Electronic Colloquium on Computational Complexity, Report No. 220 (2025)

A Note on Avoid vs MCSP

Edward A. Hirsch* Ilya Volkovich†

Abstract

A recent result of Ghentiyala, Li, and Stephens-Davidowitz (ECCC TR 25-210) shows that any language reducible to the Range Avoidance Problem (Avoid) via deterministic or randomized Turing reductions is contained in $\mathsf{AM} \cap \mathsf{coAM}$. In this note, we present a different potential avenue for obtaining the same result via the Minimal Circuit Size Problem (MCSP).

1 Introduction

The *Range Avoidance* (Avoid) problem is defined as follows: given a Boolean circuit $C : \{0,1\}^n \to \{0,1\}^m$ with $m > n$, output a string $y \in \{0,1\}^m$ such that $y \notin \mathrm{Im}\,(C)$, i.e., such that y does not have a pre-image w.r.t. C. The problem has been studied since 1980s in mathematical logic literature [PWW88], where it is referred to as the "dual Weak Pigeonhole Principle" (dWPHP). Its "new life" in the computational complexity theory has begun with the work of Kleinberg, Korten, Mitropolsky, and Papadimitriou [KKMP21]. Since then, the problem attracted significant attention (see e.g. [GLSD25] and the references therein for a survey from the complexity-theoretic perspective, and [Kra25] for the body of work in the field of bounded arithmetic).

Avoid is a natural example of a total search problem, that is, a search problem for which a solution always exists. Furthermore, a randomly selected element of $\{0,1\}^m$ is a solution with probability at least $1 - 2^{-(m-n)}$. Thus, the hardest instances[1] occur when $m = n+1$, in which case the trivial randomized algorithm could succeed with probability close to $\frac{1}{2}$. In order to amplify the success probability, one could, naturally, draw several random samples and check them. This will result in a "zero-error" $\mathsf{ZPP}^{\mathsf{NP}}$ algorithm. The NP oracle, however, appears to be necessary, since it is unclear how to *efficiently* verify or select a correct solution, even when one is presented. This raises the following natural question:

> *"Can we amplify the success probability of solving* Avoid *without an* NP *oracle, or at least using an oracle for an easier problem"?*

In the *Minimum Circuit Size Problem* (MCSP), we are given the truth table $tt \in \{0,1\}^{2^n}$ of an n-variate Boolean function $f : \{0,1\}^n \to \{0,1\}$ and a parameter $0 \le s \le 2^n$, and the goal is

*Department of Computer Science, Ariel University, Israel. This research was conducted with the support of the State of Israel, the Ministry of Immigrant Absorption, and the Center for the Absorption of Scientists. Email: `edwardh@ariel.ac.il`

†Computer Science Department, Boston College, Chestnut Hill, MA. Email: `ilya.volkovich@bc.edu`

[1] There is a stronger version corresponding to the dual version of the classical pigeonhole problem where the domain is just one element smaller than the image. This version has also been studied in the literature, yet its complexity properties are quite different.

ISSN 1433-8092

to determine whether f can be computed by a Boolean circuit of size at most s. The problem shares a similar fate to Avoid: although its origins trace back to the 1960s (see e.g. [Tra84]), the interest in the problem was renewed following its reintroduction by Kabanets and Cai in [KC00]. In that work, it was also observed that $\mathsf{MCSP} \in \mathsf{NP}$. However, the true complexity of the problem remains unknown. In particular, it is neither known nor believed to be NP-hard under standard (deterministic) many-to-one reductions [MW15, HP15]. Nevertheless, several recent results provide evidence of NP-hardness of MCSP under more general notions of reduction [ILO20, Ila20, Ila23]. More precisely, [ILO20, Ila20] establish NP-hardness of *variants* of MCSP (such as "multi-output" and partial) under *randomized* many-to-one reductions while the recent work [Ila23] shows that MCSP is NP-hard in the *Randomized Oracle Model* and that MCSP^O — the relativized version of MCSP — is NP-hard under P/poly reductions for a random oracle O. Finally, MCSP and, in fact, MCSP^B can be viewed as special cases of *Natural Properties* [RR97, KC00] for any oracle B.

