. Let $s : \mathbb{N} \to \mathbb{N}$ satisfy $s(n) \geq n$ for all n. We start by considering the following circuit synthesis problem:

Problem: MCSP$[s(n)]$
Given: A function $f : \{0,1\}^n \to \{0,1\}$, presented as a truth table of $N = 2^n$ bits.
Decide: Does f have a (fan-in two) Boolean circuit of size at most $s(n)$?

We can naturally view MCSP as a compression problem: given a string, find a small circuit whose truth table reproduces the

[*] Supported by NSF CAREER 1741615.

[†] Portions of this work were completed while the author was a PhD student at MIT, and as a Research Fellow at the Simons Institute, UC Berkeley.

[‡] Portions of this work were completed while visiting the Simons Institute and UC Berkeley EECS department.

STOC '19, June 23–26, 2019, Phoenix, AZ, USA

ACM ISBN 978-1-4503-6705-9/19/06…$15.00
https://doi.org/10.1145/3313276.3316396

string. MCSP seems to have first been studied in the 1950s [Tra84]. Note MCSP$[s(n)]$ is only non-trivial for sufficiently small $s(n)$, e.g., $s(n) \leq 2^{n+1}/n$ (otherwise, *every* input is a YES instance), and for such non-trivial $s(n)$, MCSP$[s(n)]$ is in **NP**. Moreover, there is an algorithm for MCSP$[s(n)]$ running in both $s(n)^{O(s(n))}$ time and $\tilde{O}(s(n)+n)$ space (enumerating all possible size-$s(n)$ circuits). No further improvements in the time complexity of MCSP$[s(n)]$ are known.

In 2001, Kabanets and Cai [KC00] revived the "modern" study of MCSP in theoretical computer science, observing that the Natural Proofs barrier [RR97] strongly suggests that MCSP$[n^c]$ is *not* in **P** for sufficiently large c; otherwise, strong one-way functions do not exist.[1] Thus it is widely believed that MCSP[poly(n)] is not efficiently solvable, although it is still open whether the general MCSP problem is **NP**-hard, despite much recent work [MW15, AHK17, HP15, HW16, AH17, HS17, HOS18].[2] MCSP has recently taken on larger significance for cryptography, since Hirahara recently showed in a celebrated work [Hir18] that there is a worst-case to average-case reduction for the approximation version.

Considering the enormous difficulty of proving polynomial-time lower bounds on problems in **NP**, we may ask if it is possible to prove instead that MCSP[poly(n)] cannot be solved in almost-linear time (e.g. $N^{1+o(1)}$) and sub-polynomial space (e.g. $N^{o(1)}$). Over the years, there has been considerable progress in lower bounds for the low time-space setting [BJS01, Ajt02, BSSV03, Ajt05, FLvMV05, Wil08, BW15], so there is hope that such lower bounds could be proved.

In this work, we show (among many other results) that an $N \cdot \text{poly}(\log N)$-time lower bound on poly$(\log N)$-space **deterministic streaming algorithms** for MCSP$[n^c]$ is already as difficult as separating **P** from **NP**! Note that deterministic streaming algorithms are very restricted (one cannot even compute whether two given N-bit strings are equal with $o(N)$ space). Our result is one consequence of a more general theorem proved about a generalization of MCSP, in which we allow compression by circuits with oracle gates (as defined in prior work [AHK17, IKV18]). This "boosting" of a modest time-space lower bound for an **NP** problem to $\mathbf{P} \neq \mathbf{NP}$ can be seen as a form of *hardness magnification*, a term recently coined by Oliveira and Santhanam [OS18]. Intuitively, a hardness magnification result is a kind of "slippery slope" theorem, showing how a very modest-looking lower bound for a believably-hard problem counterintuitively implies incredibly strong lower bounds (perhaps for a different hard problem). Other examples of similar phenomena are known, for $n^{1-\varepsilon}$-approximations to Clique [Sri03], low-depth circuit lower bounds for $\mathbf{NC}^1$ [AK10], and sublinear-depth circuit lower bounds for **P** [LW12].

1.1 Our Results

To state our theorems, we first need a couple of concepts for "generalized" Boolean circuits. Let $A : \{0,1\}^\star \rightarrow \{0,1\}$. An *A-oracle circuit* has a gate basis of OR, AND, NOT, and all finite slices of A (the function A restricted to inputs of a fixed length). The MCSP$^A[s(n)]$ problem asks, given $f : \{0,1\}^n \rightarrow \{0,1\}$ presented as a truth table, whether f has an A-oracle circuit of at most $s(n)$ gates. Its search version, search-MCSP$^A[s(n)]$, requires *outputting* an A-oracle circuit when it exists. The Σ_3SAT^A problem asks, given an A-oracle formula $F(x,y,z)$ *as input* (a Boolean formula over Boolean variables with $\wedge$, $\vee$, $\neg$ and A-oracle predicates), whether there exists an assignment to x such that for all assignments to y, there exists an assignment to z such that $F(x,y,z)$ outputs true.

Our first main theorem constructs a super-efficient **AC** circuit family for MCSP^A with short oracle calls to Σ_3SAT^A:

Theorem 1.1 (Section 3). *Let $s(n) \geq n$ and let $\ell(n) \geq s(n)^2$ for all n, where both are time constructible. Let $A : \{0,1\}^\star \rightarrow \{0,1\}$ be an arbitrary oracle. There is a uniform* **AC** *circuit family for MCSP$^A[s(n)]$ on 2^n-bit inputs of $\tilde{O}(2^n \cdot s(n)^2)$ size and $O(n/\log \ell(n))$ depth with Σ_3SAT^A oracle gates, where each oracle gate takes only $\tilde{O}(\ell(n))$ bits of input.*

Note there are three tunable parameters in Theorem 1.1: the oracle A, the MCSP parameter $s(n)$, and a larger parameter $\ell(n)$ which affects the circuit depth and the fan-in of the oracle gates. By setting them appropriately, we can derive many consequences regarding hardness magnification for MCSP.

Our second main theorem gives an efficient streaming algorithm for MCSP$^A[s(n)]$, assuming the algorithm has oracle access to Σ_3SAT^A and can make short queries to the oracle.

Theorem 1.2 (Section 4). *Let $s(n) \geq n$ for all n, where $s(n)$ is time constructible. Let $A : \{0,1\}^\star \rightarrow \{0,1\}$ be an arbitrary oracle. There is a (one-pass) streaming algorithm for search-MCSP$^A[s(n)]$ on 2^n-bit inputs running in $2^n \cdot \tilde{O}(s(n))$ time with $\tilde{O}(s(n)^2)$ update time and $\tilde{O}(s(n))$ space, using an oracle for Σ_3SAT^A with queries of length $\tilde{O}(s(n))$.*

One way to prove Theorem 1.2 would be to show that the circuit family derived in Theorem 1.1 can be evaluated in a time and space-efficient way. We instead give a direct proof (an explicit streaming algorithm).

Weak Streaming Lower Bounds for Compression Imply $\mathbf{P} \neq \mathbf{NP}$. Applying the assumption $\mathbf{P} = \mathbf{NP}$ to the oracle calls in Theorem 1.2, we obtain a magnification from modest streaming algorithm lower bounds all the way to $\mathbf{P} \neq \mathbf{NP}$:

Theorem 1.3 (Section 5). *If there is some time-constructible $s(n) \geq n$ and an $A \in$* **PH** *such that search-MCSP$^A[s(n)]$ is not solvable by a poly$(s(n))$-space streaming algorithm with poly$(s(n))$ update time, then* $\mathbf{P} \neq \mathbf{NP}$.

(Note that we do not require streaming algorithms to read a fresh bit in each step; the algorithm may work for some time before requesting the next bit of the stream.) For example, if one can show that we cannot compress N-bit strings to SAT-oracle circuits of size $N^{o(1)}$ by streaming algorithms running in $N^{1+\varepsilon}$ total time and N^ε space for all $\varepsilon > 0$, then $\mathbf{P} \neq \mathbf{NP}$. Note there is no *information-theoretic* barrier to such a streaming algorithm, only a computational one: in N^ε space, one could easily hold a circuit of size $N^{o(1)}$. Let us stress that the hypothesis of Theorem 1.3 is widely believed to hold: if it were false, then MCSP is trivially in **P**

[1] In particular, if MCSP$[n^c] \in \mathbf{P}$ for all $c \geq 1$, then there are **P**-natural properties useful against **P**/poly. For a more fine-grained discussion of the relationship, see Chow [Cho11].

[2] Observe that MCSP$[n^c]$ can be solved in $2^{\tilde{O}(n^c)}$ time, which is quasi-polynomial in the input length $N = 2^n$, so it is probably not **NP**-hard; nevertheless it is widely believed to not be in **P**.

(and much more), implying that strong one-way functions do not exist and extremely strong compression is possible efficiently.

Comparing Theorem 1.3 With [OS18].

