New Insights on the (Non-)Hardness of Circuit Minimization and Related Problems

ERIC ALLENDER, Rutgers University, USA
SHUICHI HIRAHARA, National Institute of Informatics, JAPAN

The Minimum Circuit Size Problem (MCSP) and a related problem (MKTP) that deals with time-bounded Kolmogorov complexity are prominent candidates for NP-intermediate status. We show that, under very modest cryptographic assumptions (such as the existence of one-way functions), the problem of approximating the minimum circuit size (or time-bounded Kolmogorov complexity) within a factor of $n^{1-o(1)}$ is indeed NP-intermediate. To the best of our knowledge, these problems are the first natural NP-intermediate problems under the existence of an arbitrary one-way function. Our technique is quite general; we use it also to show that approximating the size of the largest clique in a graph within a factor of $n^{1-o(1)}$ is also NP-intermediate unless \( \text{NP} \subseteq \text{P/poly} \).

We also prove that MKTP is hard for the complexity class DET under non-uniform NC\(^0\) reductions. This is surprising, since prior work on MCSP and MKTP had highlighted weaknesses of "local" reductions such as \( \leq_{\text{in}} \). We exploit this local reduction to obtain several new consequences:

- MKTP is not in AC\(^0\)[p].
- Circuit size lower bounds are equivalent to hardness of a relativized version MKTP\(^A\) of MKTP under a class of uniform AC\(^0\) reductions, for a significant class of sets \( A \).
- Hardness of MCSP\(^A\) implies hardness of MKTP\(^A\) for a significant class of sets \( A \). This is the first result directly relating the complexity of MCSP\(^A\) and MKTP\(^A\), for any \( A \).

CCS Concepts: • Theory of computation → Complexity classes; Circuit complexity.

Additional Key Words and Phrases: computational complexity, Kolmogorov complexity, Circuit size, MCSP

ACM Reference Format:

1 INTRODUCTION

The Minimum Circuit Size Problem (MCSP) has attracted intense study over the years, because of its close connection with the natural proofs framework of Razborov and Rudich [RR97], and because it is a prominent candidate for NP-intermediate status. It has been known since the work of Ladner [Lad75] that NP-intermediate problems exist if \( \text{P} \neq \text{NP} \), but “natural” candidates for this status are rare. Problems such as factoring and Graph Isomorphism are sometimes put forward as candidates, but there are not strong complexity-theoretic arguments for why these problems...
should not lie in $P$. We prove that a very weak cryptographic assumption implies that a $n^{1-o(1)}$ approximation for $MCSP$ is NP-intermediate.

$MCSP$ is hard for $SZK$ [AD17] under $BPP$ reductions, but the situation is quite different when more restricted notions of reducibility are considered. Recent results [AHK17, HW16, MW17] have suggested that $MCSP$ might not even be hard for $P$ under logspace or $AC^0$ reductions (although the evidence is still inconclusive).

The input to $MCSP$ consists of a pair $(T, s)$, where $T$ is a bit string of length $2^m$ representing the truth table of an $m$-variate Boolean function, and $s \in \mathbb{N}; (T, s) \in MCSP$ if there is a circuit computing $T$ having size at most $s$. Note that for different models of circuit (type of gates, allowable fan-in, etc.) and different measures of size (number of gates, number of wires, size of the description of the circuit, etc.) the resulting $MCSP$ problems might have different complexity. No efficient reduction is known between different variants of the problem. However, all prior work on $MCSP$ (such as [ABK*06, AD17, AHK17, AKRR10, HW16, HP15, IKV18, KC00, MW17, Rud17]) applies equally well to any of these variants. $MCSP$ is also closely related to a type of time-bounded Kolmogorov complexity known as $KT,$ which was defined in [ABK*06]. The problem of determining $KT$ complexity, formalized as the language $MKTP = \{(x, s) : KT(x) \leq s\}$ has often been viewed as just another equivalent “encoding” of $MCSP$ in this prior work. (In particular, our results mentioned in the paragraphs above apply also to $MKTP$.) Recently, however, some reductions were presented that are not currently known to apply to $MCSP$ [AGvM*18, HS17]. For instance, it was shown in [AGvM*18] that the Graph Isomorphism problem and several other problems with an algebraic flavor are contained in $ZPP^{MKTP},$ and it is shown in [HS17] that the “random 3SAT problem” reduces to $MKTP$. It is not known if these statements are true for $MCSP$.

In this section, we outline the ways in which this paper advances our understanding of $MCSP$ and related problems, while reviewing some of the relevant prior work.

**Hardness is equivalent to circuit size lower bounds.** Significant effort (e.g. [KC00, MW17, AHK17, HW16]) has been made in order to explain why it is so difficult to show NP-hardness of $MCSP$ or $MKTP$. Most of the results along this line showed implications from hardness of $MCSP$ to circuit size lower bounds: If $MCSP$ or $MKTP$ is NP-hard under some restricted types of reductions, then a circuit size lower bound (which is quite difficult to obtain via current techniques of complexity theory) follows. For example, if $MCSP$ or $MKTP$ is hard for $TC^0$ under Dlogtime-uniform reductions, then $NP \nsubseteq P/poly$ and $\text{DSpace}(n) \nsubseteq \text{io-SIZE}(2^{\epsilon n})$ [MW17, AHK17].

Murray and Williams [MW17] asked if, in general, circuit lower bounds imply hardness of the circuit minimization problems. We answer their questions affirmatively in certain settings: An oracle circuit lower bound $\text{DSpace}(n) \nsubseteq \text{io-SIZE}^{MKTP}(2^{\epsilon n})$ implies that $MKTP$ is hard for $DET$ under logspace-uniform reductions (Theorem 4.5).

At this point, it is natural to ask if the circuit lower bounds are in fact equivalent to hardness of $MKTP$. We indeed show that this is the case, when we consider the minimum oracle circuit size problem. For an oracle $A$, $MCSP_A$ is the set of pairs $(T, s)$ such that $T$ is computed by a size-$s$ circuit that has “oracle gates” for $A$ in addition to standard AND, OR, and NOT gates. The related $MKTP_A$ problem asks about the time-bounded Kolmogorov complexity of a string, when the universal Turing machine has access to the oracle $A$. For a significant class of oracles $A$ that are hard for $PH$, we show that $\text{DSpace}(n) \nsubseteq \text{io-SIZE}^{A}(2^{\epsilon n})$ for some $\epsilon > 0$ if and only if $MKTP_A$ is hard for $DET$ under a certain class of reductibilities. (See Theorem 4.7, and the Remark after Corollary 4.10.)

That is, it is impossible to prove hardness of $MKTP_A$ (under some reductibilities) without proving circuit lower bounds, and vice versa. Our results clearly connect the fact that it is difficult to obtain hardness of $MKTP_A$ with the fact that circuit size lower bounds are difficult.
Hardness under local reductions, and unconditional lower bounds. Murray and Williams [MW17] showed that MCSP and MKTP are not hard for TC⁰ under so-called local reductions computable in time less than \( \sqrt{n} \) – and thus in particular they are not hard under NC⁰ reductions that are very uniform (i.e., there is no routine computable in time \( t(n) < n^{5/6-\epsilon} \) that, on input \( (n, i) \) outputs the \( O(1) \) queries upon which the \( i \)-th output bit of such an NC⁰ circuit depends). Murray and Williams speculated that this might be a promising first step toward showing that MCSP is not hard for NP under Dlogtime-uniform AC⁰ reductions, since it follows from [Agr11] that any set that is hard for TC⁰ under P-uniform AC⁰ reductions is also hard for TC⁰ under P-uniform NC⁰ reductions. Indeed, the results of Murray and Williams led us to expect that MCSP and MKTP are not even hard for PARITY under non-uniform NC⁰ reductions.

Contrary to these expectations, we show that MKTP is hard not only for TC⁰ but even for the complexity class DET under non-uniform NC⁰ reductions (Theorem 4.3). Consequently, MKTP is not in AC⁰[\( p \)] for any prime \( p \).¹ Note that it is still not known whether MCSP or \( R_{\text{KT}} = \{ x : \text{KT}(x) \geq |x| \} \) is in AC⁰[\( p \)]. It is known [ABK⁺06] that neither of these problems is in AC⁰. Under a plausible derandomization hypothesis, this non-uniform reduction can be converted into a Dlogtime-uniform \( \leq_{\text{tt}}^{AC^0} \) reduction that is an AND of NC⁰-computable queries. Thus “local” reductions are more effective for reductions to MKTP than may have been suspected.

Our DET-hardness result is proved by building on a randomized reduction of [AGvM⁺18] reducing Graph Isomorphism to MKTP. We modify that construction, to obtain a nonuniform AC⁰ reduction (Corollary 4.2). The restricted version of Graph Isomorphism that we use is known to be hard for DET [Tor04]. Our proof of Theorem 4.3 then appeals to the “Gap Theorem” of [AAR98], in order to conclude that DET \( \leq_{\text{tt}}^{NC^0} \) MKTP; the Gap Theorem states that, for any class \( C \) closed under TC⁰ reductions, \( C \)-hardness under \( \leq_{\text{tt}}^{AC^0} \) reductions implies \( C \)-hardness under \( \leq_{\text{tt}}^{NC^0} \) reductions.

Somewhat remarkably, Oliveira and Santhanam [OS17] have independently shown that MCSP and MKTP are hard for DET under non-uniform \( \leq_{\text{tt}}^{TC^0} \) reductions. Their proof relies on self-reducibility properties of the determinant, whereas our proof relies on the fact that Graph Isomorphism is hard for DET [Tor04]. Their results have the advantage that they apply to MCSP rather than merely to MKTP, but because their reduction is more complex (TC⁰, as contrasted with AC⁰), they do not obtain unconditional lower bounds, as in Corollary 4.4.

Our hardness results (both unconditional hardness results under nonuniform reductions, and conditional uniform hardness results) are summarized in Table 1.

Table 1. Hardness for MKTP\(^A\): MKTP\(^A\) is hard for DET under the type of reducibility listed in the first column, if oracle \( A \) satisfies the condition listed in the second column. The last column shows the theorem where the result is stated in the paper.

<table>
<thead>
<tr>
<th>reductions</th>
<th>condition on ( A )</th>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>nonuniform ( \leq_{\text{tt}}^{NC^0} )</td>
<td>every ( A )</td>
<td>Theorem 4.3</td>
</tr>
<tr>
<td>P-uniform ( \leq_{\text{tt}}^{AC^0} )</td>
<td>( E \nsubseteq \text{io-SIZE}^{\text{MKTPA}}(2^{cn}) )</td>
<td>Corollary 4.9</td>
</tr>
<tr>
<td>L-uniform ( \leq_{\text{tt}}^{AC^0} )</td>
<td>( \text{DSPACE}(n) \nsubseteq \text{io-SIZE}^{\text{MKTPA}}(2^{cn}) )</td>
<td>Theorem 4.5</td>
</tr>
<tr>
<td>Dlogtime-uniform ( \leq_{\text{tt}}^{AC^0} )</td>
<td>( \Sigma_d \text{TIME}(n) ) hard on average for ( \text{io-SIZE}^{\text{MKTPA}}(2^{cn}) )</td>
<td>Theorem 4.6</td>
</tr>
</tbody>
</table>

Implications among hardness conditions for MKTP and MCSP. No \( \leq^p \) reductions are known between MKTP\(^A\) or MCSP\(^A\) for any \( A \). Although most previous complexity results for one of the problems have applied immediately to the other, via essentially the same proof, there

¹Subsequent to our work, a stronger average-case lower bound against AC⁰[\( p \)] was proved [HS17]. The techniques of [HS17] do not show how to reduce DET, or even smaller classes such as TC⁰, to MKTP. Thus our work is incomparable to [HS17].
has not been any proven relationship among the problems. For the first time, we show that, for many oracles $A$, hardness for $\text{MCSP}^A$ implies hardness for $\text{MKTP}^A$ (Theorem 4.7).

A reduction that is not “oracle independent”. Hirahara and Watanabe [HW16] observed that all of the then-known reductions to $\text{MCSP}$ and $\text{MKTP}$ were “oracle-independent”, in the sense that, for any class $C$ and reducibility $\leq_r$, all proofs that $\text{MCSP}$ (or $\text{MKTP}$) is hard for $C$ under $\leq_r$ also show that $\text{MCSP}^A$ ($\text{MKTP}^A$) is also hard for $C$, for every $A$. In addition, they showed an inherent limitation of oracle-independent proofs: They showed that oracle-independent $\leq_r$-reductions cannot show hardness for any class larger than $P$.

This motivates the search for reductions that are not oracle-independent. We give a concrete example of a Dlogtime-uniform $\leq^{\text{AC}^0}_{\text{cont}}$ reduction that (under a plausible complexity assumption) reduces $\text{DET}$ to $\text{MKTP}$. This is not an oracle independent reduction, since $\text{MKTP}^{\text{QBF}}$ is not hard for $\text{DET}$ under this same class of reductions (Corollary 4.10).

A clearer picture of how hardness “evolves”. It is instructive to contrast the evolution of the class of problems reducible to $\text{MKTP}^A$ under different types of reductions, as $A$ varies from very easy ($A = \emptyset$) to complex ($A = \text{QBF}$). For this thought experiment, we assume the very plausible hypothesis that $\text{DSPACE}(n) \not\subseteq \text{io-SIZE}(2^n)$. Restrictions of $\text{QBF}$ give a useful parameterization for the complexity of $A$. Consider $A$ varying from being complete for each level of $\text{PH}$ (that is, quantified Boolean formulas with $O(1)$ alternations between $\forall$ and $\exists$ quantifiers), to instances of $\text{QBF}$ with $\log^* n$ alternations, then to $O(\log n)$ alternations etc., through to $2^{\sqrt{\log n}}$ alternations, and so on, until finally $A = \text{QBF}$. Since $\text{DSPACE}(n) \subseteq \text{p^{QBF}/poly}$, at some point in this evolution we have $\text{DSPACE}(n) \subseteq \text{io-SIZE}(2^{\epsilon n})$; it is plausible to assume that this doesn’t happen until $A$ has at least $\log n$ quantifier alternations, or more.