The purpose of this paper is to connect Avoid to MCSP in a natural setting: under randomized Turing reductions. Recently Ghentiyala, Li, and Stephens-Davidowitch [GLSD25] connected it to Arthur–Merlin protocols and showed that $\mathsf{BPP}^{\mathsf{Avoid}} \subseteq \mathsf{AM} \cap \mathsf{coAM}$ (a corresponding statement holds for the promise versions of these classes). They also noted that for a modestly large stretch $m = n+\omega(\log n)$ the statement becomes trivial since this oracle Avoid_m does not bring more power to BPP, namely, $\mathsf{BPP}^{\mathsf{Avoid}_m} = \mathsf{BPP}$. We ask what is the complexity of Avoid for other values of m, specifically for the hardest instance of the problem when $m = n + 1$? We show that even in this case one can replace the Avoid oracle with an oracle for MCSP.

While, at face value, $\mathsf{BPP}^{\mathsf{MCSP}}$ and $\mathsf{AM} \cap \mathsf{coAM}$ appear incomparable given our current state of knowledge, an observation of Hirahara and Watanabe on *oracle-independent* reductions to MCSP [HW16] suggests that $\mathsf{BPP}^{\mathsf{MCSP}}$ may, "after all", be contained in $\mathsf{AM} \cap \mathsf{coAM}$. For a further discussion on this matter see Section 4.

1.1 Results

Our main result shows how to amplify the success probability of solving Avoid given oracle access to MCSP.

Theorem 1. *There exists a randomized algorithm that given $\varepsilon > 0$, a Boolean circuit $C : \{0,1\}^n \to \{0,1\}^m$ (for $m > n$), and oracle access to* MCSP*, outputs a string $y \in \{0,1\}^m$ such that $y \notin \mathrm{Im}(C)$, with probability at least $1 - \varepsilon$, in time polynomial in $1/\varepsilon$ and the size of C.*

As an immediate corollary, we obtain an upper bound on the set of all languages (and promise problems) that reduce to Avoid via randomized Turing reductions.

Corollary 2. $(\mathsf{pr})\mathsf{BPP}^{\mathsf{Avoid}} \subseteq (\mathsf{pr})\mathsf{BPP}^{\mathsf{MCSP}}$.

2 Preliminaries

We begin by formally defining and recalling the relevant problems.

Definition 2.1 (Avoid)**.** *Given a Boolean circuit $C : \{0,1\}^n \to \{0,1\}^m$ with $m > n$, find an string $y \in \{0,1\}^m$ such that $y \notin \mathrm{Im}(C)$.*

Definition 2.2 (MCSP, MCSP^B)**.** *Given the truth table $tt \in \{0,1\}^{2^n}$ of an n-variate Boolean function $f : \{0,1\}^n \to \{0,1\}$ and a parameter $0 \le s \le 2^n$, decide whether f can be computed by a Boolean circuit of size at most s. In the relativized version of the problem, MCSP^B, the Boolean circuit is also allowed to use B-oracle gates.*

2.1 Inversion of polynomial-time computable functions

Allender et al. [ABK$^+$06] showed that sufficiently dense sets of strings with sufficiently high time-bounded Kolmogorov complexity (in an appropriate sense)[2] can be used to break (i.e., invert) any polynomial-time computable function. They further observed that such sets can be constructed in $\mathsf{P}^{\mathsf{MCSP}}$, and hence an MCSP oracle will suffice for this purpose:

Lemma 2.3 ([ABK$^+$06, Theorem 45 and discussion after that])**.** *Let $f_z(x) = f(z, x)$ be a function computable uniformly in time polynomial in $|x|$. There exists a probabilistic oracle Turing machine $A^\bullet$ and $k \in \mathbb{N}$ such that for any n and z:*

$$\Pr_{|x|=n,\ \tau}\left[f_z\left(A^{\mathsf{MCSP}}(z, f_z(x), \tau)\right) = f_z(x)\right] \geq 1/n^k,$$

where x is chosen uniformly at random and τ denotes the randomness of A; this procedure runs in time polynomial in $|x|$ (and $|z|$ and $|f_z(x)|$, but it does not matter due to the uniformity condition).