Oliveira and Santhanam [OS18] show that if *approximately* solving MCSP[$2^{\sqrt{n}}$] (the decision version of MCSP, with no oracle) to within an $O(n)$ factor requires $\Omega(N)$-time by randomized algorithms with two-sided error, then $\mathbf{BPP} \not\subseteq \mathbf{NP}$ (equivalent to $\mathbf{RP} \neq \mathbf{NP}$). In particular, their approximation version of MCSP only requires that a candidate algorithm distinguish truth tables with circuit complexity at most $s(n)$, from truth tables which need a *constant fraction of entries modified* in order to have circuit complexity at most $s(n)$. Proving a lower bound against such a weak guarantee seems intuitively much more difficult than proving a worst-case lower bound against MCSP. (They only need an $\Omega(N)$-time lower bound, because — as in Srinivasan [Sri03] — a random sample of the input preserves the answer to the approximation problem with high probability; this does not hold for the worst-case version.) While their consequence is stronger ($\mathbf{RP} \neq \mathbf{NP}$ rather than $\mathbf{P} \neq \mathbf{NP}$), the hypothesis of Theorem 1.3 only requires a slightly super-linear time worst-case lower bound for the *search* version of MCSP against deterministic streaming algorithms using sub-linear space.

Weak Circuit Lower Bounds for Compression Imply Super Polynomial Circuit Lower Bounds. Applying Theorem 1.1 with different parameter settings, we show how modest $\mathbf{TC}^0$ lower bounds for MCSP^{SAT} imply $\mathbf{NP} \not\subset \mathbf{TC}^0$, modest log-depth circuit lower bounds imply $\mathbf{NP} \not\subset \mathbf{NC}^1$, and larger-depth circuit lower bounds imply $\mathbf{NP} \not\subset \mathbf{P}$/poly.

Theorem 1.4 (Section 5). *Let $s(n) \geq n$, and let $A \in$* **PH**.

- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, search-MCSP^A[$2^{\varepsilon n/c}$] on inputs of length $N = 2^n$ does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth* $\mathbf{TC}^0$ *circuits, then* $\mathbf{NP} \not\subset \mathbf{TC}^0$.[3]
- *If search-MCSP^A[$s(n)$] on inputs of length $N = 2^n$ does not have circuits of $N \cdot poly(s(n))$ size and $O(\log N)$ depth, then* $\mathbf{NP} \not\subset \mathbf{NC}^1$.
- *If search-MCSP^A[$s(n)$] on inputs of length $N = 2^n$ does not have circuits of $N \cdot poly(s(n))$ size and $poly(s(n))$ depth, then* $\mathbf{NP} \not\subset \mathbf{P}$*/poly.*

For example, if we find an $c \geq 1$ and A in the polynomial-time hierarchy such that compression of N-bit strings to $(\log N)^c$-size A-oracle circuits cannot be performed by $N \cdot \text{poly}(\log N)$-size $O(\log N)$-depth circuits, then we can conclude **NP** does not have polynomial-size formulas. If the depth lower bound can be improved to arbitrary polylogs, then already **NP** is not in **P**/poly.

Comparison With [OS18] and [OPS18]. Oliveira and Santhanam show that $N \cdot \text{poly}(\log N)$-size *formula* lower bounds for solving MCSP[n^c] (decision version, with no oracle) in an *approximate* or *average-case* setting would imply $\mathbf{NP} \not\subset \mathbf{NC}^1$. While their required formula lower bound is more modest, the "decision version", "average-case", "approximate", and "no oracle" restrictions would presumably make proving a lower bound considerably more difficult. Oliveira, Pich, and Santhanam [OPS18] show that lower bounds on $O(n)$-approximation to MCSP[$2^{\varepsilon n}$] (for all $\varepsilon > 0$) with $N^{1+\delta}$-size circuits imply $\mathbf{NP} \not\subset \mathbf{P}$/poly, whereas our results show $\mathbf{NP} \not\subset \mathbf{P}$/poly follows from $N^{1+\delta}$-size N^{δ}-depth circuit lower bounds on *exactly* solving MCSP[$2^{\varepsilon n}$]. However, note that proving even lower bounds for $O(N)$-size $O(\log N)$-depth circuits remains an open challenge in complexity theory, even for functions with $N^{\Omega(1)}$ outputs.

Comparison With [AK10]. It is also interesting to compare the $\mathbf{TC}^0$ results of Theorem 1.4 with the work of Allender and Koucký on amplifying lower bounds for low-depth circuit classes [AK10]. They showed that for some natural $\mathbf{NC}^1$ problems such as Boolean Formula Evaluation, proving that there are no $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth $\mathbf{TC}^0$ circuits for those problems imply a full separation: $\mathbf{NC}^1 \not\subset \mathbf{TC}^0$. The first bullet of Theorem 1.4 manages to prove an analogous result for some problems in **PH**! Allender and Koucký also show unconditionally that for sufficiently large d, SAT does not have $n^{1+1/(3d)}$-size d-depth *uniform* $\mathbf{TC}^0$ circuits. The proof of Theorem 1.4 implies that slightly stronger results for MCSP^{SAT}[$2^{o(n)}$] would separate **NP** and uniform $\mathbf{TC}^0$. (In fact, analogous statements hold for compression problems in $\oplus$**P**, $\mathbf{P}^{\mathbf{PP}}$, **PSPACE**, and **EXP**; see Theorems 1.5 and 1.6 below.)

Super-Polynomial Lower Bounds for Larger Complexity Classes. In general, MCSP^A gets "harder" as the oracles A get more expressive. By choosing more powerful oracles A in the statement of Theorem 1.1, we obtain magnification for problems in **EXP**, **PSPACE**, $\mathbf{P}^{\mathbf{PP}}$, and $\oplus$**P** as well. Allender-Buhrman-Koucký-van Melkebeek-Ronneburger [ABK$^+$06] showed that MCSP^{QBF} is **PSPACE** complete under randomized poly-time Turing reductions: for all $\delta > 0$, every **PSPACE** language can be decided in randomized polynomial time with oracle calls to MCSP^{QBF}[$2^{\delta n}$].[4] They also showed MCSP^{EXP} is **EXP**-complete under **P**/poly and **NP** Turing reductions. Impagliazzo, Kabanets, and Volkovich [IKV18] gave oracles $A \in \oplus P$ and $B \in$ **PP** such that MCSP^A is $\oplus$**P**-complete and MCSP^B is **PP**-complete under randomized poly-time Turing reductions. For these complexity classes C, Theorem 1.1 implies that modest lower bounds for MCSP^A with $A \in C$ already proves $\mathbf{P} \neq C$ and C. Here are two formalizations of this general statement.

Theorem 1.5 (Magnifying Streaming Lower Bounds for Harder MCSP Versions, Section 5). *Let C be one of* **NP**, **PP**, $\oplus$**P**, *or* **PSPACE**. *Suppose there is a constructible $s(n)$ and oracle $A \in C$ such that for all $c \geq 1$, search-MCSP^A[$s(n)$] on inputs of length 2^n has no $s(n)^c$-space streaming algorithm with update time $s(n)^c$. Then* $\mathbf{P} \neq C$.

Theorem 1.6 (Magnifying Low-Depth Circuit Lower Bounds for Harder MCSP, Section 5). *Let C be one of* **NP**, **PP**, $\oplus$**P**, *or* **PSPACE**. *Suppose there is some $s(n)$ and oracle $A \in C$ such that for all $c \geq 1$, search-MCSP^A[$s(n)$] on inputs of length 2^n has no circuits of depth $O(n)$ and $2^n \cdot s(n)^c$ size. Then C does not have polynomial-size formulas (i.e., $C \not\subset$* $\mathbf{NC}^1$*).*

For instance, proving MCSP^{EXP}[n^{10}] does not have quasi-linear size circuits of logarithmic depth would imply $\mathbf{EXP} \not\subset \mathbf{NC}^1$.

Kt and KT Complexity. Our techniques can also be applied to time-bounded analogues of Kologorov complexity:

[3] Note that $\mathbf{TC}^0$ could be substituted with any other constant-depth circuit family, such as $\mathbf{AC}^0$[6].

[4] Their result was not explicitly stated in this way, but it follows easily from padding.

Problem: MKtP[$p(N)$] [Lev84]
Given: A string $x \in \{0,1\}^N$.
Decide: Is there a Turing machine M of description length c that prints x in at most t steps, where $c + \log_2(t) \leq p(N)$?

Problem: MKTP[$p(N)$] [All01]
Given: A string $x \in \{0,1\}^N$.
Decide: Is there a Turing machine M of description length c that, given i, prints the i-th bit of x in at most t steps, where $c + t \leq p(N)$?

