Electronic Colloquium on Computational Complexity, Revision 5 of Report No. 49 (2025)

Range Avoidance and Remote Point: New Algorithms and Hardness

Shengtang Huang *　　Xin Li †　　Yan Zhong ‡

Abstract

The Range Avoidance (Avoid) problem $\mathscr{C}$-Avoid$[n, m(n)]$ asks that, given a circuit in a class $\mathscr{C}$ with input length n and output length $m(n) > n$, find a string not in the range of the circuit. This problem has been a central piece in several recent frameworks for proving circuit lower bounds and constructing explicit combinatorial objects. Previous work by Korten (FOCS' 21) and by Ren, Santhanam, and Wang (FOCS' 22) showed that algorithms for Avoid are closely related to circuit lower bounds. In particular, Korten's work reinterpreted an earlier result from bounded arithmetic, originally proved by Jeřábek (Ann. Pure Appl. Log. 2004), as an equivalence in computational complexity between the existence of $\mathbf{FP}^{\mathbf{NP}}$ algorithms for the general Avoid problem and $2^{\Omega(n)}$ lower bounds against general Boolean circuits for the class $\mathbf{E}^{\mathbf{NP}}$. In this work, we significantly complement these works by generalizing the equivalence result to restricted circuit classes and obtain the following:

- For any constant depth unbounded fan-in circuit class $\mathscr{C} \supseteq \mathsf{AC}^0$, there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-Avoid$[n, n^{1+\varepsilon}]$ (for any constant $\varepsilon > 0$) if and only if $\mathbf{E}^{\mathbf{NP}}$ cannot be computed by $\mathscr{C}$ circuits of size $2^{o(n)}$. This addresses an open problem by Korten (Bulletin of EATCS' 25).
- If $\mathbf{E}^{\mathbf{NP}}$ cannot be computed by $o(2^n/n)$ size formulas, then there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-Avoid$[n, 2n]$. Note that by an extension of Ren, Santhanam, and Wang (FOCS' 22), an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0_4-Avoid$[n, n+n^\delta]$ for any constant $\delta \in (0,1)$ implies $\mathbf{E}^{\mathbf{NP}}$ cannot be computed by $o(2^n/n)$ size formulas.

These results yield the first characterizations of $\mathbf{FP}^{\mathbf{NP}}$ $\mathscr{C}$-Avoid algorithms for low-complexity circuit classes such as AC^0.

We also consider the average-case analog of Avoid, the Remote Point (Remote-Point) problem, and establish:

- For some suitable function $c(n)$ and constant $\gamma > 0$, there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point$[n, n^{6+\gamma}, c(O_\gamma(\log n))]$ if and only if $\mathbf{E}^{\mathbf{NP}}$ cannot be $(1/2-c(n))$-approximated by circuits of size $2^{o(n)}$.

Finally, we also present two improved algorithms for NC^0-Avoid:

- A family of $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time algorithms for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$ for any $\varepsilon > 0$, exhibiting the first subexponential-time algorithm for any super-linear stretch.
- Faster local algorithms for NC^0_k-Avoid$[n, n+1]$ running in time $O(n2^{\frac{k-2}{k-1}n})$, improving the naive $2^n \cdot \mathrm{poly}(n)$ bound.

*School of the Gifted Young, University of Science and Technology of China, peanuttang@mail.ustc.edu.cn. Work done while visiting Johns Hopkins University.

†Department of Computer Science, Johns Hopkins University, lixints@cs.jhu.edu. Supported by NSF CAREER Award CCF-1845349 and NSF Award CCF-2127575.

‡Department of Computer Science, Johns Hopkins University, yzhong36@jhu.edu. Supported by NSF CAREER Award CCF-1845349.

ISSN 1433-8092

Contents

1 Introduction

The *Range Avoidance* problem (AVOID for short) is a total search problem introduced in [KKMP21, Kor21, RSW22], which has recently garnered significant attention. This interest stems from several natural motivations, such as identifying natural total search problems in the polynomial hierarchy (more specifically $\mathbf{\Sigma}_2$) and compelling applications in proof complexity. Notably, Korten [Kor21] demonstrated that numerous explicit constructions of important combinatorial objects can be reduced to instances of AVOID. These include optimal Ramsey graphs, expander graphs, rigid matrices, and hard functions, among others.

At its core, the Range Avoidance problem captures a broad class of objects whose existence is typically proven via the probabilistic method [Erd47]. As such, solving AVOID offers a potentially unified way for constructing these objects explicitly. We now define the problem formally.