By observing that evaluating a given Boolean circuit on an input x can be carried uniformly in time polynomial in the size of the circuit, and using a standard transformation between strong and weak one-way functions we obtain the following:

Lemma 2.4. *There exists a probabilistic oracle Turing machine $M^\bullet$ that given $\varepsilon > 0$ and a Boolean circuit C on n inputs and $m = n + 1$ outputs, runs in time $\mathrm{poly}(|C|, 1/\varepsilon)$ and satisfies:*

$$\Pr_{|x|=n,\ \tau}\left[C\left(M^{\mathsf{MCSP}}(1/\varepsilon, C, C(x), \tau)\right) = C(x)\right] \geq 1 - \varepsilon,$$

where x is chosen uniformly at random and τ denotes the randomness of M.

Although the result of [ABK$^+$06] has been used in this form before (see, e.g., [AGvM$^+$18]), for completeness we formally derive it from its original form (Lemma 2.3) in Section A of the Appendix.

Remark 2.5. *One can further observe that the MCSP oracle in Lemmas 2.3 and 2.4 can be replaced with MCSP^B for any language B and, in fact, with any exponentially-useful "natural property" in the sense of [RR97]. This includes MKTP and other measures. See further discussion in Section 4.*

For notational convenience, we denote the following event of "successful inversion":

Definition 2.6. *$I_{C,y,\tau}$ denotes the event that $C\left(M^{\mathsf{MCSP}}(1/\varepsilon, C, y, \tau)\right) = y$. That is, the event that given oracle access to MCSP and randomness τ, the machine M outputs a pre-image of y under C.*

In particular, note that for any $y \notin \mathrm{Im}(C)$ and any τ we have that $\Pr_\tau[I_{C,y,\tau}] = 0$. Furthermore, given the above notation, we can succinctly rephrase Lemma 2.4 as

$$\Pr_{|x|=n,\ \tau}\left[I_{C,C(x),\tau}\right] \geq 1 - \varepsilon.$$

Fix a Boolean circuit $C : \{0,1\}^n \to \{0,1\}^m$. The following lemma relates the probability of inverting a random *input* of C with the probability of inverting a random *element* in the co-domain of C, i.e., $\{0,1\}^m$. We include this (very standard) proof for the sake of self-containment:

[2] Namely, one needs a language L containing at least $2^n/n^k$ bit strings of each length n such that for every $x \in L, \mathsf{KT}(x) \geq |x|^{1/k}$, where KT denotes their version of time-bounded Kolmogorov complexity.

Lemma 2.7. $\Pr_{|y|=m,\tau}\left[I_{C,y,\tau}\right] \leq \frac{1}{2^{m-n}} \cdot \Pr_{|x|=n,\tau}\left[I_{C,C(x),\tau}\right].$

Proof.

$$\Pr_{x,\tau}\left[I_{C,C(x),\tau}\right] = \sum_{y\in\{0,1\}^m} \Pr_{\tau}\left[I_{C,y,\tau}\right] \cdot \Pr_{x}\left[C(x)=y\right] = \sum_{y\in\mathrm{Im}(C)} \Pr_{\tau}\left[I_{C,y,\tau}\right] \cdot \Pr_{x}\left[C(x)=y\right] \geq$$

$$\sum_{y\in\mathrm{Im}(C)} \Pr_{\tau}\left[I_{C,y,\tau}\right] \cdot \frac{1}{2^n} = \frac{2^m}{2^n} \cdot \frac{1}{2^m} \cdot \sum_{y\in\{0,1\}^m} \Pr_{\tau}\left[I_{C,y,\tau}\right] = \frac{2^m}{2^n} \cdot \Pr_{y,\tau}\left[I_{C,y,\tau}\right]. \qquad \square$$

Our next lemma shows that if the inverter from Lemma 2.4 fails to produce a pre-image for a random element y then with high probability this element y has no such pre-image. That is, y lies outside the range of C.