While MKTP[$p(N)$] is an **NP** problem like MCSP (in fact, the KT complexity of a truth table, which minimizes the measure in the MKTP definition above, is always within polynomial factors of the minimum circuit complexity [All01]), MKtP[$p(N)$] is a more difficult problem (due to the logarithm of running time). Furthermore, MKtP has analogous properties to MCSP$^{\mathbf{EXP}}$: for large enough $p(N)$, MKtP[$p(N)$] is complete for **EXP** under **NP** and **P**/poly Turing reductions [ABK$^+$06]. Thus MKTP and MKtP are closely related to MCSP and MCSP$^{\mathbf{EXP}}$, respectively. We observe that the proof strategy of Theorems 1.1 and 1.2 extend to MKTP and MKtP in an analogous way (Theorems 3.2 and 4.2 for MKTP, Theorems 3.3 and 4.2 for MKtP), leading to the following consequences:

Theorem 1.7 (Consequences for MKTP and MKtP). *Let $p(N) \geq \log(N)$ be time constructible.*

- *If MKTP[$p(N)$] is not solvable by a poly($p(N)$)-space streaming algorithm with poly($p(N)$) update time, then* $\mathbf{P} \neq \mathbf{NP}$.
- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, MKTP[$N^{\varepsilon/c}$] does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth* $\mathbf{TC}^0$ *circuits, then* $\mathbf{NP} \not\subset \mathbf{TC}^0$.
- *If MKTP[$p(N)$] does not have $N \cdot poly(p(N))$-size $O(\log(N))$-depth circuits, then* $\mathbf{NP} \not\subset \mathbf{NC}^1$.
- *If MKTP[$p(N)$] does not have $N \cdot poly(p(N))$-size poly($p(N)$)-depth circuits, then* $\mathbf{NP} \not\subset \mathbf{P}/poly$.
- *If MKtP[$p(N)$] does not have $N \cdot poly(p(N))$-size poly($p(N)$)-depth circuits, then* $\mathbf{EXP} \not\subset \mathbf{P}/poly$.
- *If MKtP[$p(N)$] does not have $N \cdot poly(p(N))$-size $O(\log(N))$-depth circuits, then* $\mathbf{EXP} \not\subset \mathbf{NC}^1$.
- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, MKtP[$N^{\varepsilon/c}$] does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth* $\mathbf{TC}^0$ *circuits, then* $\mathbf{EXP} \not\subset \mathbf{TC}^0$.

As might be expected, the above claims also hold for the corresponding search versions, search-MKTP and search-MKtP, and relativized versions with MKTP^A and MKtP^A hold with appropriate modifications.

Comparison With [OS18] and [OPS18]. Both references [OS18] and [OPS18] show that lower bounds on approximating MKtP would imply lower bounds against **EXP**. Consider the problem Gap-MKtP[$\alpha(N), \beta(N)$] where we are promised that the Kt complexity of a given string of length N is either at most $\alpha(N)$ or at least $\beta(N)$, and we have to distinguish the two cases. The first reference shows that if there is a $\delta > 0$ and an $N^{1+\delta}$-size lower bound on Gap-MKtP[$N^\varepsilon, N^\varepsilon + 5\log(N)$] for all $\varepsilon > 0$, then $\mathbf{EXP} \not\subset \mathbf{P}$/poly. The second reference generalizes this connection to other circuit classes $\mathcal{C} \subset \mathbf{P}$/poly, showing that an $N^{1+\delta}$-size lower bound on $\mathcal{C}$-circuits for Gap-MKtP[$N^\varepsilon, N^\varepsilon + c\log(N)$] for all $\varepsilon > 0$ would imply $\mathbf{EXP} \not\subset \mathcal{C}$. Theorem 1.7 strengthens several of the prior magnification results in multiple ways. In particular, we show lower bounds on the *exact* MKtP problem (with no gap) against $N^{1+\varepsilon}$-size N^ε-depth circuits are already sufficient for $\mathbf{EXP} \not\subset \mathbf{P}$/poly. Finally, [OPS18] also show other interesting connections of MKtP to other classes, e.g., B_2 and U_2 formula lower bounds and AND-THR-THR-XOR circuit lower bounds, which we do not discuss here. Our techniques do not seem to apply to these cases.

1.2 Intuition

To give a feeling for our results, let us describe the proof of Theorem 1.3 at a high level: how minor streaming lower bounds on compression by circuits would imply $\mathbf{P} \neq \mathbf{NP}$. We proceed by proving the contrapositive: assuming $\mathbf{P} = \mathbf{NP}$, we construct an extremely efficient streaming algorithm that can compress 2^n-bit strings with small $s(n)$-size circuits, where $s(n) \geq n$.

It would suffice to design a good streaming algorithm that always maintains a circuit of size $s(n)$ that "agrees" with all bits it has seen so far, or reports when no such circuit exists. Our first idea is to identify an intermediate problem that we call Circuit-Min-Merge, which is not too complex, but can help maintain such a circuit over time. For simplicity, here let's say that Circuit-Min-Merge takes as input two $s(n)$-size circuits C_1 and C_2, along with disjoint intervals $I_1, I_2 \subseteq [1, 2^n]$ (specified in $O(n)$ bits), and produces a circuit C' of size $s(n)$ which agrees with C_1 on all inputs x_i where $i \in I_1$, and agrees with C_2 on all inputs x_j where $j \in I_2$. (If there is no such C', Circuit-Min-Merge reports that.) Now we observe:

- Circuit-Min-Merge is computable in the polynomial-time hierarchy. In particular, the problem of printing bits from the description of a fixed C' can be placed in $\Sigma_3\mathbf{P}$, by guessing a C' of size at most $s(n)$, checking for all inputs in I_1 and I_2 that C' is consistent with C_1 and C_2 (respectively), and checking for all $C'' < C$ (with respect to some ordering) that C'' is not consistent with C_1 and C_2. Our assumption $\mathbf{P} = \mathbf{NP}$ implies that Circuit-Min-Merge has a poly($s(n)$) time algorithm on inputs of length $\tilde{O}(s(n))$.
- Armed with a poly($s(n)$)-time algorithm for Circuit-Min-Merge, we *can* quickly maintain a circuit of size $s(n)$ consistent with the input read: inductively suppose that we have a circuit C that is consistent with all previous bits read. For the next block B of $s(n)$ consecutive bits of the input, we can construct a trivial DNF F of size $n \cdot s(n)$ that is consistent with the inputs in B, and run Circuit-Min-Merge in poly($s(n)$) time on C and F to produce a new circuit that is consistent with all input read so far, including B. This uses poly($s(n)$) update time per bit read, and poly($s(n)$) space.

We conclude that if $\mathbf{P} = \mathbf{NP}$, then for all (reasonable) functions $s(n)$, we can compress strings to size-$s(n)$ circuits with a streaming algorithm using poly($s(n)$) update time and poly($s(n)$) space.

The above approach can be generalized in many ways. For one, we can easily extend it to MCSP with oracle gates, by defining a stronger Circuit-Min-Merge problem that can handle such gates. (For example, the SAT-oracle version of Circuit-Min-Merge is in $\Sigma_4\mathbf{P}$, thus $\mathbf{P} = \mathbf{NP}$ also implies efficient streaming algorithms for compression with SAT-oracle circuits.) For another, the ideas can be extended to time-bounded Kolmogorov complexity (both KT and Kt complexity), by defining appropriate analogues of Circuit-Min-Merge which allow us to consistently maintain a short Turing machine description of our input over time.

Extending the above outline to restricted circuit classes (rather than streaming algorithms) is more involved. To show that (for example) $\mathbf{NP} \subset \mathbf{TC}^0$ implies small $\mathbf{TC}^0$ circuits for compression by $s(n)$-size circuits, we need to make a few modifications we briefly describe at a high level (but require care to work). First, we allow Circuit-Min-Merge to take in an unbounded number of circuits on disjoint intervals, and its task is to "merge" all of them into one small circuit. Second, we build an $O(1)$-depth "tree" of Circuit-Min-Merge oracle gates, where each oracle gate in $\text{poly}(s(n))$ inputs (and outputs of "child" gates feed into the inputs of "parent" gates), whose leaves span the entire 2^n-length input, and where the output gate (the root of this tree) either prints a small circuit for the input or reports that none exists. Finally, applying the assumption $\mathbf{NP} \subset \mathbf{TC}^0$, we argue that inserting $\text{poly}(s(n))$-size $\mathbf{TC}^0$ circuits for Circuit-Min-Merge in this tree (in place of the oracle calls) would yield $2^n \cdot \text{poly}(s(n))$ size $\mathbf{TC}^0$ circuits for size-$s(n)$ MCSP on inputs of length 2^n.

1.3 What Do These Results Mean?

As is the case with other "amplification" and "magnification" results [Sri03, AK10, LW12, MP17, OS18], the results described in this paper have a strong "slippery slope" property: rather innocent-looking lower bounds on solving compression problems contained in a class C are in fact as hard as proving very strong complexity lower bounds for C.

It seems obligatory to ask: ***What are we to make of such theorems?*** Should we seriously consider these results as suggesting an approach towards major lower bounds such as $\mathbf{EXP} \not\subset \mathbf{TC}^0$ and even $\mathbf{P} \neq \mathbf{NP}$? Or, if we accept the somewhat popular style of argument that "if A implies B, and B is hard to prove, then A is hard to prove", should we believe we have found a new kind of barrier for proving innocent lower bounds? In all honesty, we are not sure. However, two comments seem significant.