At all stages in this evolution $\text{SZK} \subseteq \text{BPP}^{\text{MKTP}^A}$ [AD17], until at some point $\text{BPP}^{\text{MKTP}^A}$ expands to coincide with $\text{PSpace}$ [ABK+06]. Also, at all stages in this evolution $\text{DET} \leq_{\text{m}}^{\text{NC}^0} \text{reduces to } \text{MKTP}^A$. No larger class is known to $\leq_{\text{m}}^{\text{NC}^0}$-reduce to $\text{MKTP}^A$; even when $A = \text{QBF}$ we do not know, for instance, if $\text{NC}^3 \leq_{\text{m}}^{\text{NC}^0}$-reduces to $\text{MKTP}^A$). Thus these reductions behave “monotonically”, in the sense that as the complexity of $A$ increases, the class of problems reducible to $\text{MKTP}^A$ does not shrink noticeably, and sometimes appears to grow markedly.

The situation is much more intriguing when we consider the uniform class of $\leq_{\text{T}}^{\text{AC}^0}$ reductions that arise from derandomizing the nonuniform $\leq_{\text{m}}^{\text{NC}^0}$ reductions from $\text{DET}$. At the start, when $A = \emptyset$, we have $\text{DET}$ reducing to $\text{MKTP}^A$, and this is maintained until $A$ becomes complex enough so that $\text{DSPACE}(n) \subseteq \text{io-SIZE}(2^{\epsilon n})$. At this point, not only does $\text{DET}$ not reduce to $\text{MKTP}^A$, but neither does $\text{PARITY}$! (See Theorem 4.7.)

This helps place the results of [AHK17] in the proper context. In [AHK17] strong evidence was presented against $\text{MCSP}^{\text{QBF}}$ being hard for $P$ under $\leq_{\text{m}}^{\text{L}}$ reductions, and this was taken as indirect evidence that $\text{MCSP}$ itself should not be hard for $P$, since $\text{MCSP} \in \text{NP}$ and thus is much “easier” than the $\text{PSPACE}$-complete problem $\text{MCSP}^{\text{QBF}}$. However, we expect that $\text{MCSP}^A$ and $\text{MKTP}^A$ should behave somewhat similarly to each other, and it can happen that a class can reduce to $\text{MKTP}$ (Theorem 4.5) and not reduce to $\text{MKTP}^A$ for a more powerful oracle $A$ (Corollary 4.10).

Some of these results are summarized in the following tables. Table 2 indicates what is known to follow for the (unrelativized) MCSP and MKTP problems.

In the “intermediate” case, where the oracle $A$ is a complete set for $\text{PP}$, Table 3 shows both hardness and non-hardness consequences of some plausible hypotheses.

When the oracle $A$ is the $\text{PSpace}$-complete set $\text{QBF}$, the situation is summarized in Table 4.

**Hardness of the Gap problem.** Our new hardness results for $\text{MKTP}^A$ share with earlier reductions the property that they hold even for “Gap” versions of the problem. That is, for some
Table 2. Consequences of hardness for MCSP and MKTP: If MCSP or MKTP is $C$-hard under $R$, then condition $S$ holds. The last column shows where the result is found.

<table>
<thead>
<tr>
<th>class $C$</th>
<th>reductions $R$</th>
<th>statement $S$</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC$^0$</td>
<td>Dlogtime-uniform $\leq_{\text{Dlog}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly and LTH $\not\subseteq$ io-SIZE$[2^{\epsilon n}]$</td>
<td>[AHK17]</td>
</tr>
<tr>
<td>PARITY</td>
<td>L-uniform $\leq_{\text{L}}^{ac_0}$</td>
<td>MKTP $\not\subseteq$ P/poly or DSPACE$(n) \not\subseteq$ io-SIZE$[2^{\epsilon n}]$</td>
<td>4.7</td>
</tr>
<tr>
<td>TC$^0$</td>
<td>Dlogtime-uniform $\leq_{\text{Dlog}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly or CH = PH (hence NP $\not\subseteq$ TC$^0$)</td>
<td>4.15</td>
</tr>
<tr>
<td>NC$^1$</td>
<td>Dlogtime-uniform $\leq_{\text{Dlog}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly or PSPACE = PH (hence NP $\not\subseteq$ NC)</td>
<td>4.15</td>
</tr>
<tr>
<td>NP</td>
<td>Dlogtime-uniform $\leq_{\text{Dlog}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly or NEXP = MA (hence NP $\not\subseteq$ MA $\cap$ P/poly)</td>
<td>4.16</td>
</tr>
<tr>
<td>NP</td>
<td>L-uniform $\leq_{\text{L}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly or NEXP = PSPACE</td>
<td>4.19</td>
</tr>
<tr>
<td>NP</td>
<td>P-uniform $\leq_{\text{P}}^{ac_0}$</td>
<td>NP $\not\subseteq$ P/poly or NEXP = EXP</td>
<td>4.19</td>
</tr>
</tbody>
</table>

Table 3. Hardness and non-hardness for MKTP$^\text{PP}$ and MCSP$^\text{PP}$: If condition $S$ holds, then the problems in the first column are hard (or not-hard, as indicated) for the class $C$ under $R$. Note that, in going from the first line to the second, $C$ becomes larger, and (except for the uniformity notion) the class of reductions becomes more restrictive. This highlights the importance of the uniformity condition, in determining if these problems are hard for a given class. The last column shows the theorem where the result is stated in the paper.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Hard?</th>
<th>class $C$</th>
<th>reductions $R$</th>
<th>statement $S$</th>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>MKTP$^\text{PP}$ &amp; MCSP$^\text{PP}$</td>
<td>Not hard</td>
<td>NC$^1$</td>
<td>Dlogtime-uniform $\leq_{\text{Dlog}}^{ac_0}$</td>
<td>PSPACE $\neq$ PH$^\text{PP}$</td>
<td>4.12</td>
</tr>
<tr>
<td>MKTP$^\text{PP}$</td>
<td>Is hard</td>
<td>DET</td>
<td>L-uniform $\leq_{\text{L}}^{ac_0}$</td>
<td>DSPACE$(n) \not\subseteq$ io-SIZE$^{\text{MKTP}^\text{PP}}[2^{\epsilon n}]$</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Table 4. Non-hardness for MKTP$^\text{QBF}$ and MCSP$^\text{QBF}$: Neither MKTP$^\text{QBF}$ nor MCSP$^\text{QBF}$ is $C$-hard under $R$, assuming that the condition $S$ holds. The last column shows the theorem where the result is stated in the paper.

<table>
<thead>
<tr>
<th>class $C$</th>
<th>reductions $R$</th>
<th>condition $S$</th>
<th>Theorem/Corollary</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARITY</td>
<td>L-uniform natural $\leq_{\text{L}}^{ac_0}$</td>
<td>True</td>
<td>4.10</td>
</tr>
<tr>
<td>PSPACE</td>
<td>$\leq_{\text{T}}^{\text{L}}$</td>
<td>True</td>
<td>4.11</td>
</tr>
<tr>
<td>NP</td>
<td>$\leq_{\text{T}}^{\text{L}}$</td>
<td>PSPACE $\neq$ NEXP</td>
<td>4.11</td>
</tr>
<tr>
<td>NP</td>
<td>L-uniform $\leq_{\text{L}}^{ac_0}$</td>
<td>PSPACE $\neq$ NEXP</td>
<td>4.18</td>
</tr>
<tr>
<td>NP</td>
<td>P-uniform $\leq_{\text{P}}^{ac_0}$</td>
<td>EXP $\neq$ NEXP</td>
<td>4.18</td>
</tr>
<tr>
<td>P</td>
<td>$\leq_{\text{T}}^{\text{P}}$</td>
<td>PSPACE $\neq$ EXP</td>
<td>4.11</td>
</tr>
<tr>
<td>DET</td>
<td>nonuniform $\leq_{\text{L}}^{ac_0}$</td>
<td>False</td>
<td>4.3</td>
</tr>
</tbody>
</table>

For $\epsilon > 0$, the reduction works correctly for any solution to the promise problem with "yes" instances $\{(x, s) : KT^A(x) \leq s\}$ and "no" instances $\{(x, s) : KT^A(x) > s + |x|^{\epsilon}\}$. However, we do not know if they carry over to instances with a wider "gap" between the Yes and No instances; earlier hardness results such as those of [ABKT06, AKRR10, AD17, Rud17] hold for a much wider gap (such as with the Yes instances having KT(x) < |x|^{\epsilon}, and the No instances with KT(x) $\geq$ |x|), and this is one reason why they applied both to MKTP and to MCSP. Thus there is interest in whether it is possible to reduce MCSP with small "gap" to MCSP with large "gap". If this were possible, then MCSP and MKTP would be interreducible in some sense.

Earlier work [AHK17] had presented unconditional results, showing that "gap" versions of MCSP could not be hard for TC$^0$ under $\leq_{\text{ac}_0}$ reductions, unless those reductions had large "stretch" (mapping short inputs to long outputs). In Section 3.2, we show that BPP-Turing reductions among gap MCSP problems require large stretch, unless MCSP $\in$ BPP.
Natural NP-intermediate Problems. Using very similar techniques, in Section 3 we also consider gap MCSP problems where the "gap" is quite large (i.e., problems of approximating the minimum circuit size for a truth table of size $n$ within a factor of $n^{1-o(1)}$). Problems of this sort are of interest, because of the role they play in the natural proofs framework of [RR97], if one is trying to prove circuit lower bounds of size $2^{n^{o(1)}}$. Our Theorem 3.6 shows that these problems are NP-intermediate in the sense that they do not lie in $P/poly$ and are not NP-hard under $P/poly$ reductions, under modest cryptographic assumptions (weaker than assuming that factoring or discrete log requires superpolynomial-size circuits, or assuming the existence of a one-way function).

To the best of our knowledge, these problems are the first natural NP-intermediate problems under the existence of an arbitrary one-way function.

Our new insight on MCSP here is that, if the gap problems are NP-hard, then MCSP is “strongly downward self-reducible”: that is, any instance of MCSP of size $n$ can be reduced to instances of size $n'$. In the past, many natural problems have been shown to be strongly downward self-reducible (see [AK10]); Our contribution is to show that MCSP also has such a property (under the assumption that the gap MCSP problems are NP-hard). In fact, we also present a similar argument showing that a $n^{1-o(1)}$ approximation for CLIQUE is NP-intermediate if $NP \not\subseteq P/poly$.

2 PRELIMINARIES

We assume the reader is familiar with standard DTIME and DSPACE classes. We also occasionally refer to classes defined by time-bounded alternating Turing machines: $ATIME(t(n))$, or by simultaneously bounding time and the number of alternations between existential and universal configurations: $ATIME-ALT(t(n), a(n))$.

We refer the reader to the text by Vollmer [Vol99] for background and more complete definitions of the standard circuit complexity complexity classes

$NC^0 \subseteq AC^0 \subseteq AC^0[p] \subseteq TC^0 \subseteq NC^1 \subseteq P/poly$,

as well as the standard complexity classes $L \subseteq P \subseteq NP \subseteq PH \subseteq PSPACE \subseteq EXP \subseteq NEXP$. $E$ denotes $DTIME(2^{O(n)})$; $E$ contains many of the standard complete sets for $EXP$. We shall also have occasion to refer to the counting hierarchy, $CH$ [Tor91], which consists of the classes $PP, PPP, PP^{PP}, PP^{PP^{PP}},$ etc.

This brings us to the topic of reducibility. Let $C$ be either a class of functions or a class of circuits. We say that $A \leq_C^B$ if there is a function $f \in C$ (or $f$ computed by a circuit family in $C$, respectively) such that $x \in A$ iff $f(x) \in B$. We will make use of $\leq_{TC}^m, \leq_{AC}^m, \leq_{NC}^m, and \leq_{NC^0}$ reducibility. The more powerful notion of Turing reducibility also plays an important role in this work. Here, $C$ is a complexity class that admits a characterization in terms of Turing machines or circuits, which can be augmented with an “oracle” mechanism, either by providing a “query tape” or “oracle gates”.

We say that $A \leq_T^C B$ if there is a oracle machine in $C$ (or a family of oracle circuits in $C$) accepting $A$, when given oracle $B$. We make use of $\leq_{T}^{P/poly}, \leq_{T}^{BPP}, \leq_{T}^{P}, \leq_{T}^{AC^0}$ and $\leq_{T}^{NC^1}$ reducibility; instead of writing $A \leq_T^{P/poly} B$ or $A \leq_T^{BPP} B$, we will more frequently write $A \in P^B/poly$ or $A \in BPP^B$. Turing reductions that are "nonadaptive" – in the sense that the list of queries that are posed on input $x$ does not depend on the answers provided by the oracle – are called truth table reductions. We make use of $\leq_{T}^{AC^0} and \leq_{T}^{NC^1}$ reducibility.

Now that circuit-based notions of reducibility have been introduced, we can present the definition of one more class that plays a large role in this work: DET is the class of problems that are reducible to the problem Det of computing the determinant of integer matrices, by $NC^1$-Turing reductions. DET lies between $L$ and $P$. Although $AC^0 \not\subseteq NC^1 \subseteq L$, the reader should be aware that $L^Det$ is contained in the class of problems $\leq_{T}^{AC^0}$ reducible to Det, which in turn is contained in DET.
(See [AO96, All04] for more details.) In fact, since \( \leq_{\text{P}}^{k} \) reductions are nonadaptive [LL76], in many situations \( \leq_{\text{P}}^{k} \) reductions can be simulated by \( \leq_{\text{AC}}^{0} \) reductions.