Lemma 2.8. $\Pr_{|y|=m,\tau}[y \notin \mathrm{Im}\,(C) \mid \bar{I}_{C,y,\tau}] \geq \Pr_{|x|=n,\tau}\left[I_{C,C(x),\tau}\right].$

Proof.

$$\Pr_{y,\tau}[y \notin \mathrm{Im}\,(C) \mid \bar{I}_{C,y,\tau}] = \frac{\Pr_{y,\tau}[y \notin \mathrm{Im}\,(C) \wedge \bar{I}_{C,y,\tau}]}{\Pr_{y,\tau}[\bar{I}_{C,y,\tau}]} = \frac{\Pr_{y,\tau}[y \notin \mathrm{Im}\,(C)]}{\Pr_{y,\tau}[\bar{I}_{C,y,\tau}]} \geq \frac{1-\frac{1}{2^{m-n}}}{1-\frac{1}{2^{m-n}}\cdot \Pr_{|x|=n,\tau}\left[I_{C,C(x),\tau}\right]} =$$

$$\frac{2^{m-n}-1}{2^{m-n}-\Pr_{|x|=n,\tau}\left[I_{C,C(x),\tau}\right]} \geq \Pr_{|x|=n,\tau}\left[I_{C,C(x),\tau}\right]. \qquad \square$$

3 Reducing Avoid to MCSP: A Proof of the Main Result

We are now ready to prove our main result.

Proof of Theorem 1. Let $M^{\bullet}$ be the machine from Lemma 2.4. The following procedure solves Avoid on input $C : \{0,1\}^n \to \{0,1\}^m$ with high probability:

1. Repeat the following t times:
   (a) Pick $y \in \{0,1\}^m$ uniformly at random.
   (b) If $C\left(M^{\mathsf{MCSP}}(1/\varepsilon, C, y, \tau)\right) \neq y$ then Output y; otherwise continue.
2. Output $\perp$.

This procedure has two sources of error:

1. The procedure outputs an element of $\mathrm{Im}\,(C)$. By Lemmas 2.4 and 2.8 this probability is at most εt.
2. The procedure outputs $\perp$. It happens only if all the random picks fall inside $\mathrm{Im}\,(C)$ (and all the executions of M^{MCSP} succeed, but it does not really matter). This probability is at most $(2^{n-m})^t \leq 1/2^t$ even when $m = n+1$.

The total error is therefore at most $\varepsilon t + 2^{-t}$, which can be made less than any $\varepsilon' > 0$ by choosing $t = n$ and $\varepsilon = \varepsilon'/(2n)$. $\square$

Proof of Corollary 2. Assume that the $\mathsf{BPP}^{\mathsf{Avoid}}$ machine runs in time $p(n)$ for some polynomial $p(n)$. By choosing $\varepsilon = \frac{1}{8p(n)}$ in Theorem 1, all queries will be answered correctly with probability at least $\frac{7}{8}$, and the error of the machine will remain bounded. $\square$

4 Oracle-independent reductions and comparison to previous results

We remark that in the procedure solving Avoid described above, the machine M operates correctly not only when given oracle access to MCSP itself, but also when MCSP is replaced by MCSP^B for any *arbitrary* oracle B. This notion of *oracle-independent* reductions to MCSP was introduced by Hirahara and Watanabe [HW16], who observed that most known reductions to MCSP (including those of [ABK+06]) operate in this manner (see [HW16, Section 3]). In that work, they also studied the power of such reductions. In particular, they proved the following:

Theorem ([HW16, Theorem 2])**.** *If a language L is reducible to MCSP via an oracle-independent randomized reduction with negligible error that makes at most one query, then $L \in \mathsf{AM} \cap \mathsf{coAM}$. In other words,*

$$\bigcap_B \mathsf{BPP}^{\mathsf{MCSP}^B[1]} \subseteq \mathsf{AM} \cap \mathsf{coAM}.$$

They also showed that deterministic, oracle-independent, polynomial-time Turing reductions do not provide additional power, *regardless* of the number of queries:

Theorem ([HW16, Theorem 1])**.** $\bigcap_B \mathsf{P}^{\mathsf{MCSP}^B} = \mathsf{P}$.

As our procedure for solving Avoid is oracle-independent (see also Remark 2.5) we obtain the following "oracle-independent" extension to Corollary 2:

Corollary 3. $\mathsf{BPP}^{\mathsf{Avoid}} \subseteq \bigcap_B \mathsf{BPP}^{\mathsf{MCSP}^B}$.

Unfortunately, the above results/characterizations by [HW16] do not apply to our procedure since our reduction is randomized and makes multiple queries. Meanwhile, a recent result of Ghentiyala, Li, and Stephens-Davidowitz [GLSD25] shows that $\mathsf{BPP}^{\mathsf{Avoid}} \subseteq \mathsf{AM} \cap \mathsf{coAM}$. Taken together, these results provide supporting evidence for the following conjecture, which we put forward:

Conjecture 4.1. $\bigcap_B \mathsf{BPP}^{\mathsf{MCSP}^B} \subseteq \mathsf{AM} \cap \mathsf{coAM}$.

That is, if a language L is reducible to MCSP via an oracle-independent randomized reduction then $L \in \mathsf{AM} \cap \mathsf{coAM}$, regardless of the number of queries.

At the same time, Sdroievski, da Silva, and Vignatti [SdSV19] devised a randomized algorithm that, given oracle access to MCSP, solves the Hidden Subgroup Problem (HSP). We note that HSP is a central problem in the theory of quantum computing. Incidentally, their reduction is also oracle-independent[3]. Thus, if true, the conjecture would not only imply that our result provides an alternative (and potentially stronger) avenue for placing $\mathsf{BPP}^{\mathsf{Avoid}}$ in $\mathsf{AM} \cap \mathsf{coAM}$, but would also place $\mathsf{BPP}^{\mathsf{HSP}}$ in $\mathsf{AM} \cap \mathsf{coAM}$, which, to the best of our knowledge, is currently unknown.

[3] Although the formal statement is that HSP can be solved in $\mathsf{BPP}^{\mathsf{MTKP}}$, their reduction is actually based on Theorem 45 of [ABK+06] (Lemma 2.4). As in our case, this results in an oracle-independent containment in $\mathsf{BPP}^{\mathsf{MCSP}}$.

An additional piece of evidence supporting the conjecture (as observed in [HW16]) was provided by the result of Allender and Das [AD14], which shows that $\mathsf{SZK} \subseteq \mathsf{BPP}^{\mathsf{MCSP}}$ via an oracle-independent reduction. This complements the previously known containment $\mathsf{SZK} \subseteq \mathsf{AM} \cap \mathsf{coAM}$ [For89, Oka00, SV03]. Could the techniques of [GLSD25] be extended to prove the conjecture?

References

[ABK$^+$06] E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006.

[AD14] E. Allender and B. Das. Zero knowledge and circuit minimization. In *Mathematical Foundations of Computer Scienc MFCS*, pages 25–32, 2014.

[AGvM$^+$18] E. Allender, J. A. Grochow, D. van Melkebeek, C. Moore, and A. Morgan. Minimum circuit size, graph isomorphism, and related problems. *SIAM J. Comput.*, 47(4):1339–1372, 2018.

[For89] L. Fortnow. The complexity of perfect zero-knowledge. *Advances in Computing Research*, 5:327–343, 1989.