(1) The aforementioned magnification and amplification results collected so far constitute a remarkable intuition-breaking phenomenon that demands closer attention. In the case of [OS18, OPS18] and this paper, the magnification results highlight the peculiar "weirdness" of MCSP, MKtP, and MKTP, showing how extremely weak-looking lower bounds for these problems turn out to already be as difficult as separating polynomially-strong complexity classes. In this regard, we believe such theorems to be noteworthy contributions towards a better understanding of these fundamental compression problems, and a better understanding of how close/far we are from proving major separations in complexity theory.

(2) It seems strange to call an implied consequence B a "barrier" to establishing A, when everyone expects B to be true (as is the case for all lower bound consequences in this paper). Certainly this is not the sort of barrier that arises with relativization [BGS75], natural proofs [RR97], and algebrization [AW09], where we either have unconditionally true claims (a relativizing/algebrizing proof *cannot* resolve $\mathbf{P}$ vs $\mathbf{NP}$) or implied consequences we do not believe to be true (a natural proof of $\mathbf{NP} \not\subset \mathbf{P}/\text{poly}$ implies there are no strong one-way functions). It is intriguing to wonder whether other impediments to complexity lower bounds (or algorithms) follow from the algorithms and circuits constructed in this paper. (Recall that our main results are not implications per se, but rather unconditional constructions of fast algorithms and circuits with short calls to $\Sigma_3 SAT$.)

2 PRELIMINARIES

We assume basic familiarity with computational complexity theory [AB09], especially circuit complexity and classes such as $\mathbf{AC}^0$, $\mathbf{ACC}^0$, $\mathbf{TC}^0$, and $\mathbf{NC}^1$. We assume these circuit classes are *non-uniform* unless otherwise specified. In this paper, an **AC circuit family** is any circuit family over the basis of NOT, unbounded fan-in OR, and unbounded fan-in AND.

Streaming Algorithms. We use a standard model for streaming algorithms. A *space-$s(n)$ streaming algorithm* with update (and reporting) time $u(n)$ on an input $x \in \{0,1\}^n$ has a working storage of $s(n)$ bits. At any point, the algorithm can either choose to perform one operation (from a standard palette of RAM operations) on $O(1)$ bits in storage, or it can choose to read the next bit x_i from the input (starting with x_1). The total time between two next-bit reads is at most $u(n)$, and the final output is reported in $O(u(n))$ time.

2.1 An Important Intermediate Problem

In Theorem 1.1 (in Section 3), we utilize an intermediate problem we call Circuit-Min-Merge^A, and show that Circuit-Min-Merge^A can be solved efficiently using $\Sigma_3\text{SAT}^A$ oracle gates. (A similar problem will be defined for Theorem 1.2, in Section 4.)

Problem: Circuit-Min-Merge$^A[s(n)]$
Given: Descriptions of A-oracle circuits $C_1, \ldots, C_t$ with n inputs and one output, a list of integers $(a_1, b_1), \ldots, (a_t, b_t) \in [2^n] \times [2^n]$ such that $a_i < b_i \leq a_{i+1}$ for all i.
Output: ***Either*** the string $1\langle C' \rangle$ where C' is the lexigraphically first minimum-size A-oracle circuit of size at most $s(n)$ such that for all $x \in \{0,1\}^n$ and all i, if $a_i \leq x < b_i$ then $C'(x) = C_i(x)$, ***or*** the all-0 string of length $100s(n)\log(s(n))$ when there is no such circuit.

(In the above, 100 is a placeholder for a sufficiently large constant c for encoding size $s(n)$ circuits in $cs(n)\log_2(s(n))$ size; certainly 100 suffices.) Note that any bit of the output string $1\langle C' \rangle$ can be computed by a Σ_3^A machine in $\tilde{O}(m)$ time for inputs of length m, by guessing and checking the (unique) circuit C' in the output specification, then verifying that every circuit C'' that comes before C' (with respect to the natural ordering) fails the specification. By a standard reduction (described further in the proof of Theorem 1.1), we can compute Circuit-Min-Merge$^A[s(n)]$ with $100s(n)\log(n)$ parallel queries to $\Sigma_3\text{SAT}^A$ (where each query computes a bit of the output). Because the queries can be done in parallel, we can then compute Circuit-Min-Merge$^A[s(n)]$ with either a low depth circuit with $\Sigma_3\mathrm{SAT}^A$ oracle gates, or $\Sigma_3\mathrm{SAT}^A$ oracle queries in a streaming algorithm.

3 EFFICIENT ORACLE CIRCUIT FAMILY FOR MCSP

In this section, we give an efficient circuit family for $\mathrm{MCSP}^A[s(n)]$ with low fan-in $\Sigma_3\mathrm{SAT}^A$ gates.

Reminder of Theorem 1.1. *Let $s(n) \geq n$ and let $\ell(n) \geq s(n)^2$ for all n, where both are time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. There is a uniform* **AC** *circuit family for $MCSP^A[s(n)]$ on 2^n-bit inputs of $\tilde{O}(2^n \cdot s(n)^2)$ size and $O(n/\log \ell(n))$ depth with $\Sigma_3 SAT^A$ oracle gates, where each oracle gate takes only $\tilde{O}(\ell(n))$ bits of input.*

To prove Theorem 1.1, we first show that there are small low-depth circuits using Circuit-Min-Merge^A oracle gates. (See the Preliminaries in Section 2.1 for the definition of Circuit-Min-Merge^A.)

Lemma 3.1. *Let $s(n) \geq n$ and let $\ell(n) \geq s(n)^2$ for all n, where both are time constructible. There is a uniform* **AC** *circuit family for $MCSP^A[s(n)]$ of $\tilde{O}(2^n \cdot s(n)/\ell(n))$ size and $O(n/\log \ell(n))$ depth with Circuit-Min-Merge$^A[s(n)]$ oracle gates, where each oracle gate takes only $\ell(n)$ bits of input.*

Proof. Suppose $s(n) \geq n$ and $\ell(n) \geq s(n)^2$ for all n, and both are time constructible. For the sake of clarity, we use the letter "G" to describe Circuit-Min-Merge^A gates used in the construction of our final circuit C_{2^n}, and the letter "D" to describe *bit-descriptions* of circuits. To simplify our description, we assume (without loss of generality) that there is no "integer roundoff", i.e., quantities like $\log_d(2^n)$ are integers.

Construction. Let T be an input of 2^n bits. To make the circuit C_{2^n}, we start by taking 2^n *descriptions* of n-input constant-size circuits $D_0^0, D_1^0, \ldots, D_{2^n-1}^0$ such that $D_x^0(y) = T(x)$ for all $x, y \in \{0,1\}^n$, each circuit encoding one bit of the input truth table T. Since these are constant-size circuits, the length of each descriptions D_i is at most $s(n)$ for almost all input lengths n.

Let $d = \ell(n)/(100s(n)\log s(n))$. Our circuit C_{2^n} will include a d-ary tree of Circuit-Min-Merge^A gates G_j^i of fan-in $\ell(n)$. At each layer, we divide the previous layer of circuits into blocks of d circuits; for each block of d circuits, we use one Circuit-Min-Merge$^A[s(n)]$ gate to combine the block into one output circuit.

To demonstrate, start with the bottom layer. Taking the constant-size circuits D_j^0 as input, we build a layer of $2^n/d$ circuits

$$G_0^1, \ldots, G_{(2^n/d)-1}^1,$$

where $G_j^1 = \text{Circuit-Min-Merge}^A[s(n)](D_{j\cdot d}^0, \ldots D_{(j+1)\cdot d}^0, (j\cdot d, j\cdot d+1), \ldots, ((j+1)\cdot d-1, (j+1)\cdot d))$. That is, each G_j^1 takes a contiguous block of d descriptions from the bottom layer, and outputs one circuit of size at most $s(n)$ consistent with these d circuits (or reports that no such circuit exists).

We repeat this process on the *descriptions output by the circuits* G_j^1: at each new layer, we divide the previous layer of circuits into blocks of d circuits, using one Circuit-Min-Merge$^A[s(n)]$ gate for each block. That is, for $i \in \{1, \ldots, \log_d(2^n)\}$ at layer i, the Circuit-Min-Merge$^A[s(n)]$ gate G_j^i takes as input circuit descriptions $D_{j\cdot d}^{i-1}, \ldots, D_{(j+1)d-1}^{i-1}$ along with the ordered pairs

$$(jd^i, jd^i+d^{i-1}), (jd^i+d^{i-1}, jd^i+2d^{i-1}), \ldots, ((j+1)d^i-d^{i-1}, (j+1)d^i)$$

and outputs the description of a new circuit D_j^i.