Kabanets and Cai [KC00] sparked renewed interest in MCSP and highlighted connections between MCSP and more recent progress in derandomization. They introduced a class of reductions to MCSP, which they called natural reductions. Recall that instances of MCSP are of the form \((T, s)\) where \(s\) is a "size parameter". A \( \leq_{\text{P}}^{0} \) reduction \( f \) is called natural if \( f(x) \) is of the form \( f(x) = (f_1(x), f_2(|x|)) \). That is, the "size parameter" is the same, for all inputs \( x \) of the same length.

The notation \( \text{io-SIZE}(s(n)) \) denotes the class of all languages \( A \) such that, for infinitely many lengths \( n \), there is a circuit of size at most \( s(n) \) accepting exactly the strings of length \( n \) in \( A \). (Although this definition can depend upon the precise notion of "circuit size" being considered, every statement that we make using this notation holds using any reasonable notion of "size".) If \( B \) is a language, then \( \text{io-SIZE}^B(s(n)) \) denotes the class of all languages \( A \) such that, for infinitely many lengths \( n \), there is an "oracle circuit" (that is, a circuit that has "oracle gates" in addition to the standard Boolean gates) of size at most \( s(n) \) accepting exactly the strings of length \( n \) in \( A \), when given \( B \) as an oracle.

Whenever circuit families are discussed (either when defining complexity classes, or reducibilities), one needs to deal with the issue of uniformity. For example, the class \( \text{AC}^0 \) (corresponding to families \( \{C_n : n \in \mathbb{N}\} \) of unbounded fan-in AND, OR, and NOT gates having size \( n^{O(1)} \) and depth \( O(1) \)) comes in various flavors, depending on the complexity of computing the mapping \( 1^n \mapsto C_n \). When this is computable in polynomial time (or logarithmic space), then one obtains \( \text{P-uniform AC}^0 \) (logspace-uniform \( \text{AC}^0 \), respectively). If no restriction at all is imposed, then one obtains non-uniform \( \text{AC}^0 \). As discussed in [Vol99], the more restrictive notion of Dlogtime-uniform \( \text{AC}^0 \) is frequently considered to be the "right" notion of uniformity to use when discussing small complexity classes such as \( \text{AC}^0, \text{AC}^0[p] \) and \( \text{TC}^0 \). If these classes are mentioned with no explicit mention of uniformity, then Dlogtime-uniformity is intended. For uniform \( \text{NC}^1 \) the situation is somewhat more complicated, as discussed in [Vol99]; there is wide agreement that the "correct" definition coincides with \( \text{ATIME}(O(\log n)) \).

There are many ways to define time-bounded Kolmogorov complexity. The definition \( \text{KT}(x) \) was proposed in [ABK+06], and has the advantage that it is polynomially-related to circuit size (when a string \( x \) is viewed as the truth table of a function).

**Definition 2.1.** Let \( U \) be a Turing machine and let \( A \) be an oracle. The measure \( \text{KT}^A_U(x) \) is defined to be

\[
\text{KT}^A_U(x) = \min \{ |d| + t : \forall b \in \{0, 1, *\} \forall i \leq |x| + 1, \text{ the machine } U^{A,d}(i, b) \text{ accepts in } t \text{ steps iff } x_i = b \}.
\]

We omit the superscript \( A \) if \( A = \emptyset \).

It is observed in [ABK+06] that, for any two universal machines \( U \) and \( U' \), there is a constant \( c \) such that \( \text{KT}^A_U(x) \leq c \cdot \text{KT}^A_{U'}(x) \log \text{KT}^A_{U'}(x) \). We pick one such universal machine \( U \) and define \( \text{KT}(x) \) to be \( \text{KT}_U(x) \). The definition of \( \text{KT} \) was designed in such a way as to make it useful in proving theorems about MCSP. The fact that the machine \( U \) is allowed to have random access to the bits of the "description" \( d \) has the by-product that certain algorithms are easier to implement using the machines of the \( \text{KT} \) formalism than using the hardware formalisms of circuit complexity. This helps explain why some of the theorems that we prove for \( \text{MKTP} \) are not currently known to hold for MCSP.

For completeness, we include a definition of \( \text{MCSP}^A \) (although this already appears informally in the introduction).
We claim that \( \epsilon \) even when \( |T| \leq r \) if \( B \leq r A \) for every set \( A \) that is a solution to the promise problem.

3 GAPMCSP

In this section, we consider the "gap" versions of MCSP and MKTP. We focus primarily on MCSP, and for simplicity of exposition we consider the "size" of a circuit to be the number of AND and OR gates of fan-in two. (NOT gates are "free"). The arguments can be adjusted to consider other circuit models and other reasonable measures of "size" as well. Given a truth table \( T \), let \( CC(T) \) be the size of the smallest circuit computing \( T \), using this notion of "size".

Definition 3.1. For any function \( \epsilon : \mathbb{N} \to (0, 1) \), let \( Gap_\epsilon MCSP \) be the approximation problem that, given a truth table \( T \), asks for outputting a value \( f(T) \in \mathbb{N} \) such that

\[
CC(T) \leq f(T) \leq |T|^{1-\epsilon(|T|)} \cdot CC(T).
\]

Note that this approximation problem can be formulated as the following promise problem. (See also [Gol06] for similar comments.)

Proposition 3.2. \( Gap_\epsilon MCSP \) is polynomial-time Turing equivalent to the following promise problem \((Y, N)\):

\[
Y := \{ (T, s) \mid CC(T) < s/|T|^{1-\epsilon(|T|)} \},
\]

\[
N := \{ (T, s) \mid CC(T) \geq s \},
\]

where \( T \) is a truth table and \( s \in \mathbb{N} \).

Proof. Given a solution \( A \) of the promise problem \((Y, N)\), one can compute an approximation \( f(T) \) of \( CC(T) \) as follows:

\[
f(T) := \max \{ s \in \mathbb{N} \mid (T, s) \notin A \}
\]

We claim that \( f(T) \) satisfies the approximation guarantee as given in Definition 3.1. By the definition of \( f(T) \), we have \( (T, f(T)) \notin A \), which implies that \( (T, f(T)) \notin Y \), and thus \( CC(T) \geq f(T)/|T|^{1-\epsilon(|T|)} \).

Similarly, by the definition of \( f(T) \), we have \( (T, f(T) + 1) \in A \), which implies that \( (T, f(T) + 1) \notin N \), and thus \( CC(T) \leq f(T) \). To summarize, we have \( CC(T) \leq f(T) \leq |T|^{1-\epsilon(|T|)} \cdot CC(T) \) and thus \( f(T) \) satisfies Definition 3.1.

On the other hand, suppose that an approximation \( f(T) \) of \( CC(T) \) is given. We can define a solution \( A \) of the promise problem \((Y, N)\) as \( A := \{ (T, s) \mid f(T) < s \} \). We claim that \( A \) indeed is a solution of \((Y, N)\). If \( (T, s) \in Y \), then \( f(T) \leq |T|^{1-\epsilon(|T|)} \cdot CC(T) < s \) and therefore \( (T, s) \in A \). On the other hand, if \( (T, s) \in N \), then \( f(T) \geq CC(T) \geq s \), which implies \( (T, s) \notin A \).

Note that \( Gap_\epsilon MCSP \) becomes easier when \( \epsilon \) becomes smaller. If \( \epsilon(n) = o(1) \), then (using the promise problem formulation) it is easy to see that \( Gap_\epsilon MCSP \) has a solution in \( \text{DTIME}(2^{n^{o(1)}}) \), since the Yes instances have witnesses of length \( |T|^{o(1)} \). However, it is worth emphasizing that, even when \( \epsilon(n) = o(1) \), \( Gap_\epsilon MCSP \) is a canonical example of a combinatorial property that is
useful in proving circuit size lower bounds of size $2^{\Omega(n)}$, in the sense of [RR97]. Thus it is of interest that MCSP cannot reduce to $\text{Gap}_e\text{MCSP}$ in this regime under very general notions of reducibility, unless MCSP itself is easy.

**Theorem 3.3.** For any polynomial-time-computable nonincreasing $\epsilon(n) = o(1)$, if $\text{MCSP} \in \text{BPP}^{\text{Gap}_e\text{MCSP}}$ then $\text{MCSP} \in \text{BPP}$.

A new idea is that the $\text{Gap}_e\text{MCSP}$ is "strongly downward self-reducible." We will show that any $\text{Gap}_e\text{MCSP}$ instance of length $n$ is reducible to $n^{1-\epsilon}$ $\text{MCSP}$ instances of length $n^c$. To this end, we will exploit the following simple fact.

**Lemma 3.4.** For a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, a string $x \in \{0, 1\}^k$ and $k \in \mathbb{N}$, let $f_k : \{0, 1\}^{n-k} \rightarrow \{0, 1\}$ be a function defined as $f_k(y) := f(x, y)$. Then, the following holds:

$$\max_{x \in \{0, 1\}^k} \text{CC}(f_k) \leq \text{CC}(f) \leq 2^k \cdot \left( \max_{x \in \{0, 1\}^k} \text{CC}(f_k) + 3 \right),$$

(In other words, $\max_{x \in \{0, 1\}^k} \text{CC}(f_k)$ gives an approximation of $\text{CC}(f)$ within a factor of $2^k$.)

**Proof.** We first claim that $\max_{x \in \{0, 1\}^k} \text{CC}(f_k) \leq \text{CC}(f)$. Indeed, let $C$ be a minimum circuit that computes $f$ and let $x$ be an arbitrary string of length $k$. For each $x \in \{0, 1\}^k$, define a circuit $C_x$ as $C_x(y) := C(x, y)$ on input $y \in \{0, 1\}^{n-k}$. Then, since $C_x$ computes $f_k$ and the size of $C_x$ is at most that of $C$, we have $\text{CC}(f_k) \leq \text{CC}(f)$.

Next, we claim that $\text{CC}(f) \leq 2^k \cdot \left( \max_{x \in \{0, 1\}^k} \text{CC}(f_k) + O(1) \right)$. For any $x \in \{0, 1\}^k$, let $C_x$ be a minimum circuit that computes $f_k$. We build a circuit that computes $f =: f_e$ recursively as follows: $f_e(x, y) = (x_1 \land f_{g_0}(x_2, \ldots, x_k, y)) \lor (x_1 \land f_{g_1}(x_2, \ldots, x_k, y))$ for any string $z$ of length less than $k$, and $f_e(y) = C_x(y)$ for any $x \in \{0, 1\}^k$. Since $\text{CC}(f_e) \leq \text{CC}(f_{g_0}) + \text{CC}(f_{g_1}) + 3$ we obtain

$$\text{CC}(f) \leq \sum_{x \in \{0, 1\}^k} C_x(y) + 3 \cdot (2^k - 1) \leq 2^k \cdot \left( \max_{x \in \{0, 1\}^k} \text{CC}(f_k) + 3 \right).$$

**Proof of Theorem 3.3.** Let $M$ be an oracle BPP Turing machine which reduces MCSP to $\text{Gap}_e\text{MCSP}$. Let $|T|^c$ be an upper bound for the running time of $M$, given a truth table $T$, and let $|T| = 2^n$.

We recursively compute the circuit complexity of $T$ by the following procedure: Run $M$ on input $T$. If $M$ makes a query $S$ to the $\text{Gap}_e\text{MCSP}$ oracle, then divide $S$ into consecutive substrings $S_1, \ldots, S_{2^k}$ of length $|S| \cdot 2^{-k}$ such that $S_1 \cdot S_2 \cdots S_{2^k} = S$ (where $k$ is a parameter, chosen later, that depends on $|S|$), and compute the circuit complexity of each $S_i$ recursively for each $i \in [2^k]$. Then continue the simulation of $M$, using the value $2^k \cdot \left( \max_{i \in [2^k]} \text{CC}(S_i) + 3 \right)$ as an approximation to $\text{CC}(S)$.

We claim that the procedure above gives the correct answer. For simplicity, let us first assume that the machine $M$ has zero error probability. It suffices to claim that the simulation of $M$ is correct in the sense that every query of $M$ is answered with a value that satisfies the approximation criteria of $\text{Gap}_e\text{MCSP}$. Suppose that $M$ makes a query $S$. By the assumption on the running time of $M$, we have $|S| \leq |T|^c = 2^{nc}$. By Lemma 3.4, we have

$$\text{CC}(S) \leq 2^k \cdot \left( \max_{i \in [2^k]} \text{CC}(S_i) + 3 \right) \leq 2^k \cdot (\text{CC}(S) + 3).$$
In particular, the estimated value satisfies the promise of GapMCSP if $2^k \cdot (CC(S) + 3) \leq |S|^{1-\epsilon(|S|)} \cdot CC(S)$. Since we may assume without loss of generality that $CC(S) \geq 3$, it suffices to make sure that $2^{k+1} \cdot CC(S) \leq |S|^{1-\epsilon(|S|)} \cdot CC(S)$. Let $|S| = 2^n$. Then, in order to satisfy $k + 1 \leq (1 - \epsilon(|S|)) \cdot m$, let us define $k := (1 - \epsilon(|S|)) \cdot m - 1$. For this particular choice of $k$, the estimated value $2^k \cdot \left(\max_{x \in [2^k]} CC(S_x) + 3\right)$ of the circuit complexity of $S$ satisfies the promise of GapMCSP, which implies that the reduction $M$ computes the correct answer for MCSP.