[GLSD25] S. Ghentiyala, Z. Li, and N. Stephens-Davidowitz. Range avoidance, Arthur-Merlin, and TFNP. *Electron. Colloquium Comput. Complex.*, TR25-210, 2025.

[HP15] J. M. Hitchcock and A. Pavan. On the NP-completeness of the minimum circuit size problem. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS*, pages 236–245, 2015.

[HW16] S. Hirahara and O. Watanabe. Limits of minimum circuit size problem as oracle. In *31st Conference on Computational Complexity, CCC*, pages 18:1–18:20, 2016.

[Ila20] R. Ilango. Approaching MCSP from above and below: Hardness for a conditional variant and AC0[p]. In *11th Innovations in Theoretical Computer Science Conference, ITCS*, volume 151 of *LIPIcs*, pages 34:1–34:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[Ila23] R. Ilango. SAT reduces to the minimum circuit size problem with a random oracle. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 733–742. IEEE, 2023.

[ILO20] R. Ilango, B. Loff, and I. Carboni Oliveira. NP-hardness of circuit minimization for multi-output functions. In *35th Conference on Computational Complexity, CCC*, volume 169 of *LIPIcs*, pages 22:1–22:36. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[KC00] V. Kabanets and J. Cai. Circuit minimization problem. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.

[KKMP21] R. Kleinberg, O. Korten, D. Mitropolsky, and C. Papadimitriou. Total Functions in the Polynomial Hierarchy. In James R. Lee, editor, *12th Innovations in Theoretical*

Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 44:1–44:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[Kra25] Jan Krajíček. *Proof Complexity Generators*. London Mathematical Society Lecture Note Series. Cambridge University Press, 2025.

[MW15] C. D. Murray and R. R. Williams. On the (non) NP-hardness of computing circuit complexity. In *30th Conference on Computational Complexity, CCC*, pages 365–380, 2015.

[Oka00] T. Okamoto. On relationships between statistical zero-knowledge proofs. *J. Comput. Syst. Sci.*, 60(1):47–108, 2000.

[PWW88] J. Paris, A. Wilkie, and A. R. Woods. Provability of the pigeonhole principle and the existence of in?nitely many primes. *J. Symb. Log.*, (53):1235–1244, 1988.

[RR97] A. A. Razborov and S. Rudich. Natural proofs. *J. of Computer and System Sciences*, 55(1):24–35, 1997.

[SdSV19] N. M. Sdroievski, M. V. G. da Silva, and A. L. Vignatti. The hidden subgroup problem and MKTP. *Theor. Comput. Sci.*, 795:204–212, 2019.

[SV03] A. Sahai and S. P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003.

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

[Yao82] A. Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.

A Proof sketch of Lemma 2.4

Proof Sketch of Lemma 2.4. We need essentially to apply a standard construction of converting a weak one-way function into a strong one-way function and to take care of the "uniform" setting.

In terms of Lemma 2.3 we will define the following function $f_z(x)$:

$$f_C(x) = (C(x_1), C(x_2), \dots, C(x_t)),$$

for $x_i \in \{0,1\}^n$ and for large enough $t = O(\frac{n}{\varepsilon})$. Then similarly to [Yao82] the following algorithm M^{MCSP} succeeds with probability $1 - 1/\varepsilon$: it repeats an even larger polynomial number of times the following: it takes $j \in \{1, \dots, t\}$ at random, takes all x_i's at random, employs A^{MCSP} on $(x_1, \dots, x_t)$ with x_j replaced by y, and outputs the result if it is correct.

Now to make our function uniform in C (that is, make its time polynomial in $|x|$ only), assume that C has at least $\sqrt{|C|}$ inputs. If it has fewer inputs, add enough fake inputs (and outputs) to it and consider a new circuit D' that computes the identity function on them. It won't change the probability of inverting this circuit (as ε) is a constant, and the running time of $M^\bullet$ will become polynomial in $|C|$ (rather than just in n). □

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