Finally, to get a circuit for MCSP, the output of the circuit C_{2^n} is the AND of the first output bit of each G_j^i in the circuit. To get a circuit for search-MCSP, we can AND the output gate of C_{2^n} with each bit of $G_0^{\log_d(2^n)}$. This circuit either prints the all-zeroes string when no circuit exists, or prints a size $s(n)$ circuit for the entire input.

Correctness. To prove that C_n computes $\mathrm{MCSP}^A[s(n)]$ on 2^n-bit inputs, we will first prove by induction that, assuming small circuits exist for the input truth table T, the circuit D_j^i output by gate G_j^i matches the input on bits jd^i through $(j+1)d^i - 1$. By construction, the $O(1)$-size circuit D_j^0 matches T on bit j for every j. For $i > 0$, suppose that all circuits D_j^{i-1} match T on bits jd^{i-1} through $(j+1)d^{i-1} - 1$. The circuit D_j^i is the output of $G_j^i = \text{Circuit-Min-Merge}^A[s(n)]$ on circuits $D_{j\cdot d}^{i-1}, \ldots, D_{(j+1)d-1}^{i-1}$ with ordered pairs

$$(jd^i, jd^i+d^{i-1}), (jd^i+d^{i-1}, jd^i+2d^{i-1}), \ldots, ((j+1)d^i-d^{i-1}, (j+1)d^i).$$

This means that D_j^i matches D_{jd}^{i-1} on bits jd^i through $jd^i + d^{i-1} - 1$, and since D_{jd}^{i-1} matches T on bits $(jd)d^{i-1} = jd^i$ through $(jd+1)d^{i-1} - 1 = jd^i + d^{i-1} - 1$, we get that D_j^i matches T on these bits as well. Similarly, the bits on which D_j^i are forced to match a D_*^{i-1} circuit are exactly those bits on which the D_*^{i-1} circuit matches T, which means that the final circuit D_j^i matches T on all bits covered by D_{jd}^{i-1} through $D_{(j+1)d-1}^{i-1}$, which is jd^i through $(j+1)d^i - 1$, completing the induction.

So for all i, j, the output of G_j^i is then the description of a circuit D_j^i of size at most $s(n)$ whose truth table matches the desired truth table T on bits $j \cdot d^i$ through $(j+1) \cdot d^i - 1$ (or $0^{100s(n)\log s(n)}$ if such a circuit does not exist).

The final AND gate takes the first output bit of each of these G_j^i gates. We claim that all of these bits are 1 if and only if the input truth table T has a circuit of size at most $s(n)$. To prove this, we need only show

(1) Each gate G_j^i and G' outputs 1 as its first bit if T has a circuit of size at most $s(n)$.
(2) Some G_j^i or G' outputs 0 as its first bit if T does not have a circuit of size at most $s(n)$.

Suppose T has an A-oracle circuit of size at most $s(n)$. Then this oracle circuit serves as a witness for every Circuit-Min-Merge^A computation in the circuit; by construction, gate G_j^i outputs a circuit consistent with T on bits jd^i through $(j+1)d^i$ if such a circuit exists. However, the $s(n)$ size circuit which computes T will obviously be consistent with T on these bits (as well as every other bit of T), which means that the minimum circuit must have at most $s(n)$ gates. The minimum circuit may in fact be smaller (which is likely the case for the constant circuits D_j^0), but can never be larger than $s(n)$.

So if all circuits G_j^i at layer i output a circuit D_j^i which is of size at most $s(n)$, then every circuit G_j^{i+1} at layer $i+1$ will output a circuit D_j^{i+1} which is of size at most $s(n)$. By induction, for all i, j, G_j^i will output a valid circuit, with 1 as its first bit, which means that the AND gate at the top will output 1 as well.

Now suppose that T has no A-oracle circuit of size $s(n)$. Then there should be some place in the circuit where Circuit-Min-Merge^A fails to combine the intervals into a single small circuit. Start at the top Circuit-Min-Merge^A gate $G_0^{\log_d(2^n)}$. If all d inputs to this gate are valid circuits, then the output of this circuit must still be $0^{100 s(n)\log s(n)}$, since the output must be a size $s(n)$ circuit which matches T on all input bits, and we assumed that such a circuit does not exist. On the other hand, if one of the inputs to the gate is not a valid circuit then we can move to layer $\log_d(2^n) - 1$ and repeat this argument. Each time, either all inputs are valid circuits and the output is $0^{100 s(n)\log s(n)}$ (the circuit does not exist for this interval), or we can recurse on one of the invalid inputs. Since there are constant-size circuits at the bottom, there must be some gate which outputs $0^{100 s(n)\log s(n)}$ with valid circuits as input, which will force the AND to output 0 as well.

Recall the circuit C_n is a d-ary tree of Circuit-Min-Merge^A circuits with a single AND gate at the top. This gives a total depth of $\log_d(2^n) + 1 = n/\log_2(d) + 1 = n/(\log \ell(n) - \log s(n)) + 1 = O(n/\log \ell(n))$, and a size bound of $O(2^n/d)$ gates, which is a little suboptimal. To get the optimal bound, we increase the number of circuits fed into the Circuit-Min-Merge^A circuits at the bottom layer to $O(\ell(n)/n)$, and maintain a fan-in of $\ell(n)$ to each Circuit-Min-Merge^A oracle gate. With this, the size of the circuit can be reduced to $O(2^n \cdot n/\ell(n)) \leq \tilde{O}(2^n/\ell(n))$. □

For example, for any $\varepsilon > 0$, if $s(n)$ is at most $O(2^{(1-\varepsilon)n/2})$, then we can let $\ell(n) = 2^{(1-\varepsilon)n}$ and our circuit will have constant depth.

Finally, we explain why Theorem 1.1 follows (with the decision problem Σ_3SAT^A in place of Circuit-Min-Merge^A). First, recall that Circuit-Min-Merge^A is a function problem, outputting $\tilde{O}(s(n))$ bits, where each output bit is computable by a Σ_3 machine (making Σ_3-style alternations) in $\tilde{O}(n)$ time. By standard completeness results (see for example, in [FLvMV05]) there is a simple $\mathbf{AC}^0$ reduction of size $\tilde{O}(\ell(n)\cdot s(n))$ from the Circuit-Min-Merge^A problem to $\tilde{O}(s(n))$ copies of Σ_3SAT^A: we can directly map Circuit-Min-Merge^A instances with $\ell(n) > s(n)^2$ inputs and $\tilde{O}(s(n))$ outputs to $\tilde{O}(s(n))$ instances of Σ_3SAT^A where each instance has length $\tilde{O}(\ell(n))$. Hence each Circuit-Min-Merge^A oracle gate can be replaced by $\tilde{O}(\ell(n)\cdot s(n))$ extra gates plus $\tilde{O}(s(n))$ Σ_3SAT^A gates of fan-in $\tilde{O}(\ell(n))$. The size bound of Theorem 1.1 follows, and the replacement does not affect the depth of our **AC** circuit by more than a constant factor.

3.1 Other Compression Problems

Other compression problems similar to MCSP are amenable to the same kind of circuit construction as Theorem 1.1. The most obvious example is MKTP$^A[s(n)]$, which has an oracle circuit construction that is nearly identical to that of Theorem 1.1 using instead oracle calls to the following intermediate problem which can (as before) be reduced to Σ_3SAT^A.

Problem: KT-Min-Merge$^A[p(N)]$
Given: Descriptions of A-oracle machines $M_1, \ldots, M_s$ with $\lceil \log(n) \rceil$ inputs and one output, a list of integers $(a_1, b_1), \ldots, (a_s, b_s) \in [N] \times [N]$ such that $a_i < b_i \leq a_{i+1}$ for all i.
Output: ***Either*** a string $1\langle M' \rangle$ where M' is the lexicographically first A-oracle Machine M' of minimum KT complexity such that for all $x \in [n]$ and all i, if $a_i \leq x < b_i$ then $M'(x) = M_i(x)$ ***or*** a string of length $p(N)$ containing only the character 0 when there is no such machine.

Using this, we get a theorem parallel to Theorem 1.1, replacing MCSP^A with MKTP^A.

Theorem 3.2. *Let $p(N) \geq \log(N)$ and let $\ell(N) \geq p(N)^2$ for all N, where both are time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. There is a uniform* **AC** *circuit family for $MKTP^A[p(N)]$ of $\tilde{O}(N \cdot p(N)^2)$ size and $O(\log(N)/\log \ell(\log(N)))$ depth with $\Sigma_3 SAT^A$ oracle gates, where each oracle gate takes only $\ell(\log(N)) \cdot poly(\log\log(N))$ bits of input.*

We see that there is even an analogous circuit computing the Kolmogorov complexity of a string, though this case is much less interesting as we already know that the corresponding oracle gates will be for languages which are not computable. Another more interesting example is MKtP^A, which is constructed again as above but using oracles for the following intermediate problem.