Now we analyze the time complexity of the algorithm. Each recursive step makes at most $2^{2cn}$ many recursive calls, because there are potentially $2^{cn}$ many queries $S$ of $M$, each of which may produce at most $2^k \leq 2^{cn}$ recursive calls. The length of each truth table $S_i$ that arises in one of the recursive calls is $|S_i| = |S| \cdot 2^{-k} = 2^{m-k} = 2^{\epsilon(|S|) \cdot m+1}$. We claim that $|S_i| \leq 2^{1+(n/2)}$ holds for sufficiently large $n$. Let us take $n$ to be large enough so that $\epsilon(2^{n/2}) \leq 1/2c$. If $m \geq n/2$, then $|S_i| \leq 2^c(2m) \cdot m+1 \leq 2^{c(2n/2) \cdot c \cdot n+1} \leq 2^{1+(n/2)}$. Otherwise, since $m \leq n/2$ and $\epsilon(|S|) < 1$, we obtain $|S_i| \leq 2^{\epsilon(|S|) \cdot m+1} \leq 2^{1+(n/2)}$. Therefore, on inputs of length $2^n$, each recursive call produces instances of length at most $2^{1+(n/2)}$. The overall time complexity can be estimated as $2^{c'n} \cdot 2^{c'n/2} \cdot 2^{c'n/4} \cdots = 2^{c'n}$ for some constant $c'$ (say, $c' = 3c$), which is a polynomial in the input length $2^n$.

We note that the analysis above works even for randomized reductions that may err with exponentially small probability. Since we have proved that the algorithm runs in polynomial time, the probability that the algorithm makes an error is at most a polynomial times an exponentially small probability, which is still exponentially small probability (by the union bound).

\textbf{Remark:} If we drop the assumption that $\epsilon(n)$ be computable, then the proof of Theorem 3.3 still shows that if MCSP $\in P^{Gap_{\epsilon}MCSP}/poly$ then MCSP $\in P/poly$.

\textbf{Corollary 3.5.} Let $\epsilon(n) = o(1)$. If GapMCSP has no solution in $P/poly$ then Gap_{\epsilon}MCSP is not hard for NP (or even for MCSP) under $\leq_T^{P/poly}$ reductions, and is thus NP-intermediate.

\textbf{Proof.} This is immediate from the preceding remark. If MCSP $\in P^{Gap_{\epsilon}MCSP}/poly$ then MCSP $\in P/poly$, which in turn implies that Gap_{\epsilon}MCSP has a solution in $P/poly$. \hfill $\Box$

In what follows, we show that the assumption of Corollary 3.5 is true under very modest cryptographic assumptions. It is known that, for any constant $\epsilon > 0$, Gap_{\epsilon}MCSP is SZK-hard under $\leq_T^{P/poly}$ reductions [AD17]. Here, we show that if SZK is not in $P/poly$, then for some $\epsilon(n) = o(1)$, Gap_{\epsilon}MCSP has no solution in $P/poly$. In fact, we can prove something stronger: If auxiliary-input one-way functions exist, then Gap_{\epsilon}MCSP is not in $P/poly$. We now describe auxiliary-input one-way functions.

Most researchers consider the existence of cryptographically-secure one-way functions to be essential for meaningful cryptography [IL89]. That is, one requires a function $f$ computed in polynomial time such that, for any algorithm $A$ computed by polynomial-sized circuits, $Pr_x[f(A(f(x))) = f(x)] = 1/n^{o(1)}$ where $x$ is chosen uniformly at random from $\{0, 1\}^n$. A weaker notion that has been studied in connection with SZK goes by the name \textit{auxiliary-input one-way functions}. This is an indexed family of functions $f_y(x) = F(y, x)$, where $|x| = p(|y|)$ for some polynomial $p$, and $F$ is computable in time polynomial in $|y|$, such that for some infinite set $I$, for any algorithm $A$ computed by polynomial-sized circuits, for all $y \in I$, $Pr_x[f_y(A(f_y(x))) = f_y(x)] = 1/n^{o(1)}$ where $n = |y|$ and $x$ is chosen uniformly at random from $\{0, 1\}^{p(n)}$. It is known that there are promise problems in SZK that have no solution in $P/poly$ only if auxiliary-input one-way functions exist.

\footnote{We have chosen to define one-way functions in terms of security against non-uniform adversaries. It is also common to use the weaker notion of security against probabilistic polynomial-time adversaries, as in [Vad06].}
(This is due to [OW93]; a good exposition can be found in [Vad06, Theorems 7.1 & 7.5], based on earlier work of [Ost91].)

**Theorem 3.6.** If auxiliary-input one-way functions exist, then there is a function \( \epsilon(n) = o(1) \) such that \( \text{Gap}_e^{\text{MCSP}} \) is NP-intermediate. (Namely, \( \text{Gap}_e^{\text{MCSP}} \) has no solution in \( P/\text{poly} \) and \( \text{Gap}_e^{\text{MCSP}} \) is not NP-hard under \( \leq_T^{P/\text{poly}} \) reductions.)

**Remark:** In particular, either one of the following implies that some \( \text{Gap}_e^{\text{MCSP}} \) is NP-intermediate, since each implies the existence of auxiliary-input one-way functions:

1. the existence of cryptographically-secure one-way functions.
2. \( \text{SZK} \) is not in \( P/\text{poly} \).

**Proof.** Let \( F(y, x) \) define an auxiliary-input one-way family of functions \( f_y(x) \) where \( |x| = p(|y|) \) for some polynomial \( p \). Let \( S(n) \) be the size of the smallest circuit \( A \) such that for some \( y \) of length \( n \), \( Pr_y[f_y(A(f_y(x))) = f_y(x)] \geq 1/S(n) \) where \( n = |y| \) and \( x \) is chosen uniformly at random from \( \{0,1\}^{|p(n)|} \). By assumption \( S(n) \) is not bounded by any polynomial. Let \( e(n) \) be a nondecreasing unbounded function such that \( n^{e(n)} < S(n) \) for infinitely many \( n \), where \( c_0 \) is a constant that we will pick later.

At this point, we make use of some standard derandomization tools, including the HILL pseudorandom generator [HILL99], and pseudorandom function generators [GGM86, RR97]. First, we recall the HILL construction, phrased in terms of non-uniform adversaries:

**Theorem 3.7 (see [HILL99]).** Let \( F(y, x) \) be computable uniformly in time polynomial in \( |y| \), and let \( \mu : \mathbb{N} \to [0,1] \). For any oracle \( L \) and any oracle circuit \( M \) of size \( s(n) \), there is a size \( (s(n)^{O(1)}/\mu(n^{O(1)})) \) circuit \( N \) such that the following holds for any \( n \) and \( y \):

\[
\Pr_{|x|=n} \left[ M^L(y, r) = 1 \right] - \Pr_{|x|=n} \left[ M^L(y, G_{f_y}^{\text{HILL}}(x)) = 1 \right] \geq \mu(n),
\]

then

\[
\Pr_{|x|=n} \left[ F(y, N^L(y, F(y, x))) = F(y, x) \right] \geq \mu(n^{O(1)})/n^{O(1)},
\]

where \( r \) and \( x \) are chosen uniformly at random. Here \( G_{f_y}^{\text{HILL}} \) is a pseudorandom generator, where \( G_{f_y}^{\text{HILL}}(x) \) is computable in time polynomial in \( |y| \), as described in [HILL99].

Theorem 3.7 states that if there exists a distinguisher with access to an oracle \( L \) that distinguishes the output of \( G_{f_y}^{\text{HILL}} \) from the uniform distribution, then oracle access to \( L \) suffices to invert \( f_y \) on a significant fraction of the inputs. We now argue that such a distinguisher can be computed by a circuit of size \( n^{O(e(n))} \) with oracle gates for \( \text{Gap}_{e(n)}^{\text{MCSP}} \), where \( e(n) \) is the slow-growing function that we defined earlier.

Let \( G : \{0,1\}^n \to \{0,1\}^{2n} \) be a pseudorandom generator mapping strings of length \( n \) to strings of length \( 2n \), constructed from the generator \( G_{f_y}^{\text{HILL}} \). Furthermore, let \( G_0(x) \) be the first \( n \) bits of \( G(x) \), and let \( G_1(x) \) be the second \( n \) bits of \( G(x) \), so that \( G(x) = G_0(x)G_1(x) \). We now make use of the pseudorandom function generator of Razborov and Rudich [RR97], with the following parameters.

For any string \( w \) of length \( k \), let \( G_w(x) : \{0,1\}^n \to \{0,1\}^n \) be defined as \( G_w(x) = G_w(G_w(\ldots(G_w(x))\ldots)) \).

Define \( G'_y(x, w) \) to be the first bit of \( G_w(x) \), where here the subscript \( y \) refers back to the fact that \( G \) is defined from \( G_{f_y}^{\text{HILL}} \).

Now let \( z(x) \) be the truth table of \( G'_y(x, w) \) (viewed as a function of \( w \)). Since \( |w| = k, |z(x)| = 2^k \). Since \( G'_y \) is computed in time polynomial in the length of \( x \) and \( w \), \( CC(z(x)) < (n+k)^c \) for some constant \( c \).
Now, let us choose $k$ to be $(c + 1)e(n) \log n$. It follows that
\[
\text{CC}(z(x)) < (n + (c + 1)e(n) \log n)^c < n^{c+1} = \left(2^{(c+1)e(n) \log n} \right)^{1/e(n)} = |z(x)|^{1/e(n)}.
\]
That is, from the pseudorandom distribution, we always have $\text{CC}(z(x)) < |z(x)|^{1/e(n)}$, whereas a random string has CC complexity at least roughly equal to the length of the string with high probability.

Thus, an oracle gate for any oracle $L$ that satisfies the $\text{Gap}_{1/e(n)}\text{MCSP}$ promise problem can distinguish random functions from the output of the pseudorandom function generator. And [RR97] shows how to obtain a circuit $M$ of size $n^{O(e(n))}$ such that
\[
\Pr_{|r|=2n} [M_L(y, r) = 1] - \Pr_{|x|=n} [M_L(y, G_{f_y}^{\text{HILL}}(x)) = 1] \geq 1/n^{e(n)}.
\]

By Theorem 3.7, there are constants $c_1, c_2, c_3, c_4$ and there is a circuit $N$ of size $n^{c_1e(n^{c_2})}$ such that the following holds for any $n$ and $y$:
\[
\Pr_{|x|=n} [F(y, N_L(y, F(y, x))) = F(y, x)] \geq n^{c_3e(n^{c_4})}
\]
where $x$ is chosen uniformly at random.

Now if we pick $c_0$ to be greater than max\{c_1, c_2, c_3, c_4\}, it follows that $N$ is a circuit of size less than $S(n)$ that inverts $f_y(x)$ with probability greater than $1/S(n)$, contrary to the definition of $S$.

This establishes that no solution to the $\text{Gap}_{1/e(n)}\text{MCSP}$ promise problem lies in $\text{P}/\text{poly}$. By Corollary 3.5, we conclude that $\text{Gap}_{1/e(n)}\text{MCSP}$ is $\text{NP}$-intermediate.

**Remark:** Observe that Theorem 3.6 can also be rephrased in terms of uniform probabilistic adversaries, if we assume that the one-way functions require time $n^{e(n)}$ to invert, for some easy-to-compute function $e$.

### 3.1 Other $\text{NP}$-intermediate problems.

Although our focus is primarily on $\text{MCSP}$, we observe here that the strongly downward self-reducibility property that we exploited above is fairly common. For instance, it has been noticed previously that $\text{CLIQUE}$ also has this property [Sri03, AK10]. It appears to be a new observation, however, that this property yields a natural $\text{NP}$-intermediate optimization problem.

**Definition 3.8.** For any function $\epsilon : \mathbb{N} \to (0, 1)$, let $\text{Gap}_\epsilon \text{CLIQUE}$ be the approximation problem that, given an $n$-vertex graph $G$, asks for outputting a value $f(G) \in \mathbb{N}$ such that
\[
\omega(G) \leq f(G) \leq n^{1-\epsilon(n)} \cdot \omega(G).
\]
Here, as usual $\omega(G)$ denotes the clique number of $G$: the size of the largest clique in $G$.

**Theorem 3.9.** $\text{NP} \not\subseteq \text{P}/\text{poly}$ if and only if there is an $\epsilon(n) = o(1)$ such that $\text{Gap}_\epsilon \text{CLIQUE}$ has no solution in $\text{P}/\text{poly}$ and is not hard for $\text{NP}$ under $\text{P}/\text{poly}$ reductions.

**Proof.** Assume $\text{NP} \not\subseteq \text{P}/\text{poly}$. Define $\epsilon(n)$ to be the least $c$ such that, for all $m \leq n$, there is a circuit of size $m^c$ that computes a function $f(G)$ (for $m$-vertex graphs $G$) such that $\omega(G) \leq f(G) \leq m^{1-1/c} \cdot \omega(G)$. If $\epsilon(n) = O(1)$, it follows from [Hás99] that $\text{CLIQUE} \in \text{P}/\text{poly}$, contrary to assumption. Thus $\epsilon(n) = o(1)$.

Let $\epsilon = \epsilon(n) = 1/\epsilon(n)$; thus $\epsilon(n) = o(1)$. It follows immediately from the definition of $\epsilon(n)$ that $\text{Gap}_\epsilon \text{CLIQUE}$ has no solution in $\text{P}/\text{poly}$.

If we partition the vertices of an $n$-node graph $G$ into $n^{1-\epsilon}$ parts $V_1, \ldots, V_{n^{1-\epsilon}}$ of size at most $[n^\epsilon]$, then $\omega(G) \leq (n^{1-\epsilon}) \cdot \max_i \omega(G_i)$, where $G_i$ is the induced subgraph of $G$ with vertices in $V_i$. (See [Sri03, AK10] for other applications of this observation.)
Now, precisely as in the proof of Theorem 3.3, it follows that if CLIQUE were P/poly-Turing reducible to Gap$_e$ CLIQUE, then CLIQUE $\in$ P/poly, contrary to our assumption. This shows that Gap$_e$ CLIQUE is not NP-hard under P/poly reductions, and thus completes the “only if” direction of the Theorem. (The converse is trivial.)