Definition 1.1 (AVOID)**.** *The* range avoidance problem*, denoted by* AVOID*, is the total search problem in which, given a Boolean circuit* $C : \{0,1\}^n \to \{0,1\}^m$ *for* $m := m(n)$[1]$> n$*, output any* $y \in \{0,1\}^m \setminus \mathrm{Range}(C)$*, where* $\mathrm{Range}(C) := \{C(x) \mid x \in \{0,1\}^n\}$*.*

Closely related is the more general REMOTE-POINT[2] problem, which is studied extensively in previous works [KKMP21, CHLR23, CL24] and can be thought as the "average-case analog" of AVOID.

Definition 1.2 (REMOTE-POINT)**.** *Given a code where the encoding function is represented by a circuit* $C : \{0,1\}^n \to \{0,1\}^m$ *for* $m := m(n) > n$ *and the codewords are the range of the circuit, find an* m*-bit string that is far from all codewords in Hamming distance.*

While the original formulation of AVOID allows arbitrary circuits, subsequent work initiated by [RSW22] has focused on the problem for restricted circuit classes.

Definition 1.3. *Let* $\mathscr{C}$ *be a (multi-output) circuit class,*

- $\mathscr{C}$**-AVOID**$[n, m]$ *is the class of* AVOID *problems where the circuits are in* $\mathscr{C}$*, with input length* n *and output length* m*;*
- $\mathscr{C}$**-REMOTE-POINT**$[n, m, c(n)]$ *denotes the class of* REMOTE-POINT *problems where the underlying circuits belong to* $\mathscr{C}$*, with input length* n *and output length* m*, and where the desired output has relative Hamming distance* $1/2 - c(n)$ *from every string in the range of circuits in* $\mathscr{C}$*.*

A prominent motivation for studying $\mathscr{C}$-AVOID is its implication for circuit lower bounds. In particular, [RSW22] showed that for any circuit class $\mathscr{C}$ satisfying the *universality property* — namely, the *truth table generator* $\mathsf{TT}_{\mathscr{C}}$ (i.e., a circuit that, given an encoding of a circuit $C \in \mathscr{C}$, outputs C's truth table) is itself computable by $\mathscr{C}$ circuits (e.g., $\mathsf{AC}^0, \mathsf{TC}^0, \mathsf{NC}^1$) — efficient algorithms for $\mathscr{C}$-AVOID imply circuit lower bounds for $\mathscr{C}$. Specifically, solving $\mathscr{C}$-AVOID in $\mathbf{FP}$ (resp. $\mathbf{FP}^{\mathbf{NP}}$) implies that $\mathbf{E}$ (resp. $\mathbf{E}^{\mathbf{NP}}$) does not have $\mathscr{C}$ circuits.[3] Analogously, $\mathbf{FP}$ (resp. $\mathbf{FP}^{\mathbf{NP}}$) algorithms for $\mathscr{C}$-REMOTE-POINT imply average-case $\mathscr{C}$ circuit lower bounds, which are central questions in the area of average-case complexity that have resulted in a large body of works improving correlation bounds for various models of computation (e.g., [Che24, CR22, CLW20, CL23, LZ24]). On

[1]The function $m(n)$ is called the *stretch* of the circuit.

[2]We sometimes use RPP as a shorthand for REMOTE-POINT.

[3]The size of the circuit lower bound depends on the stretch of the AVOID instance.

the other hand, these results also imply that it is potentially hard to design efficient algorithms for $\mathscr{C}$-AVOID even when $\mathscr{C}$ is restricted, hence many algorithms given in previous work are *conditional*.

Furthermore, these works also demonstrate that AVOID is already extremely interesting and useful for restricted classes of circuits, for example, even when the circuit is in the class NC^0, and even when each output bit only depends on at most 4 input bits. Below, we use NC^0_k to stand for circuits in NC^0 where each output bit depends on at most k input bits. The same notation goes for the class NC^1. In this sense, the work of [RSW22] shows that, suppose for every constant $\varepsilon > 0$, there is an $\mathbf{FP}$ (resp. $\mathbf{FP^{NP}}$) algorithm for NC^0_4-AVOID$[n, n + n^\varepsilon]$, then for every $k \geq 1$, there is an $\mathbf{FP}$ (resp. $\mathbf{FP^{NP}}$) algorithm for NC^1_k-AVOID; and for every $\gamma > 0$, there is a family of functions in $\mathbf{E}$ (resp. $\mathbf{E^{NP}}$) that cannot be computed by Boolean circuits of depth $n^{1-\gamma}$. Furthermore, [GLW22] showed that constructing binary linear codes achieving the Gilbert-Varshamov bound or list-decoding capacity, and constructing rigid matrices reduce to NC^0_4-AVOID; and [GGNS23] showed that constructing rigid matrices reduces even to NC^0_3-AVOID.