Problem: Kt-Min-Merge$[p(N)]$
Given: A-oracle machines $M_1, \ldots, M_s$ with N inputs and one output, a list of integers $(a_1, b_1), \ldots, (a_s, b_s) \in [N] \times [N]$ such that $a_i < b_i \leq a_{i+1}$ for all i.
Output: ***Either*** a string $1\langle M' \rangle$ where M' is the lexicographically first A-oracle machine M' of minimum Kt complexity such that for all $x \in [N]$ and all i, if $a_i \leq x < b_i$ then $M'(x) = M_i(x)$ ***or*** a string of length $p(N)$ containing only the character 0 when there is no such machine.

Observing that Kt-Min-Merge can naively be computed in time $2^{O(s(n))}$, we again find an analogous circuit for MKtP.

Theorem 3.3. *Let $p(N) \geq \log(N)$ and let $\ell(N) \geq p(N)^2$ for all N, where both are time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. There is a language $B \in$* **TIME**$[2^{O(p(N))}]^A$ *such that there is a uniform* **AC** *circuit family for $MKtP^A[p(N)]$ of $\tilde{O}(N \cdot p(N)^2)$ size and $O(\log(N)/\log \ell(\log(N)))$ depth with B oracle gates, where each oracle gate takes only $\ell(\log(N)) \cdot poly(\log\log(N))$ bits of input.*

Note that the parameters of the circuits have to be changed in a straightforward way as an instance of MCSP^A has an input length of 2^n, while the input lengths to our other compression problems is simply N.

4 STREAMING ALGORITHM FOR MCSP

We now turn to the streaming algorithm for search-MCSP.

Reminder of Theorem 1.2. *Let $s(n) \geq n$ for all n, where $s(n)$ is time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle.*

There is a (one-pass) streaming algorithm for search-$MCSP^A[s(n)]$ on 2^n-bit inputs running in $2^n \cdot \tilde{O}(s(n))$ time with $\tilde{O}(s(n)^2)$ update time and $\tilde{O}(s(n))$ space, using an oracle for $\Sigma_3 SAT^A$ with queries of length $\tilde{O}(s(n))$.

Here we present an oracle reduction from the $MCSP^A[s(n)]$ problem to a problem related to Circuit-Min-Merge$^A[s(n)]$.

Let $x_1, \ldots, x_{2^n}$ be the list of n-bit strings in lexicographical order.

Problem: Stream-Merge$^A[s(n)]$
Given: A t-bit description of an A-oracle circuit C with n inputs and one output where $t = 100s(n) \log s(n)$, an $x_i \in \{0,1\}^n$, and a t-bit string $y = y_1 \cdots y_t$.
Output: ***Either*** the string $1\,\langle C' \rangle$ where C' is the lexigraphically first minimum-size A-oracle circuit of size at most $s(n)$ such that for all strings $x < x_i$ we have $C'(x) = C(x)$, and for all $j = 1, \ldots, t$, $C'(x_{i+j-1}) = y_j$, ***or*** the all-0 string of length t when there is no such circuit C'.

In other words, Stream-Merge takes a circuit C as input, an input x_i, and t extra bits of a truth table, and tries to output a small circuit that agrees with C on all inputs less than x_i, and agrees with the t extra bits on the t inputs $x_i, \ldots, x_{i+t-1}$. As with Circuit-Min-Merge^A in Section 2.1 and Theorem 1.1, Stream-Merge$^A[s(n)]$ can be efficiently reduced to Σ_3SAT^A queries of length $\tilde{O}(s(n))$.

Proof. Algorithm 1 presents the description of the streaming algorithm. Let $t = 100s(n) \log s(n)$. We start by building a circuit C_0 of size at most $s(n)$ that is consistent with the first t bits of the given truth table T, using $O(s(n) \log s(n))$-length queries to the Stream-Merge^A oracle. We repeat this process for each successive block of t bits of the input, attempting to generate a new circuit C_{i+1} which is consistent with both C_i and the new block of the input. If at any step we fail to generate a circuit C_i, we report that there is no circuit of size $s(n)$. Otherwise, at the end we print a circuit $C_{2^n/t}$ whose truth table is the input.

Algorithm 1 The Streaming Algorithm

```
Given: a truth table $T$ of size $2^n = N$.
Let $C$ be a trivial circuit for the constant function 0 on $n$ bits.
for $i \leftarrow 1, \ldots, 2^n/t$ do
    Let $y_1, \ldots, y_t$ be the next $t$ bits of $T$.
    $C \leftarrow$ Circuit-Min-Merge$^A[s(n)](C, x_{(i-1)\cdot t+1}, y_1 \cdots y_t)$.
    if $C$ is not a valid circuit, then report that there is no $A$-oracle circuit of size $s(n)$.
Report $C$ as an $A$-oracle circuit of size at most $s(n)$ computing $T$.
```

The proof that this algorithm computes $MCSP^A$ is similar to the proof of correctness for the circuit of Theorem 1.1. The correctness of Algorithm 1 follows by induction. If T has an A-oracle circuit C of size $s(n)$, then C can serve as a witness for every Stream-Merge^A query in the algorithm: in iteration i of the algorithm, if C_{i-1} is a circuit of size $s(n)$ consistent with T on the first $(i-1) \cdot t$ bits, then the Stream-Merge^A call outputs a circuit C_i consistent with T on the first $i \cdot t$ bits, if such a circuit exists. A circuit C of $s(n)$-size for T will be consistent with T on these bits for any i, which means the minimum circuit C_{i+1} output by Stream-Merge^A has size at most $s(n)$. By induction, if T has an A-oracle circuit of size at most $s(n)$, then Algorithm 1 will produce this circuit and report that there is such a circuit.

On the other hand, if there is no such circuit for T, then there is some $i = 1, \ldots, 2^n/t$ such that there is a size-$s(n)$ circuit consistent with T on its first $(i-1) \cdot t$ bits, but there is no size-$s(n)$ circuit consistent with T on its first $i \cdot t$ bits. (Such a circuit exists trivially for $i = 1$, but by assumption no such circuit exists for $i = 2^n/t$.) Let i' be the first such index. For all iterations before this i', Stream-Merge^A successfully outputs a circuit of size $s(n)$, but in iteration i' Stream-Merge^A fails to find such a circuit, and thus outputs $0^{100s(n)\log s(n)}$. The algorithm will notice this output is not a valid circuit, and report there is no A oracle circuit of size $s(n)$.

It remains to show that Algorithm 1 uses small time and space. In each iteration of the for-loop, we read t more bits of T, query the Stream-Merge^A oracle, and check that the t bits of query answer encodes a valid circuit. Each of these tasks can be done in $O(t)$ time. Therefore each iteration can be done in $O(t)$ time and space. Over $2^n/t$ iterations, the overall resource consumption is $O(2^n)$ time and $O(t) \leq O(s(n) \log s(n))$ space, when we have access to a Stream-Merge^A oracle.

When we only have access to a Σ_3SAT^A oracle instead, each Stream-Merge^A call of length t can be converted into $O(t)$ sequential calls of length $\tilde{O}(t)$ to Σ_3SAT^A, computable in $\tilde{O}(t)$ time each. Thus when we have a Σ_3SAT^A oracle, the running time is $2^n \cdot \tilde{O}(t)$, the worst-case update time is $\tilde{O}(t^2)$ (between the reading of a bit from T in one iteration to the next, we have to make $O(t)$ calls of length $\tilde{O}(t)$), and the space usage is $\tilde{O}(t)$. □

Note the proof of Theorem 1.2 yields the following somewhat tighter result: a linear-time streaming algorithm with an appropriate oracle B.

Theorem 4.1. *Let $s(n) \geq n$ for all n, where $s(n)$ is time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. Let $t(n) = 200s(n) \log s(n)$. There is an oracle $B_n : \{0,1\}^{t(n)} \to \{0,1\}^{t(n)}$ whose output bits are computable in* **PH**, *and a (one-pass) streaming algorithm for search-$MCSP^A[s(n)]$ on 2^n-bit inputs running in $O(2^n)$ time and $O(t(n))$ space, using an oracle for B_n with queries of length $t(n)$.*

4.1 Other Compression Problems

Theorem 1.2 readily applies to compression problems other than MCSP. For example, to model MKTP, we simply have to modify the definition of Stream-Merge in Theorem 1.2 so that, rather than taking as input a size-$s(n)$ circuit C representing the initial segment of a string, we take in a Turing machine M with KT complexity at most $p(N)$ (as in KT-Min-Merge in Theorem 3.2). Thus we can conclude Theorem 4.2:

Theorem 4.2. *Let $p(N) \geq \log(N)$ for all n, where $p(N)$ is time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. There is a streaming algorithm for $MKTP^A[p(N)]$ running in $O(N \cdot p(N))$ time and $O(p(N))$ space, using an oracle for $\Sigma_3 SAT^A$ with queries of length at most $O(p(N))$.*

Similarly, if we modify the definition of Stream-Merge so that the size-$s(n)$ circuit C is replaced by a Turing machine M with Kt complexity at most $p(N)$ (as in Kt-Min-Merge in Theorem 3.3), we obtain a similar streaming algorithm for $\mathrm{MKtP}^A[p(N)]$.