### 3.2 Reductions among GapMCSPs Require Large Stretch

In the previous section, we studied Gap$_e$ MCSP where $e(n) = o(1)$. In this section, we focus on the case where $e$ is a fixed positive constant. (When Gap$_e$ MCSP is considered as a “natural property”, this is the range of $e$ that would be considered when trying to prove a circuit size lower bound of $2^{\tilde{O}(n)}$. It is also the range of $e$ for which the hardness results of [AD17] hold.)

In what follows, we say that a reduction from Gap$_\delta$ MCSP to Gap$_e$ MCSP has stretch $n^c$ if, on input $T$, the reduction makes queries of length at most $|T|^c$.

**Theorem 3.10.** Let $0 < \epsilon < \delta < 1$. If Gap$_\delta$ MCSP is reducible to Gap$_e$ MCSP via a randomized Turing reduction of stretch at most $n^c$ for some $c < \delta/\epsilon$, then Gap$_\delta$ MCSP $\in$ BPP.

**Proof.** The argument is almost identical to the argument in the preceding section. Given an input to Gap$_\delta$ MCSP, simulate the reduction from Gap$_\delta$ MCSP to Gap$_e$ MCSP. As before, if the reduction makes a query $S$, then divide $S$ into consecutive substrings $S_1, \ldots, S_{2^k}$ of length $2^{m-k}$, where $m$ is defined as $|S| = 2^m$ and $k$ is a parameter chosen later depending on $m$. For each $i \in [2^k]$, recursively solve Gap$_\delta$ MCSP on the instance $S_i$, and let $f(S_i)$ be the answer of the recursive call. Now, we estimate the circuit complexity of $S$ as $2^k \cdot (\max_{i \in [2^k]} f(S_i) + 3)$ and continue the simulation.

We claim the correctness of the simulation for a certain choice of parameter $k = k(m).$ Let $e$ denote the estimated circuit complexity of $S$, that is, $e := 2^k \cdot (\max_{i \in [2^k]} f(S_i) + 3)$. The goal is to show that $e$ satisfies the promise of Gap$_e$ MCSP, or equivalently,

$$CC(S) \leq e \leq |S|^{1-\epsilon} \cdot CC(S).$$

We may assume that answers of recursive calls satisfy the promise of Gap$_{\delta}$ MCSP by induction: that is, $CC(S_i) \leq f(S_i) \leq |S_i|^{1-\delta} \cdot CC(S_i).$ Thus, by Lemma 3.4, we have

$$e \geq 2^k \cdot \left(\max_{i \in [2^k]} CC(S_i) + 3\right) \geq CC(S),$$

as required in the first inequality of (1). Now we turn to the second inequality of (1). We may assume, without loss of generality, that $e \leq 2^{k+1} \cdot \max_{i \in [2^k]} f(S_i)$. Therefore, we obtain

$$e \leq 2^{k+1} \cdot \max_{i \in [2^k]} f(S_i)$$

$$\leq 2^{k+1} \cdot \max_{i \in [2^k]} |S_i|^{1-\delta} \cdot CC(S_i) \quad \text{(by the promise of Gap$_{\delta}$ MCSP)}$$

$$= 2^{k+1} \max_{i \in [2^k]} \cdot \max CC(S_i)$$

$$\leq 2^{k+1+\min_{i \in [2^k]}(1-\delta)} \cdot CC(S)$$

$$\leq |S|^{1-\epsilon} \cdot CC(S),$$

where the last inequality holds if $k + 1 + (m - k)(1 - \delta) \leq m \cdot (1 - \epsilon)$, that is, $k \leq m - me/\delta - 1/\delta$. Thus we define $k$ as $k := m - me/\delta - 1/\delta$, which ensures the second inequality of (1).

Now we turn to analysis of the running time of the algorithm. Let $2^m$ be the length of the input to the algorithm. By the assumption on the stretch of the reduction, we have $|S| \leq 2^{nc}$, that is, $m \leq nc$. Therefore, $|S_i| = 2^{m-k} = 2^{me/\delta+1/\delta} \leq 2^{m+me/\delta+1/\delta}$. Since $me/\delta < 1$, the algorithm above runs in polynomial time. (Indeed, let $t(N)$ be an upper bound of the running time of the algorithm...
We now modify the ZPP elements. More precisely, let as well as the Gap theorem [AAR98], we will show that hardness under uniform reductions is closely related to lower bounds in circuit complexity, and in some cases we show that circuit lower bounds are equivalent to hardness results under uniform notions of reducibility (Theorem 4.7). These techniques allow us to prove the first results relating the complexity of MCSP\(^A\) and MKTP\(^A\) problems.

Here is the outline of this section. We will build on a randomized reduction of [AGvM+18, Section 3.1]: It is proved there that there is a ZPP reduction from the rigid\(^3\) graph isomorphism problem to MKTP. Here we modify that construction, to obtain a nonuniform AC\(^0\) reduction (Corollary 4.2). Combining Torán’s AC\(^0\) reduction [Tor04] from DET to the rigid graph isomorphism problem as well as the Gap theorem [AAR98], we will show DET\(\leq_m^{NC^0}\) MKTP (Theorem 4.3).

Next, we will establish that certain circuit size lower bounds are equivalent to the existence of certain uniform AC\(^0\) reductions to MKTP. This will be accomplished, by derandomizing the reduction of Theorem 4.3. Using this equivalence as a tool, we then close the section with a series of results presenting consequences of MKTP or MCSP being hard for various complexity classes, under different types of reducibility.

4 HARDNESS FOR DET

In this section, we present some of our main contributions. We show that MKTP is hard for DET under \(\leq_m^{NC^0}\) reductions (Theorem 4.3); prior to this, no variant of MCSP has been shown to be hard for any complexity class under any type of many-one reducibility. The \(\leq_m^{NC^0}\) reduction that we present is nonuniform; we show that hardness under uniform reductions is closely related to lower bounds in circuit complexity, and in some cases we show that circuit lower bounds are equivalent to hardness results under uniform notions of reducibility (Theorem 4.7). These techniques allow us to prove the first results relating the complexity of MCSP\(^A\) and MKTP\(^A\) problems.

4.1 Hardness of MKTP under nonuniform many-one reductions

We now modify the ZPP reduction of [AGvM+18, Section 3.1], which reduces the rigid graph isomorphism problem to MKTP, showing that it can be replaced by a nonuniform AC\(^0\) reduction.

**Lemma 4.1.** Let \(A\) be any oracle. There is a function \(f\) computable in Dlogtime-uniform AC\(^0\) such that, for any two rigid graphs \(G_0, G_1\) with \(n\) vertices:

- \(\Pr_{(r, G_0, G_1)}[f(G_0, G_1, r) \notin \text{MKTP}^A] > 1 - \frac{1}{2^{2n^k}}\) if \(G_0 \not\equiv G_1\), and
- \(\Pr_{(r, G_0, G_1)}[f(G_0, G_1, r) \in \text{MKTP}^A] = 1\) if \(G_0 \equiv G_1\).

**Proof.** We present the proof for \(A = \emptyset\); however, it is immediate that the proof carries over for any oracle \(A\). The function \(f\) is given by the reduction presented in [AGvM+18, Section 3.1], showing that the Rigid Graph Isomorphism Problem is in Promise-ZPP\(^{MKTP}\). This reduction takes graphs \(G_0\) and \(G_1\) as input, and interprets the random coin flip sequence \(r\) as a tuple \((w, \Pi)\) where \(\Pi\) is a sequence of \(t\) random permutations \(\pi_1, \ldots, \pi_t\), and \(|w| = t\).

We make use of a result of Hagerup [Hag91], showing that there is a function \(e\) computed by Dlogtime-uniform AC\(^0\) circuits,\(^4\) generating a nearly-uniform distribution on permutations of \(n\) elements. More precisely, let \(S_n\) denote the symmetric group on \([n]\), where permutation \(\sigma\) is represented as a binary string of the form \(\sigma(1) \ldots \sigma(n)\). Then for every \(\ell\) there is a \(k > \ell\) and a Dlogtime-uniform AC\(^0\)-computable function \(e : \{0, 1\}^n \rightarrow S_n \cup \{0^{n \log n}\}\) such that, for every \(\sigma \in S_n\)

\[
\Pr_{s \in \{0, 1\}^{n^k}}[e(s) = \sigma] \geq 1/n! - 2^{-n^k}.
\]

and \(\Pr_{s \in \{0, 1\}^{n^k}}[e(s) = 0^{n \log n}] \leq 2^{-n^k}\).

\(^3\)A graph is rigid if it has no nontrivial automorphisms.

\(^4\)Hagerup states this result in terms of CRCW PRAMs [Hag91] with a polynomial number of processors, running for \(O(1)\) steps. It is known [BIS90] that this class coincides with Dlogtime-uniform AC\(^0\).
Following the presentation in [AGvM⁺18], our AC⁰ reduction takes two graphs G₀ and G₁, along with a random string r = ws₁s₂ . . . sₜ where |w| = t = nΟ(1) and each sᵢ has length n̅, where k and ℓ (from the previous paragraph) are chosen so that 2⁻t/2 + t/2n̅ < 2⁻n². Thus, for a randomly-chosen r, with probability at least 1 - (t/2n̅), in AC⁰ we can compute the pair (w, Π) where Π = π₁, π₂, . . . πₜ = e(s₁), e(s₂), . . . e(sₜ). Next we compute the string xᵣ = π₁(G₁₀), . . . , πₜ(G₁ₜ). (With probability at most t/2n̅, some e(sᵢ) consists of a block of zeros, in which case our AC⁰ function will set xᵣ equal to a string of zeros, indicating failure.) We observe this is computable in AC⁰. Graphs are encoded as adjacency matrices. Thus, given a graph G and a permutation π, the bit (r, s) of π(G) is the same as the bit (i, j) in G, where π(i) = r and π(j) = s. That is, position (r, s) in the output is the ORᵢ,j (taken over all relevant positions (i, j) in the encoding of π) of [Gᵢ,j AND [the encoding of π contains the strings (i, r) and (j, s)]]. This latter condition can easily be checked in AC⁰.

The proof in [AGvM⁺18, Section 3.1] shows that, if G₀ \equiv G₁, then KT(xᵣ) ≤ t(log n!) + t/2.

On the other hand, [AGvM⁺18] observes that if G₀ \not\equiv G₁ then the entropy of the distribution on strings xᵣ (assuming t uniformly random permutations and a uniformly-randomly chosen string w) is at least t + t(log(n!)), and hence the probability that KT(xᵣ) < (t + t(log(n!)) - t/2 is at most 2⁻t/2. In our setting, the permutations are very nearly uniformly random (and it approaches the uniform distribution as ℓ increases), and there is also the possibility that xᵣ does not consist of t permuted graphs, but instead is all zeros. This latter condition arises with probability at most t/2n̅. Recalling that 2⁻t/2 + t/2n̅ < 2⁻n², we now have the following:

- If G₀ \equiv G₁, then KT(xᵣ) ≤ t(log n!) + t/2.
- If G₀ \not\equiv G₁, then with probability > 1 - 2⁻n², we have KT(xᵣ) ≥ t(log n!) + t/2.

We are now ready to define the AC⁰-computable function f: f(G₀, G₁, r) = (xᵣ, θ), where θ = t(log n!) + t/2. We have established that f has the desired properties. □

**Corollary 4.2.** Let A be any oracle. The rigid graph isomorphism problem is reducible to MKTP⁴ via a non-uniform \(\leq_{AC^0}^m\) reduction.

**Proof.** A standard counting argument shows that there is a value of r that can be hardwired into the probabilistic reduction of Lemma 4.1 that works correctly for all pairs (G₀, G₁) of n-vertex graphs. (Note that the length of the input is 2n², and the error probability is at most 1/2⁴n².) □

**Theorem 4.3.** Let A be any oracle. DET is reducible to MKTP⁴ via a non-uniform \(\leq_{AC^0}^m\) reduction. Furthermore, this reduction is “natural” in the sense of [KC00].

**Proof.** Since DET is closed under \(\leq_{AC^0}^m\) reductions, it suffices to show that MKTP⁴ is hard under \(\leq_{AC^0}^m\) reductions, and then appeal to the “Gap” theorem of [AAR98], to obtain hardness under \(\leq_{AC^0}^m\) reducibility. Torán [Tor04] shows that DET is AC⁰-reducible to GI. In fact it is shown in the proofs of Theorem 5.3 and Corollary 5.4 of [Tor04] that DET is AC⁰-reducible to GI via a reduction that produces only pairs of rigid graphs as output. Composing this reduction with the non-uniform AC⁰ reduction given by Corollary 4.2 completes the argument.

Since the same threshold θ is used for all inputs of the same length, the reduction is “natural”. □

An appeal to the circuit lower bounds of Razborov and Smolensky [Raz87, Smo87] now yields the following corollary:

**Corollary 4.4.** MKTP⁴ is not in AC⁰[p] for any oracle A and any prime p.

(An alternate proof of this circuit lower bound can be obtained by applying the pseudorandom generator of [FSUV13] that has sublinear stretch and is secure against AC⁰[p]; see [HS17], where
a stronger separation from AC$^0[p]$ is obtained in this way. Neither our argument nor those of [HS17, AGvM+18] seems easy to extend, to provide a lower bound for MCSP.)

4.2 Equivalence between hardness of MKTP and circuit lower bounds

The reader may wonder whether the non-uniform reduction can be made uniform under a suitable derandomization hypothesis. We do not know how to obtain a uniform AC$^0$-many-one reduction, but we can come close, if the oracle A is not too complex. Recall the definition of ctt-reductions: $B \leq_{\text{ctt}} C$ if there is a function $f \in C$ with the property that $f(x)$ is a list $f(x) = (y_1, \ldots, y_m)$, and $x \in B$ if and only if $y_j \in C$ for all $j$. Furthermore, we say that $f$ is a natural logspace-uniform $\leq_{\text{ctt}}$-reduction to MKTP if each query $y_j$ has the same length (and this length depends only on $|x|$), and furthermore each $y_j$ is of the form $(z_j, \theta)$ where the threshold $\theta$ depends only on $|x|$.

The following theorem can be viewed as a "partial converse" to results of [MW17, AHK17], which say that problems in LTH $\subseteq$ E require exponential size circuits if MCSP or MKTP is hard for TC$^0$ under Dlogtime-uniform $\leq_{\text{m}}$ reductions. That is, the earlier results show that very uniform hardness results imply circuit lower bounds, whereas the next theorem shows that somewhat stronger circuit lower bounds imply uniform hardness results (for a less-restrictive notion of uniformity, but hardness for a larger class). Later on, in Theorem 4.7, we present a related condition on reductions to MKTP$^A$ that is equivalent to circuit lower bounds.

**Theorem 4.5.** Let $A$ be any oracle. If there is some $\epsilon > 0$ such that $\text{DSpace}(n) \not\subseteq \text{io-SIZE}^{\text{MCSP}}(2^{\epsilon n})$, then every language in DET reduces to MKTP$^A$ via a natural logspace-uniform $\leq_{\text{ctt}}$-reduction.

**Proof.** Let $B \in \text{DET}$. Thus there is an AC$^0$ reduction $g$ reducing $B$ to the Rigid Graph Isomorphism Problem [Tor04]. Consider the following family of statistical tests $T_x(r)$, indexed by strings $x$:

- On input $r$:
  - Compute $z = f(g(x), r)$, where $f(G_0, G_1, r)$ is the function from Lemma 4.1.
  - Accept iff $(x \in B$ if $z \in \text{MKTP}^A)$.

Since $B \in \text{DET} \subseteq \text{P}$, the test $T_x(r)$ has a polynomial-size circuit with one MKTP$^A$ oracle gate. (In fact, the statistical test is an NC$^2$ circuit with one oracle gate.) If $x \in B$, then $T_x$ accepts every string $r$, whereas if $x \notin B$, $T_x$ accepts most strings $r$.

Klivans and van Melkebeek [KvM02] (building on the work of Impagliazzo and Wigderson [IW97]) show that, if DSPACE$(n)$ requires exponential-size circuits from a given class C, then there is a hitting set generator computable in logspace that hits all large sets computable by circuits from C that have size $n^k$. In particular, under the given assumption, there is a function $h$ computable in logspace such that $h(0^n) = (r_1, r_2, \ldots, r_{n^c})$ with the property that, for all strings $x$ of length $n$, there is an element of $h(0^n)$ that is accepted by $T_x$.

Now consider the logspace-uniform AC$^0$ oracle circuit family, where the circuit for inputs of length $n$ has the strings $h(0^n) = (r_1, r_2, \ldots, r_{n^c})$ hardwired into it. On input $x$, the circuit computes the queries $f(g(x), r_i)$ for $1 \leq i \leq n^c$, and accepts if, for all $i$, $f(g(x), r_i) \in \text{MKTP}^A$. Note that if $x \notin B$, then one of the $r_i$ is accepted by $T_x$, which means that $f(g(x), r_i) \notin \text{MKTP}^A$; if $x \in B$, then $f(g(x), r_i) \in \text{MKTP}^A$ for all $i$. This establishes that the reduction is correct.

It is also possible to prove a result analogous to Theorem 4.5, in terms of Dlogtime-uniform AC$^0$ reductions, in place of logspace-uniform AC$^0$ reductions. However, this requires a much stronger, average case circuit lower bound, for sets in LTH (as opposed to DSPACE$(n)$):

---

$^5$Recall that LTH = $\bigcup_k \Sigma_k \text{TIME}(O(n))$ is the linear-time analog of the polynomial time hierarchy.
THEOREM 4.6. Let $A$ be any oracle. There is a constant $c$ such that, if there is some $\epsilon > 0, b \geq 1$ and a set $B$ in the $d$-th level of LTH such that, for all large $n$ and every oracle circuit $C$ of size $2^{\epsilon n}$,

$$\Pr_{x \in \{0,1\}^n} [B(x) = C^{\text{MKTP}^A}(x)] < 1 - 1/n^b,$$

then every language in $\text{DET}$ reduces to $\text{MKTP}^A$ via a natural $D\text{logtime}$-uniform $\leq^\text{ctt}_c$-reduction of depth $d + c$.

Proof. The idea is similar to the proof of Theorem 4.5. Let $B \in \text{DET}$. We consider the same family of statistical tests $T_x(r)$.

Viola [Vio05, Theorem 4.3] shows that, under the hypothesis of Theorem 4.6, there is a pseudo-random generator $G : \{0, 1\}^{O(\log n)} \rightarrow \{0, 1\}^{n^c}$ that is secure against all statistical tests computable by circuits of size $n^c$. In particular, as in the proof of Theorem 4.5, we obtain a hitting set generator $h$. The depth required by the construction in [Vio05] is $d + O(1)$.

The rest of the proof proceeds precisely as in the proof of Theorem 4.5. \qed

We remark that the hardness assumption of Theorem 4.5 ($\text{DSPACE}(n) \not\subseteq \text{io-SIZE}^{\text{MKTP}^A}(2^{\epsilon n})$) can probably be weakened (saying that $\text{DSPACE}(n)$ requires large circuits of some restricted sort), since the class of statistical tests that need to be fooled consists only of $\text{NC}^2$ circuits with one oracle gate. On the other hand, Theorem 4.7 indicates that the hardness assumption that we use is equivalent to the existence of uniform reductions, for certain oracles $A$ – so it is not clear that there is much to be gained by searching for a weaker hardness assumption.

Theorem 4.5 deals with the oracle problem $\text{MKTP}^A$, but the most interesting case is the case where $A = \emptyset$, both because the hypothesis seems most plausible in that case, and because $\text{MKTP}$ has been studied in connection with $\text{MCSP}$, which has been studied more than the associated oracle circuit problem $\text{MCSP}^A$. The hypothesis is false when $A = \text{QBF}$, since the $\text{KT}^A$ measure is essentially the same as the KS measure studied in [ABK+06], where it is shown that $\text{PSPACE} = \text{ZPP}^\text{KS}$, and thus $\text{PSPACE}$ has polynomial-size $\text{MKTP}^{\text{QBF}}$-circuits. Strikingly, it is of interest that not only the hypothesis is false in this case – but the conclusion is false as well. (See Corollary 4.10.)

For certain oracles (and we discuss below how broad this class of oracles is), the existence of uniform reductions is equivalent to certain circuit lower bounds.

THEOREM 4.7. Let $\text{MKTP}^A \in \text{P}^A/\text{poly}$. Then the following are equivalent:

- PARITY reduces to $\text{MKTP}^A$ via a natural logspace-uniform $\leq^\text{ctt}_c$-reduction.
- For some $\epsilon > 0$, $\text{DSPACE}(n) \not\subseteq \text{io-SIZE}^A(2^{\epsilon n})$.
- For some $\epsilon > 0$, $\text{DSPACE}(n) \not\subseteq \text{io-SIZE}^{\text{MKTP}^A}(2^{\epsilon n})$.
- DET reduces to $\text{MKTP}^A$ via a natural logspace-uniform $\leq^\text{ctt}_c$-reduction.

Furthermore, if PARITY reduces to $\text{MCSP}^A$ via a natural logspace-uniform $\leq^\text{ctt}_c$-reduction, then all of the above hold.

Proof. First, we show that the first condition implies the second.

Let $\{C_n : n \in \mathbb{N}\}$ be a logspace-uniform family of oracle circuits computing PARITY, consisting of $\text{AC}^0$ circuitry feeding into oracle gates, which in turn are connected to an AND gate as the output gate. Let the oracle gates in $C_n$ be $g_1, g_2, \ldots, g_n$. On any input string $x$, let the value fed into gate $g_i$ on input $x$ be $(q_i(x), \theta)$, and recall that, since the reduction is natural, the threshold $\theta$ depends only on $n$, and thus it is a constant in $C_q$.

At this point, it is useful to recall a lemma from [AHK17] (distilled from [MW17]) that describes how the complexity depends on $\theta$:
Lemma 4.8. [AHK17, Claim 3.11] For any language A and any $0 \leq \nu \leq m$, MCSP$^A$ on inputs $f \in \{0, 1\}^m$, with the size parameter fixed to $\theta$, is solved by a DNF formula of size $m \cdot 2^{O(\theta^2 \log \theta)}$.

Thus, by Lemma 4.8, each MKTP$^\text{QBF}$ oracle gate can be replaced by a DNF formula of size at most $n^{O(1)}2^{O(\theta^2 \log \theta)}$. Inserting these DNF formulae into $C_n$ (in place of each oracle gate) results in a circuit of size $n^{O(1)}2^{O(\theta^2 \log \theta)}$ computing PARITY. Let the depth of this circuit be some constant $d$. It follows from [Hås87] that $n^{O(1)}2^{O(\theta^2 \log \theta)} \geq 2^{\Omega(n^{1/(d-1)})}$, and hence that $\theta \geq n^{1/d}$.

Note that all of the oracle gates $g_i$ must output 1 on input $0^n_i-1$, and one of the oracle gates $g_i$ must output 0 on input $0^n$. Thus we have $\text{KT}^A(q_{i_0}(0^n)) \geq \theta \geq n^{1/4d}$. It follows from [ABK’06, Theorem 11] that the function with truth table $q_{i_0}(0^n)$ has no circuit (with oracle gates for $A$) of size less than $(\text{KT}^A(q_{i_0}(0^n)))^{1/3} \geq 2^{O(1/3)} \geq n^{1/12d}$.

Note that, in order to compute the $j$-th bit of some query $q_i(0^n)$, it suffices to evaluate a logspace-uniform $\text{AC}^0$ circuit where all of the input bits are 0. Since this computation can be done in logspace on input $(0^n_i0^j)$, note that the language $H = \{(n, i, j) :$ the $j$-th bit of query $q_i(0^n)$ is 1} is in linear space. Let $m = \{(n, i, j)\}$, and let $s(m)$ be the size of the smallest circuit $D_m$ computing $H$ for inputs of length $m$. Hardwire the bits for $n$ and also set the bits for $i$ to $i_0$. The resulting circuit on $|j| < m$ bits computes the function given by $q_{i_0}(0^n)$, and it was observed above that this circuit has size at least $n^{1/20d} \geq 2^{m/20d}$.

This establishes the first implication. (Note also that a similar argument yields the same conclusion from the assumption that PARITY reduces to MCSP$^A$ via a natural logspace-uniform $\preceq_{\text{ctt}}$-reduction.)

The assumption that MKTP$^A \in \text{P}^A$/poly suffices to show that the second condition implies the third. More formally, we’ll consider the contrapositive. Assume that DSPACE$(n) \subseteq \text{io-SIZE}^\text{MKTP}^A(2^\epsilon n)$ for every $\epsilon > 0$. An oracle gate for MKTP$^A$ on inputs of size $m$ can be replaced by a circuit (with oracle gates for $A$) of size $m^c$ for some constant $c$. Carrying out this substitution in a circuit (with oracle gates for MKTP$^A$) of size $2^\epsilon n$ yields a circuit of size at most $2^\epsilon n + 2^\epsilon n(2^\epsilon n)^c$.

Let $\delta > 0$. Then we can pick $\epsilon$ small enough so that $2^\epsilon n + 2^\epsilon n(2^\epsilon n)^c < 2^\delta n$, thereby establishing that DSPACE$(n) \subseteq \text{io-SIZE}^A(2^{\delta n})$ for every $\delta > 0$. This establishes the second implication.

Theorem 4.5 establishes that the third condition implies the fourth. The fourth condition obviously implies the first.\[\square\]

To the best of our knowledge, this is the first theorem that has given conditions where the existence of a reduction to MCSP$^A$ implies the existence of a reduction to MKTP$^A$. We know of no instance where the implication goes in the opposite direction.

The logspace uniformity condition in Theorem 4.7 can be replaced by other less-restrictive uniformity conditions. We mention the following example:

**Corollary 4.9.** Let MKTP$^A \in \text{P}^A$/poly. Then the following are equivalent:

- PARITY reduces to MKTP$^A$ via a natural $\text{P}$-uniform $\preceq_{\text{ctt}}$-reduction.
- For some $\epsilon > 0$, $E \not\subseteq \text{io-SIZE}^A(2^\epsilon n)$.
- For some $\epsilon > 0$, $E \not\subseteq \text{io-SIZE}^\text{MKTP}^A(2^\epsilon n)$.
- DET reduces to MKTP$^A$ via a natural $\text{P}$-uniform $\preceq_{\text{ctt}}$-reduction.

Furthermore, if PARITY reduces to MCSP$^A$ via a natural $\text{P}$-uniform $\preceq_{\text{ctt}}$-reduction, then all of the above hold.

At this point, we should consider the class of oracles for which Theorem 4.7 applies. That is, what is the set of oracles $A$ for which MKTP$^A \in \text{P}^A$/poly? First, we observe that this condition holds for any PSPACE-complete set, which yields the following corollary:
COROLLARY 4.10. PARITY does not reduce to either MKTP\(^{QBF}\) or MCSP\(^{QBF}\) via a natural logspace-uniform \(\leq_{ctt}^{AC^0}\)-reduction.

Remark: As an instructive example of a set for which MKTP\(^A\) \(\in P^A/poly\), consider the set \(A = \{(M, x, 1^m) : M\) is an alternating Turing machine that accepts \(x\), and runs in time at most \(m\) and makes at most \(\log m\) alternations\}. \(A\) is complete for the class ATIME-ALT\((n^{O(1)}, O(\log n))\) under \(\leq_{\text{m}AC}\)-reductions. It is easy to see that MKTP\(^A\) \(\in ATIME-ALT(n^{O(1)}, O(\log n))\), and thus MKTP\(^A\) \(\in P^A\). (Other examples can easily be created in this way, using an even smaller number of alternations.) Note that, for this oracle \(A\), it seems plausible that all four conditions in Theorem 4.7 hold.