Theorem 4.3. *Let $p(N) \geq \log(N)$ for all N, where $p(N)$ is time constructible. Let $A : \{0,1\}^\star \to \{0,1\}$ be an arbitrary oracle. There is a streaming algorithm for $MKtP^A[p(N)]$ running in $O(N \cdot p(N))$ time and $O(p(N))$ space, using an oracle for $\mathbf{TIME}[2^{O(p(N))}]^A$ with queries of length at most $O(p(N))$.*

As in the case of the oracle circuit (Theorem 3.3), the parameters of Algorithm 1 have to be changed in a straightforward way to accommodate the fact that we have described the instances of MCSP^A as length 2^n strings, while we are describing instances of other compression problems as length n strings.

5 CONSEQUENCES

We now present some consequences of the oracle streaming algorithm of Section 4 and the oracle circuits constructed in Section 3.

Reminder of Theorem 1.3. *Let $s(n) \geq n$ be time constructible. If there is an $A \in \mathbf{PH}$ such that search-$MCSP^A[s(n)]$ is not solvable by a poly$(s(n))$-space streaming algorithm with poly$(s(n))$ update time, then $\mathbf{P} \neq \mathbf{NP}$.*

Proof. We show that if $\mathbf{P} = \mathbf{NP}$, then such a streaming algorithm for $\mathrm{MCSP}^A[s(n)]$ exists for all $A \in \mathbf{PH}$.

Suppose $\mathbf{P} = \mathbf{NP}$. Then the entire polynomial hierarchy collapses to $\mathbf{P}$, which means that for any possible $A \in \mathbf{PH}$, $\Sigma_3\mathrm{SAT}^A$ (and by extension Circuit-Min-Merge^A) can be solved in polynomial time. Taking our streaming algorithm from Theorem 1.2, every query to Circuit-Min-Merge$^A[s(n)]$ can be replaced by some poly$(s(n))$ time computation (all queries have length $O(s(n)\log s(n))$). As a result, we obtain (for any $A \in \mathbf{PH}$) a poly$(s(n))$-space streaming algorithm that takes poly$(s(n))$ update time, completing the proof. □

Reminder of Theorem 1.4. *Let $s(n) \geq n$, and let $A \in \mathbf{PH}$.*

- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, search-$MCSP^A[2^{\varepsilon n/c}]$ on inputs of length $N = 2^n$ does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth $\mathbf{TC}^0$ circuits, then $\mathbf{NP} \not\subset \mathbf{TC}^0$.*[5]
- *If search-$MCSP^A[s(n)]$ on inputs of length $N = 2^n$ does not have circuits of $N \cdot$ poly$(s(n))$ size and $O(\log N)$ depth, then $\mathbf{NP} \not\subset \mathbf{NC}^1$.*
- *If search-$MCSP^A[s(n)]$ on inputs of length $N = 2^n$ does not have circuits of $N \cdot$ poly$(s(n))$ size and poly$(s(n))$ depth, then $\mathbf{NP} \not\subset \mathbf{P}/poly$.*

Proof. Again, we prove these results by contrapositive. Since all the above circuit classes are closed under complement, if $\mathbf{NP} \subset C$ then the entire polynomial hierarchy collapses to C, including the Circuit-Min-Merge$^A[s(n)]$ problem. As a result, each copy of the Circuit-Min-Merge gate in the circuit constructed in Lemma 3.1 can be replaced with a C-circuit of poly$(\ell(n))$ size (where $\ell(n)$ is the length of the queries to Circuit-Min-Merge). The proper $s(n)$ or $\ell(n)$ will then yield a circuit of the desired size and depth computing $\mathrm{MCSP}^A[s(n)]$.

- Let $C = \mathbf{TC}^0$. For $s(n) = N^{\varepsilon/c}$ and $\ell(n) = N^{2\varepsilon/c}$, Lemma 3.1 gives a circuit of size $O(N^{1+\varepsilon/c})$ and depth $O(\log N / \log N^{2\varepsilon/c}) = O(c/\varepsilon)$; replacing the oracle gates with poly$(N^{2\varepsilon/c})$ size $\mathbf{TC}^0$ circuits gives a circuit of size $O(N \cdot \mathrm{poly}(N^{\varepsilon/c}))$ and depth $O(c/\varepsilon)$. Since this circuit exists for every $c \geq 1$, set c large enough to obtain a $\mathbf{TC}^0$ circuit of $N^{1+\varepsilon}$-size and $O(1/\varepsilon)$-depth.
- For $C = \mathbf{NC}^1$ and $\ell(n) = s(n)^3$, Lemma 3.1 gives an $O(N)$-size circuit of depth $O(\log N / \log s(n))$. Replacing the oracle gates with poly$(s(n))$ size $\mathbf{NC}^1$ circuits gives a circuit of size $N \cdot \mathrm{poly}(s(n))$ and depth $O((\log N/\log s(n)) \cdot \log s(n)) = O(\log N)$.
- Let $C = \mathbf{P}/poly$. For $\ell(n) = s(n)^3$, Lemma 3.1 again gives a circuit of size $O(N)$ and depth $O(\log N/\log s(n))$. Replacing the oracle gates with poly$(s(n))$ size circuits gives a circuit of size $N \cdot$ poly$(s(n))$ and depth $O((\log N/\log s(n)) \cdot \mathrm{poly}(s(n))) = \mathrm{poly}(s(n))$.

This completes the proof. □

Now we turn to proving the hardness magnification consequences for (harder) oracle versions of MCSP.

Reminder of Theorem 1.5. *[Magnifying Streaming Lower Bounds for Harder MCSP Versions] Let C be in $\{\mathbf{NP}, \mathbf{PP}, \oplus\mathbf{P}, \mathbf{PSPACE}\}$. Suppose there is a constructible $s(n)$ and oracle $A \in C$ such that for all $c \geq 1$, search-$MCSP^A[s(n)]$ on inputs of length 2^n has no $s(n)^c$-space streaming algorithm with update time $s(n)^c$. Then $\mathbf{P} \neq C$.*

Proof. Suppose $\mathbf{P} = C$. To prove the contrapositive, we wish to show that for all constructible functions $s(n)$ and oracles $A \in C$, there exists a $c \geq 1$ such that $\mathrm{MCSP}^A[s(n)]$ has an $s(n)^c$-space streaming algorithm with update time $s(n)^c$.

Since $\mathbf{NP} \subseteq C$, we know that $\mathbf{P} = \mathbf{NP}$, so again the polynomial hierarchy collapses to $\mathbf{P}$, and since $A \in \mathbf{P}$ as well Circuit-Min-Merge^A can be solved in polynomial time. Similar to Theorem 1.3, we can replace the oracle in the streaming algorithm with a deterministic algorithm running in $s(n)^c$ time for some constant c on inputs of length $O(s(n)\log s(n))$. So Theorem 1.2 gives us an $s(n)^c$-space streaming algorithm with update time $s(n)^c$. □

Reminder of Theorem 1.6. *[Magnifying Low-Depth Circuit Lower Bounds for Harder MCSP] Let C be in $\{\mathbf{NP}, \mathbf{PP}, \oplus\mathbf{P}, \mathbf{PSPACE}, \mathbf{EXP}\}$. Suppose there is some $s(n)$ and oracle $A \in C$ such that for all $c \geq 1$, search-$MCSP^A[s(n)]$ on inputs of length 2^n has no circuits of depth $O(n)$ and $2^n \cdot s(n)^c$ size. Then C does not have polynomial-size formulas (i.e., $C \not\subset \mathbf{NC}^1$).*

Proof. Suppose C has polynomial-size formulas. To prove the contrapositive, we construct $MCSP^A[s(n)]$ circuits of depth $O(n)$ and $2^n \cdot \mathrm{poly}(s(n))$ size.

Since $\mathbf{NP} \subseteq C$, we can construct polynomial-size formulas for any problem in $\mathbf{PH}^A$, including Circuit-Min-Merge^A. Take the oracle circuit family constructed in Lemma 3.1 of $O(2^n \cdot s(n)/\ell(n))$ size and $O(n/\log \ell(n))$ depth. Replace each Circuit-Min-Merge^A gate with a poly$(\ell(n))$ size formula, which will blow up both the size and depth, but not by much. There are $O(2^n)$ copies of this Circuit-Min-Merge^A formula, so the size of the resulting circuit is about $O(2^n \cdot \mathrm{poly}(\ell(n))$. Each of these formulas will have depth $O(\log(\ell(n)))$ as well, which will increase the depth of the circuit to $O(n/\log \ell(n) \cdot \log \ell(n)) = O(n)$. So if we set $\ell(n) = \mathrm{poly}(s(n))$, we

[5] Note that $\mathbf{TC}^0$ could be substituted with any other constant-depth circuit family, such as $\mathbf{AC}^0$[6].

obtain a circuit of depth $O(n)$ and size $2^n \cdot \text{poly}(s(n))$ size computing $MCSP^A[s(n)]$ on 2^n-bit inputs, which completes the proof. □

5.1 Other Compression Problems

Most of the above theorems are consequences of the existence of the circuits and streaming algorithm as given by Theorems 1.1 and 1.2. Because these circuits and streaming algorithms exist for MKTP and MKtP as per Theorems 3.2, 3.3, 4.2, and 4.3, we can conclude analogues of many of the same results as above.