Nonetheless, we do grant that this does seem to be a strong condition to place upon the oracle \(A\) – and it has even stronger consequences than are listed in Theorem 4.7. For instance, note that the proof that the first condition in Theorem 4.7 implies the second relies only on the fact that PARITY requires large \(AC^0\) circuits. Thus, an identical proof shows that these four conditions are also equivalent to the condition that PARITY is reducible to MKTP\(^A\) via a natural ctt-reduction where the queries are computed by logspace-uniform \(AC^0[7]\) circuits. (Or you can substitute any other problem and class of mod circuits, where an exponential lower bound is known because of [Raz87, Smo87].) In fact, as in the proof of [AHK17, Lemma 3.10] we can apply random restrictions in a logspace-uniform way (as described in [Agr11]) and obtain a reduction from PARITY to MKTP\(^A\) where the queries are computed by logspace-uniform \(NC^0\) circuits! For example, here is an argument showing that MAJORITY is reducible to MKTP\(^A\) (for oracle \(A\) satisfying the hypotheses of Theorem 4.7) via natural ctt-reductions computed by logspace-uniform \(AC^0[3]\) circuits iff PARITY is reducible to MKTP\(^A\) via reductions where the queries are computed by logspace-uniform \(NC^0\) circuits:

Assume first that MAJORITY is reducible to MKTP\(^A\) via natural ctt-reductions computed by logspace-uniform \(AC^0[3]\) circuits. The proof that the first condition in Theorem 4.7 implies the second also shows that the second condition holds if MAJORITY is reducible to MKTP\(^A\) via natural ctt-reductions computed by logspace-uniform \(AC^0[3]\) circuits. (The only things that need to be changed, are (1) every occurrence of “PARITY” should be changed to MAJORITY” (2) the phrase “consisting of \(AC^0\) circuitry feeding into oracle gates” should be changed to “consisting of \(AC^0[3]\) circuitry feeding into oracle gates”, and (3) “Note that all of the oracle gates \(g_i\) must output 1 on input \(0^{n^\epsilon}1^n\)” should be replaced by “Note that all of the oracle gates \(g_i\) must output 1 on input \(1^{n^\epsilon}\)”. Thus, under our assumption, all four of the conditions in Theorem 4.7 hold. In particular, PARITY reduces to MKTP\(^A\) via a natural logspace-uniform \(\leq_{ctt}^{AC^0}\)-reduction. The \(AC^0\)-computable function \(f\) that computes the list of oracle queries has the property that there is a logspace-computable restriction \(\rho\) that leaves \(n^\epsilon\) input variables unset (for some \(\epsilon > 0\)) with the property that the function \(f|\rho\) on \(n^\epsilon\) variables is \(NC^0\)-computable. (See, e.g., [AAR98, Lemma 7] and see [Agr11] to see how \(\rho\) can be computed in logspace.) This yields the claimed reduction from PARITY to MKTP\(^A\) where the queries are \(NC^0\)-computable.

Conversely, assume now that PARITY is reducible to MKTP\(^A\) via a natural \(\leq_{ctt}^{AC^0}\)-reduction where the queries are computed by logspace-uniform \(NC^0\) circuits. This is a stronger condition than the first condition on Theorem 4.7, and hence all four of these conditions hold. In particular, DET reduces to MKTP\(^A\) via \(\leq_{ctt}^{AC^0}\) reductions. The claim now follows, since MAJORITY \(\in\) DET.

We find these implications to be surprising. The “gap” phenomenon that was described in [AAR98] (showing that completeness under one class of reductions is equivalent to completeness under a more restrictive class of reductions) had not previously been observed to apply to \(AC^0[p]\) reducibility.
We want to highlight some contrasts between Theorem 4.5 and Corollary 4.10. MKTP\textsuperscript{QBF} is hard for PSPACE under ZPP-Turing reductions [ABK\textsuperscript{+}06], whereas MKTP is in NP. Thus MKTP\textsuperscript{QBF} appears to be much harder than MKTP. Yet, Theorem 4.5 shows that, under a plausible hypothesis, the “easier” set MKTP is hard for DET, whereas (by Corollary 4.10) the “harder” problem MKTP\textsuperscript{QBF} cannot even be used as an oracle for PARITY under this same reducibility.

In other words, the (conditional) natural logspace-uniform \( \leq_{\text{tt}} \) reductions from problems in DET to MKTP given in Theorem 4.5 are not “oracle independent” in the sense of [HW16]. Prior to this work, there had been no reduction to MCSP or MKTP that did not work for every MCSP\textsuperscript{A} or MKTP\textsuperscript{A}, respectively.

Prior to this work, it appears that there was no evidence for any variant of MCSP or MKTP being hard for a reasonable complexity class under \( \leq_{\text{tt}} \) reductions. All prior reductions (such as those in [AD17, ABK\textsuperscript{+}06, AGvM\textsuperscript{+}18]) had been probabilistic and/or non-uniform, or (even under derandomization hypotheses) seemed difficult to implement in NC. But Theorem 4.7 shows that it is quite likely that MKTP is hard for DET under \( \leq_{\text{tt}} \) reductions (and even under much more restrictive reductions). Previously, we had viewed the results of [AHK17] as providing evidence that none of these variants would be hard for P under, say, logspace reducibility. Now, we are no longer sure what to expect.

### 4.3 On the importance of uniformity

Surprisingly (to us), the notion of uniformity appears to be central. In particular, the reader is probably wondering whether the logspace-uniformity condition in Theorem 4.5 (relating hardness of MKTP\textsuperscript{A} to worst-case circuit lower bounds) can be improved to Dlogtime-uniformity. As a partial answer to this question, we note that Viola [Vio05] shows that there is no black-box construction of a pseudorandom generator computable in AC\textsuperscript{0} that is based on worst-case circuit lower bounds. In this section, in Theorem 4.12, we show that, when considering hardness of MKTP and MCSP, small details about the complexity of the reduction (including the precise depth, and the notion of uniformity) cannot be ignored.

First, we recall Corollary 3.7 of [AHK17], which states that MKTP\textsuperscript{QBF} is not hard for P under \( \leq_{\text{tt}} \) reductions unless PSPACE = EXP. It turns out that this holds even for logspace-Turing reductions.

**Theorem 4.11.** MKTP\textsuperscript{QBF} is not hard for P (or NP) under \( \leq_{\text{tt}} \) reductions unless PSPACE = EXP (PSPACE = NEXP, respectively). MKTP\textsuperscript{QBF} is not hard for PSPACE under \( \leq_{\text{tt}} \) reductions. The same holds for MCSP\textsuperscript{QBF}.

We include this proof here, both because it improves a Corollary in [AHK17], and because the proof can be viewed as a warm-up for the proof of Theorem 4.12.

**Proof.** First, note that \( \leq_{\text{tt}} \) and \( \leq_{\text{it}} \) reducibilities coincide [LL76]. Thus assume that MKTP\textsuperscript{QBF} is hard for P under \( \leq_{\text{it}} \) reductions; we will show that PSPACE = EXP. (The proof for MCSP\textsuperscript{QBF} is identical, and the variant concerning hardness for NP is analogous.)

The proof idea is based on [HW16]: Assume that P \( \subseteq \text{L}_{\text{tt}}^{\text{MKTPQBF}} \). (Here, L\textsubscript{tt} means a \( \leq_{\text{tt}} \) reduction.) By standard padding, we obtain EXP \( \subseteq \text{PSPACE}_{\text{tt}}^{\text{MKTPQBF}} \). Any query of a PSPACE\textsubscript{tt} machine has low KT\textsuperscript{QBF} complexity. Moreover, one can check whether a string has low KT\textsuperscript{QBF} complexity in PSPACE. Combining these two facts, we obtain EXP \( \subseteq \text{PSPACE}_{\text{tt}}^{\text{MKTPQBF}} = \text{PSPACE} \). A formal proof follows.

Let B \( \in \text{EXP} \). Let \( B' = \{ x10^{2|x|} : x \in B \} \) and note that \( B' \in \text{P} \). Consider the \( \leq_{\text{tt}} \) reduction that reduces \( B' \) to MKTP\textsuperscript{A}. On any input string \( y \), let the \( i \)-th oracle query be \( q_i(y) \). The language \( \{ (i, j, x) \)
the $j$-th bit of $q_i(x10^{2|x|})$ is 1} is in PSPACE and thus is in $P^{QBF}$. It follows that $q_i(x10^{2|x|})$ is of the form $(y_i, \theta_i)$, where $K^QBF(y_i) = |x, i, j|^{O(1)}$. Thus, to evaluate the oracle query $q_i$ on input $x10^{2|x|}$, this PSPACE computation (on input $x$) suffices: Compute the bits of $\theta_i$; this can be done in PSPACE, since the number of bits in $\theta_i$ is at most $|x|^{O(1)}$, and each bit is computable in PSPACE. If $\theta_i > |x, i, j|^{c}$ (for the appropriate value of $c$), then return “1” since the query $y_i$ certainly has $K^A$ complexity less than this. Otherwise, try all descriptions $d$ of length at most $\theta_i$, to determine whether there is some such $d$ for which $U^{QBF}(d, j)$ is equal to the $j$-th bit of $q_i$ (allowing at most $|x, i, j|^{c}$ steps for the computation of $U$).

The rest of the $\leq_{R}$ reduction on input $x10^{2|x|}$ can be computed in space $|x|^{O(1)}$, by re-computing the values of the oracle queries, as required.

The unconditional result that MKTP$^{QBF}$ is not hard for PSPACE under $\leq_{R}$ reductions follows along the same lines, choosing $B \in EXPSPACE$, and leading to the contradiction EXPSPACE = PSPACE.

A similar approach yields the following result:

**Theorem 4.12.** For each $d \geq 0$, if $\Sigma^p_{d+2} \subseteq \text{P}^{A}/\text{poly}$ and PSPACE $\nsubseteq$ PH$^A$, then neither MKTP$^A$ nor MCSP$^A$ is hard for NC$^1$ under Dlogtime-uniform $\leq_{\text{it}}^{AC^0}$ reductions of depth $d$.

**Proof.** We present the proof for MKTP$^A$; the proof for MCSP$^A$ is identical.

Assume that MKTP$^A$ is hard for NC$^1$ under Dlogtime-uniform $\leq_{\text{it}}^{AC^0}$ reductions of depth $d$; we will show that PSPACE $\nsubseteq$ PH$^A$.

By the closure properties of PH, it will suffice to show that ATIME$(n) \nsubseteq$ PH$^A$.

Let $B \in$ ATIME$(n)$. Let $B' = \{x10^{2|x|} : x \in B\}$ and note that $B' \in$ NC$^1$. Consider the oracle family $(C_m)$ that reduces $B'$ to MKTP$^A$. Let the oracle gates in $C_{2^{n+2}+1}$ be $g_1, g_2, \ldots, g_{\ell}$. On any input string $y$, let the query that is fed into gate $g_i$ be $q_i(y)$. The language $\{(2|x| + |x| + 1, i, j, x) :$ the $j$-th bit of $q_i(x10^{2|x|})$ is 1$\}$ is in $\Sigma^p_{d+2}$ and thus is in P$^A$/poly. It follows that $q_i(x10^{2|x|})$ is of the form $(y_i, \theta_i)$, where $K^A(y_i) = |x, i, j|^{O(1)}$. Thus, to evaluate oracle gate $g_i$ on input $x10^{2|x|}$, this PH$^A$ computation (on input $x$) suffices: Compute the bits of $\theta_i$; this can be done in PH, since the number of bits in $\theta_i$ is at most $|x|^{O(1)}$, and each bit is computable in PH. If $\theta_i > |x, i, j|^{c}$ (for the appropriate value of $c$), then return “1” since the query $y_i$ certainly has $K^A$ complexity less than this. Otherwise, guess a description $d$ of length at most $\theta_i$, and universally check (for each $j$ that $U^A(d, j)$ is equal to the $j$-th bit of $q_i$ (allowing at most $|x, i, j|^{c}$ steps for the computation of $U$).

To evaluate the rest of the circuit, note that the unbounded fan-in AND and OR gates that sit just above the oracle gates can also be evaluated in PH$^A$ (at one level higher in the hierarchy than is required to evaluate the oracle gates). Repeating this process through the remaining $O(1)$ levels of the circuit yields the desired PH$^A$ algorithm for $B$.\hfill \square

**Remark:** The significance of Theorem 4.12 is best viewed by combining it with Theorem 4.5. If we choose $A$ to be any PP-complete set, or if we choose $A$ to be one of the sets discussed in the Remark after Corollary 4.10, then for all $d$ we have $\Sigma^p_{d+2} \subseteq \text{P}^{A}$ and both of the hypotheses

- PSPACE $\nsubseteq$ PH$^A$, and
- DSPACE$(n) \nsubseteq$ io-SIZE$^{MKTP^A}(2^{\varepsilon n})$

are plausible. Thus, for such oracles $A$, under a plausible hypothesis, we have both MKTP$^A$ is not hard for NC$^1$ under Dlogtime-uniform $\leq_{\text{it}}^{AC^0}$ reductions, and MKTP$^A$ is hard for DET under logspace-uniform $\leq_{\text{it}}^{C^\text{tt}}$ reductions. Thus different notions of uniformity are a key part of the puzzle, when trying to understand the hardness of problems such as MKTP and MCSP.
As another example, choose $A$ to be any set that is complete for $\Sigma^P_{d+2}$, and assume $\text{PSPACE} \neq \text{PH}$. Then under the strong-but-plausible hypothesis that there is a set $B \in \Sigma^P_d \text{TIME}(n)$ that has large symmetric difference with any set in $\text{SIZE}^{\text{MKTP}^A}(2^n)$, we have $\Sigma^P_{d+2} \subseteq P^A$ and $\text{PSPACE} \not\subseteq \text{PH}^A = \text{PH}$, thereby satisfying the hypotheses of both Theorem 4.12 and Theorem 4.6. Thus, for this choice of $A$, under a plausible hypothesis, we have both $\text{MKTP}^A$ is not hard for $\text{NC}^1$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions of depth $d$, and $\text{MKTP}^A$ is hard for $\text{DET}$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions of depth $d' + c$ (where $c$ is the constant from Theorem 4.6).

In both of these examples, the key ingredient seems to be that, in order for $\text{AC}^0$ to be able to reduce problems to $\text{MCSP}^A$ or $\text{MKTP}^A$, it is essential to be able to formulate useful queries, by either having sufficient depth, or by having sufficient power in the uniformity condition.

We are even able to extend our approach in some cases, to apply to $\text{AC}^0$-Turing reducibility.

**Theorem 4.13.** Let $\text{NP}^A \subseteq P^A/\text{poly}$. If $\text{PSPACE} \not\subseteq \text{PH}^A$, then neither $\text{MKTP}^A$ nor $\text{MCSP}^A$ is hard for $\text{NC}^1$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions.

**Proof.** The proof is similar to that of Theorem 4.12. Assume that $\text{MKTP}^A$ is hard for $\text{NC}^1$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions; we will show that $\text{ATIME}(n) \subseteq \text{PH}^A$ by presenting a $\text{PH}^A$ algorithm to evaluate the gates in the $\leq^{\text{AC}^0}_T$ reduction of the $\text{NC}^1$ language $B'$ from the proof of Theorem 4.12.

Note that in a circuit computing an $\leq^{\text{AC}^0}_T$ reduction, there is an “initial” layer of oracle gates, whose queries are computed nonadaptively, while all oracle gates at deeper levels have queries whose values depend upon oracle gates at earlier levels in the circuit. Note also that, under the given assumption $\text{NP}^A \subseteq P^A/\text{poly}$, we can conclude that $\text{PH}^A \subseteq P^A/\text{poly}$.

The proof now proceeds along precisely the same lines as the proof of Theorem 4.12, which shows that a $\text{PH}^A$ computation can compute the value of each wire that feeds into the “initial” layer of oracle gates. Similarly, as in the proof of Theorem 4.12, all of the AND, OR, and NOT gates at higher levels can be computed in $\text{PH}^A$, given that the gates at lower levels can be evaluated in $\text{PH}^A$. Thus, we need only show how to deal with oracle gates at deeper levels.

Consider any such oracle gate $g$. On any input string $y$, let the query that is fed into gate $g$ when evaluating the circuit on input $y$ be $q_g(y)$. The language $\{2^{\lceil |y| \rceil} + |x| + 1, g, j, x\}$: the $j$-th bit of $q_g(x, 10^{|x|})$ is 1} is in $\text{PH}^A$ and thus (by our new assumption) is in $P^A/\text{poly}$. It follows that $q_g(x, 10^{|x|})$ is of the form $(y, \theta)$, where $\text{KT}^A(y) = |x, g, j|^{O(1)}$. Thus, to evaluate oracle gate $g$ on input $x, 10^{|x|}$, this $\text{PH}^A$ computation (on input $x$) suffices: Compute the bits of $\theta$; this can be done in $\text{PH}^A$, since the number of bits in $\theta$ is at most $|x|^{O(1)}$, and each bit is computable in $\text{PH}$. If $\theta > |x, g, j|$ (for the appropriate value of $c$), then return “1” since the query $y$ certainly has $\text{KT}^A$ complexity less than this. Otherwise, guess a description $d$ of length at most $\theta$, and universally check (for each $j$) that $U^A(d, j)$ is equal to the $j$-th bit of $q_g$ (allowing at most $|x, i, j|^c$ steps for the computation of $U$).

In order to compare our results with those of [AHK17, MW17], we also state a related theorem, whose proof is similar:

**Theorem 4.14.** Let $\text{NP}^A \subseteq P^A/\text{poly}$. If $\text{CH} \not\subseteq \text{PH}^A$, then neither $\text{MKTP}^A$ nor $\text{MCSP}^A$ is hard for $\text{TC}^0$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions.

**Proof.** The proof is nearly identical to that of Theorem 4.13.

Under the assumption that $\text{MKTP}^A$ is hard for $\text{TC}^0$ under $\text{Dlogtime-uniform } \leq^{\text{AC}^0}_T$ reductions, it suffices to show that the linear-time counting hierarchy (see [AKR+01] for a definition) is contained in $\text{PH}^A$ by presenting a $\text{PH}^A$ algorithm for a set $B$ in the linear-time counting hierarchy. The language $B'$ is now in $\text{TC}^0$, instead of merely being in $\text{NC}^1$. The rest of the proof proceeds virtually
unchanged. (One can modify the statement of Theorem 4.12 in a similar way, but we do not include this modification here.)

A consequence of Theorems 4.13 and 4.14 is the following corollary, which has the same flavor of results of the form "MCSP is hard for class C implies a likely but hard-to-prove consequence" as presented by Murray and Williams [MW17], but moving beyond the $\leq_m^{\AC^0}$ reductions considered by them, to the more general $\leq_T^{\AC^0}$ reductions.

**Corollary 4.15.** If either of MKTP or MCSP is hard for NC$^1$ (or TC$^0$) under Dlogtime-uniform $\leq_T^{\AC^0}$ reductions, then NP $\neq$ NC (NP $\neq$ TC$^0$, respectively).

**Proof.** This follows from Theorems 4.13 and 4.14 when $A = \emptyset$. If NP $\subseteq$ P/poly, and PH $\neq$ NC $\neq$ PSPACE. Thus neither MKTP nor MCSP is hard for NC$^1$ under Dlogtime-uniform $\leq_m^{\AC^0}$ reductions. Also, if NP $= \text{TC}^0$, then NP $\subseteq$ P/poly, and PH $= \text{TC}^0$ $\neq$ CH [All99]. Thus neither MKTP nor MCSP is hard for TC$^0$ under Dlogtime-uniform $\leq_T^{\AC^0}$ reductions.

Corollary 4.15 should be compared to the earlier work of [MW17, AHK17]. Murray and Williams presented nonuniform lower bounds that would follow from MCSP or MKTP being hard for NP under Dlogtime-uniform $\leq_m^{\AC^0}$ reductions. In [AHK17] even stronger nonuniform consequences were shown to follow from the weaker assumption of hardness for TC$^0$. (See Table 2.) In Corollary 4.13, we present a weaker uniform lower bound that follows from the weaker assumption that MCSP or MKTP is hard for TC$^0$ under a more powerful notion of reducibility.

We also present another result in this vein, about NP-completeness. Prior work [MW17, AHK17] had obtained stronger consequences from the stronger assumption that MCSP is NP-complete under Dlogtime-uniform $\leq_m^{\AC^0}$ reductions.

**Corollary 4.16.** If either of MKTP or MCSP is hard for NP under Dlogtime-uniform $\leq_T^{\AC^0}$ reductions, then

$$\text{NP} \neq \text{MA} \cap \text{P/poly}.$$  

**Proof.** If you modify the proof of Theorem 4.13, replacing NC$^1$ by NP and replacing PSPACE by NEXP, you obtain that, if NP $\subseteq$ P/poly, then NEXP $\neq$ PH implies that neither MKTP nor MCSP is hard for NP under Dlogtime-uniform $\leq_T^{\AC^0}$ reductions. (That is, if we assume that MKTP is hard for NP under Dlogtime-uniform $\leq_T^{\AC^0}$ reductions, then the argument from Theorem 4.13 shows that NEXP $\subseteq$ PH, by presenting a PH algorithm to evaluate the gates in an AC$^0$ oracle circuit reducing an NP language $B'$ to MKTP.)

Or, restating this using the same hypothesis as in the statement of the corollary, if MKTP or MCSP is hard for NP under Dlogtime-uniform $\leq_T^{\AC^0}$, then either NP $\not\subseteq$ P/poly or NEXP $=$ PH. Since (NP $\subseteq$ P/poly and NEXP $=$ PH) is equivalent to NEXP $\subseteq$ P/poly, and since NEXP $\subseteq$ P/poly is equivalent to NEXP $= \text{MA}$ [IKW02], we obtain that NP-hardness of MCSP or MKTP implies NP $\not\subseteq$ P/poly or NEXP $= \text{MA}$. (Murray and Williams obtain essentially this same consequence under the stronger assumption that MCSP is complete under $\leq_m^{\AC^0}$ reductions, but are also able to show that NEXP $\not\subseteq$ P/poly in this case.)

In either case, we obtain the consequence NP $\neq$ MA $\cap$ P/poly.

We close this section with another variant of Theorem 4.13, proved via the same technique:

**Theorem 4.17.** Let NP$^A$ $\subseteq$ P$^A$/poly. If NEXP $\not\subseteq$ PSPACE$^A$ (or NEXP $\not\subseteq$ EXP$^A$), then neither MKTP$^A$ nor MCSP$^A$ is hard for NP under logspace-uniform $\leq_T^{\AC^0}$ reductions (P-uniform $\leq_T^{\AC^0}$ reductions, respectively).
Corollary 4.18. \( \text{MKTP}^{\text{QBF}} \) is not hard for \( \text{NP} \) under logspace-uniform \( \leq_{T}^{C_{0}} \) reductions (P-uniform \( \leq_{T}^{C_{0}} \) reductions) unless \( \text{PSPACE} = \text{NEXP} \) (\( \text{EXP} = \text{NEXP} \), respectively). The same holds for \( \text{MCSP}^{\text{QBF}} \).

Although the following corollary discusses \( \leq_{T}^{C_{0}} \) reductions, it also says something about \( \leq_{T}^{L} \) reducibility. This is because, assuming \( \text{DSPACE}(n) \not\subseteq \text{io-SIZE}^{\text{MKTP}}(2^{\epsilon n}) \), any \( \leq_{T}^{L} \) reduction to \( \text{MKTP} \) can be simulated by a logspace-uniform \( \leq_{T}^{AC_{0}} \) reduction to \( \text{MKTP} \). (To see this, note that, by Theorem 4.5, \( \text{MKTP} \) is hard for \( \text{DET} \) under this class of reductions, and hence each of the logspace-computable (nonadaptive) queries can be computed using oracle gates for \( \text{MKTP} \), and similarly the logspace computation that uses the queries can also be simulated using \( \text{MKTP} \). Similar observations arise in [AO96].)

Corollary 4.19. If either of \( \text{MKTP} \) or \( \text{MCSP} \) is hard for \( \text{NP} \) under logspace-uniform \( \leq_{T}^{C_{0}} \) reductions (P-uniform \( \leq_{T}^{C_{0}} \) reductions), then \( \text{NP} \not\subseteq \text{P/poly} \) or \( \text{NEXP} = \text{PSPACE} \) (\( \text{NEXP} = \text{EXP} \), respectively).

5 CONCLUSIONS AND OPEN QUESTIONS

Conclusions. At a high level, we have advanced our understanding about \( \text{MCSP} \) and \( \text{MKTP} \) in the following two respects:

(1) On one hand, under a very weak cryptographic assumption, the problem of approximating \( \text{MCSP} \) or \( \text{MKTP} \) is indeed \( \text{NP} \)–intermediate under general types of reductions when the approximation factor is quite huge. This complements the work of [MW17] for very restricted reductions.

(2) On the other hand, if the gap is small, \( \text{MKTP} \) is \( \text{DET} \)–hard under nonuniform \( \text{NC}_{0} \) reductions (contrary to previous expectations). This suggests that nonuniform reductions are crucial to understanding hardness of \( \text{MCSP} \). While there are many results showing that \( \text{NP} \)-hardness of \( \text{MCSP} \) under uniform reductions is as difficult as proving circuit lower bounds, can one show that \( \text{MCSP} \) is \( \text{NP} \)-hard under \( \text{P/poly} \) reductions (without proving circuit lower bounds)?

Open Questions. It should be possible to prove unconditionally that \( \text{MCSP} \) is not in \( \text{AC}_{0}[2] \); we conjecture that the hardness results that we are able to prove for \( \text{MKTP} \) hold also for \( \text{MCSP} \). (We refer the reader to [AGvM*18, Section 7] for a more detailed discussion about the obstacles that need to be overcome, in order to extend these theorems to \( \text{MCSP} \).)

We suspect that it should be possible to prove more general results of the form “If \( \text{MCSP}^{A} \) is hard for class \( C \), then so is \( \text{MKTP}^{A} \).” We view Theorem 4.7 to be just a first step in this direction. One way to prove such a result would be to show that \( \text{MCSP}^{A} \) reduces to \( \text{MKTP}^{A} \), but (with a few exceptions such as \( A = \text{QBF} \)) no such reduction is known. Of course, the case \( A = \emptyset \) is the most interesting case.

Is \( \text{MKTP} \) hard for \( \text{P} \)? Or for some class between \( \text{DET} \) and \( \text{P} \)? Is it more than a coincidence that \( \text{DET} \) arises both in this investigation of \( \text{MKTP} \) and in the work of Oliveira and Santhanam on \( \text{MCSP} \) [OS17]?

Is there evidence that \( \text{Gap}_{\epsilon} \text{MCSP} \) has intermediate complexity when \( \epsilon \) is a fixed constant, similar to the evidence that we present for the case when \( \epsilon(n) = o(1) \)?

ACKNOWLEDGMENTS

We thank Ryan Williams, Rahul Santhanam, Salil Vadhan, Marina Knittel, Rahul Ilango, and Prashant Nalini Vasudevan for helpful discussions. We thank Emanuele Viola for calling our attention to the work of Hagerup [Hag91]. We thank the anonymous referees for their careful reading and helpful suggestions. This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing. E. A. was supported by National Science Foundation grants CCF-1514164
and CCF-1555409; some of the work was done while this author was visiting the Institute for Mathematical Sciences, National University of Singapore in 2017, on a visit that was supported by the Institute. S.H. was supported by JSPS KAKENHI Grant Number JP16J06743.

REFERENCES


Received January 2018