Reminder of Theorem 1.7. *[Consequences for MKTP and MKtP] Let $p(N) \geq \log(N)$ be time constructible.*

- *If MKTP$[p(N)]$ is not solvable by a poly$(p(N))$-space streaming algorithm with poly$(p(N))$ update time, then* $\mathbf{P} \neq \mathbf{NP}$.
- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, MKTP$[N^{\varepsilon/c}]$ does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth* $\mathbf{TC}^0$ *circuits, then* $\mathbf{NP} \not\subset \mathbf{TC}^0$.
- *If MKTP$[p(N)]$ does not have $N \cdot$ poly$(p(N))$-size $O(\log(N))$-depth circuits, then* $\mathbf{NP} \not\subset \mathbf{NC}^1$.
- *If MKTP$[p(N)]$ does not have $N \cdot$ poly$(p(N))$-size poly$(p(N))$-depth circuits, then* $\mathbf{NP} \not\subset \mathbf{P}/poly$.
- *If MKtP$[p(N)]$ does not have $N \cdot$ poly$(p(N))$-size poly$(p(N))$-depth circuits, then* $\mathbf{EXP} \not\subset \mathbf{P}/poly$.
- *If MKtP$[p(N)]$ does not have $N \cdot$ poly$(p(N))$-size $O(\log(N))$-depth circuits, then* $\mathbf{EXP} \not\subset \mathbf{NC}^1$.
- *If there is an $\varepsilon > 0$ such that for all $c \geq 1$, MKtP$[N^{\varepsilon/c}]$ does not have $N^{1+\varepsilon}$-size $O(1/\varepsilon)$-depth* $\mathbf{TC}^0$ *circuits, then* $\mathbf{EXP} \not\subset \mathbf{TC}^0$.

We omit the proof of Theorem 1.7, as each claim follows easily from the arguments given above for the analogous claims about MCSP^A in Theorems 1.4, 1.5, and 1.6 citing instead Theorems 3.2, 3.3, 4.2, and 4.3 for the existence of efficient oracle circuits and streaming algorithms for MKTP and MKtP. Relativized versions of Theorem 1.7 also hold for MKTP^A and MKtP^A with an oracle A; we leave the details to the interested reader.

6 CONCLUSION

We conclude with a few related open problems.

- **What is the moral of this story?** What lessons do we draw from these results? As we stated in the introduction, it does not seem right to call our results a "barrier" to proving weak time-space lower bounds, because we believe all of the consequential lower bounds of this paper! Still, there ought to be more consequences of the fact that certain "weak-looking" lower bound problems are deceptive, and in fact are much stronger than they appear.
- **Sparse Problems?** We have attributed the "hardness magnification" results of this paper to the "weirdness" of problems like MCSP and MKTP. One wonders whether these results are due not to some inherent weirdness, but rather the simple fact that MCSP$[s(n)]$ has only $s(n)^{O(s(n))}$ YES-instances: that is, the problem is a *sparse* language when $s(n)$ is relatively small. However, as far as we can tell, sparsity does not seem to be enough to yield such results: the compressibility of YES-instances also seems crucial. Studying what happens when sparse problems in **NP** have small circuits is an interesting next step.
- **Bounded Nondeterminism Problems?** Another key property of MCSP$[s(n)]$ (and related compression problems) is that the nondeterminism needed to solve the problem is only $O(s(n) \log s(n))$ bits: the number of bits needed to write down a circuit of size $s(n)$. When $s(n) \leq 2^{o(n)}$, MCSP$[s(n)]$ is a problem with *sub-linear nondeterminism.* It is natural to ask whether hardness magnification holds for similar problems. For example, consider the SAT problem with $s(n)$ variables and 2^n clauses. Are there interesting consequences of proving weak time lower bounds for such SAT problems, for algorithms using poly$(s(n))$ space?
- **Truth tables presented differently?** Our main results for MCSP rely on the input truth table of f being presented in a canonical way, namely as a 2^n-bit string
$$f(x_1)\cdots f(x_{2^n}),$$
where $x_1, \ldots, x_{2^n}$ is the list of n-bit strings in lexicographical order. Our results can also extend to the case of other efficiently-computable orderings on strings. What about when the truth table is presented in an *arbitrary* order, as a list of pairs
$$(x_1, f(x_1)), \ldots, (x_{2^n}, f(x_{2^n}))$$
where $x_1, \ldots, x_{2^n}$ is an arbitrary permutation of $\{0,1\}^n$? Can similar hardness magnification results be proved for this version of the problem? Note that, in this representation, each Boolean function corresponds to $(2^n)!$ distinct strings of length $\Theta(n2^n)$, so the underlying language MCSP$[s(n)]$ is no longer as sparse as it used to be.

Acknowledgements. We thank Josh Alman for helpful proofreading, Igor Carboni Oliveira for useful discussions on the literature and many careful comments, the STOC reviewers for helpful comments, and the Princeton Theory Lunch crowd for useful feedback on the results.

REFERENCES

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[ABK+06] Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35(6):1467–1493, 2006. Preliminary version in FOCS'02.

[AH17] Eric Allender and Shuichi Hirahara. New insights on the (non-)hardness of circuit minimization and related problems. In *MFCS*, pages 54:1–54:14, 2017.

[AHK17] Eric Allender, Dhiraj Holden, and Valentine Kabanets. The minimum oracle circuit size problem. *Computational Complexity*, 26(2):469–496, 2017. Preliminary version in STACS'15.

[Ajt02] Miklós Ajtai. Determinism versus nondeterminism for linear time rams with memory restrictions. *Journal of Computer and System Sciences*, 65(1):2–37, 2002.

[Ajt05] Miklós Ajtai. A non-linear time lower bound for boolean branching programs. *Theory of Computing*, 1(8):149–176, 2005.

[AK10] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *JACM*, 57(3), 2010.

[All01] Eric Allender. When worlds collide: Derandomization, lower bounds, and kolmogorov complexity. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science*, 2001.

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *TOCT*, 1(1):2:1–2:54, 2009.

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

[BJS01] Paul Beame, Thathachar S Jayram, and Michael Saks. Time-space tradeoffs for branching programs. *Journal of Computer and System Sciences*, 63(4):542–572, 2001.

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

[BW15] Samuel R. Buss and Ryan Williams. Limits on alternation trading proofs for time-space lower bounds. *Computational Complexity*, 24(3):533–600, 2015.

[Cho11] Timothy Y. Chow. Almost-natural proofs. *J. Comput. Syst. Sci.*, 77(4):728–737, 2011.

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

[Hir18] Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:138, 2018.

[HOS18] Shuichi Hirahara, Igor Carboni Oliveira, and Rahul Santhanam. Np-hardness of minimum circuit size problem for OR-AND-MOD circuits. In *CCC*, pages 5:1–5:31, 2018.

[HP15] John M. Hitchcock and Aduri Pavan. On the NP-completeness of the minimum circuit size problem. In *35th IARCS Conf. Found. of Software Tech. and Theoret. Comput. Sci. (FSTTCS'15)*, volume 45 of *LIPIcs*, pages 236–245, 2015.

[HS17] Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In *CCC*, pages 7:1–7:20, 2017.

[HW16] Shuichi Hirahara and Osamu Watanabe. Limits of minimum circuit size problem as oracle. In *CCC*, volume 50, pages 18:1–18:20, 2016. Available at ECCC.

[IKV18] Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. In *CCC*, pages 7:1–7:20, 2018.

[KC00] Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *STOC*, pages 73–79, 2000.

[Lev84] Leonid A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61(1):15–37, 1984.

[LW12] Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time with applications. In *CCC*, pages 1–9, 2012.

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

[MW15] Cody D. Murray and R. Ryan Williams. On the (non) NP-hardness of computing circuit complexity. In *CCC*, volume 33 of *LIPIcs*, pages 365–380. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[OPS18] Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:158, 2018.

[OS18] Igor Carboni Oliveira and Rahul Santhanam. Hardness magnification for natural problems. In *FOCS*, 2018. Available at ECCC.

[RR97] Alexander Razborov and Steven Rudich. Natural proofs. *JCSS*, 55(1):24–35, 1997.

[Sri03] Aravind Srinivasan. On the approximability of clique and related maximization problems. *Journal of Computer and System Sciences*, 67(3):633 – 651, 2003.

[Tra84] B. A. Trakhtenbrot. A survey of russian approaches to perebor (brute-force searches) algorithms. *Annals of the History of Computing*, 6(4):384–400, Oct 1984.

[Wil08] R. Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. *Computational Complexity*, 17(2):179–219, 2008.