Driven by these motivations and applications, there have been several works studying both algorithms and hardness results for AVOID and REMOTE-POINT. On the algorithm side, [CHLR23] designed an unconditional $\mathbf{FP^{NP}}$ algorithm for ACC^0-REMOTE-POINT$[n, \mathrm{qpoly}(n), 1/\mathrm{poly}(n)]$ ($\mathrm{qpoly}(n)$ denotes quasi-polynomial(n)), recovering the state-of-the-art average-case lower bound for ACC^0 against $\mathbf{E^{NP}}$. A recent breakthrough [CHR24, Li24] showed that $\mathsf{S_2E} \not\subset$ *i.o.*-$\mathsf{SIZE}[2^n/n]$[4] via a single-valued $\mathsf{FS_2P}$ algorithm to AVOID, improving over the decades' old lower bound that $\Delta_3\mathsf{E} = \mathsf{E}^{\Sigma_2} \not\subset \mathsf{SIZE}[2^{o(n)}]$ [MVW99]. On the hardness side, Ilango, Li, and Williams [ILW23] showed that under the assumption that subexponential secure indistinguishability obfuscation ($i\mathcal{O}$) exists [JLS21] and $\mathbf{NP} \neq \mathbf{coNP}$, we have that AVOID $\notin \mathbf{FP}$ (i.e., there are no polynomial-time algorithms to solve AVOID). A subsequent work by Chen and Li [CL24] generalizes the framework and shows that under plausible cryptographic assumptions, $\mathscr{C}$-AVOID and $\mathscr{C}$-REMOTE-POINT are not in $\mathbf{FP}$, or even not in $\mathbf{SearchNP}$, when the underlying $\mathscr{C}$ has small enough stretch (e.g., in the case of NC^0-AVOID, the hardness works for the minimal stretch $m(n) = n + 1$).

However, for certain applications (e.g., explicit constructions of important combinatorial objects) one would desire *relatively efficient* algorithms (e.g., polynomial-time algorithms or at least $\mathbf{FP^{NP}}$ algorithms). Yet even for the case of NC^0-AVOID, the current state-of-the-art results only work for large stretches. For example, the polynomial-time algorithms for NC^0_k-AVOID [GLW22, GGNS23] require the stretch to be at least $n^{k-1}/\log(n)$. Most recently, this was improved to $\widetilde{O}(n^{k/2})$ for even k by [KPI25], which also improved the stretch to $(\widetilde{O}(n^{k/2+(k-2)/(2k+4)}))$ with an $\mathbf{FP^{NP}}$ algorithm for odd k. A conditional $\mathbf{FP^{NP}}$ algorithm was proposed in [RSW22] for NC^0-AVOID with stretch $n^{1+\varepsilon}$ for any constant ε, and whether there is an unconditional $\mathbf{FP^{NP}}$ algorithm for such stretch is left as a central open question in [RSW22]. Even if one allows for subexponential ($2^{O(n^{1-\varepsilon})}$) time, the best known algorithms for NC^0_k-AVOID only works for stretch $n^{k-2+\varepsilon}$ [GGNS23].

A recent work by Kuntewar and Sarma [KS25] showed that the monotone version of NC^0_3-AVOID$[n, n+1]$, i.e., MONOTONE-NC^0_3-AVOID$[n, n+1]$ can be solved in polynomial time; the symmetric version of NC^0_3-AVOID$[n, 8n+1]$, i.e., SYMMETRIC-NC^0_3-AVOID$[n, n+1]$ can be solved in polynomial time.

These results fall short of the above mentioned goal of a unified approach towards explicit constructions of combinatorial objects, as most interesting explicit construction problems only reduce to $\mathscr{C}$-AVOID with very small *stretch*. For example, in the case of NC^0-AVOID, to show a better circuit lower bound, one needs $m = n + n^{o(1)}$; while finding rigid matrices enough for Valiant's application needs $m = n + n^{2/3}$ [GGNS23]. This was also noted and remarked in [RSW22].

[4]The prefix "*i.o.*-" indicates that $\mathsf{S_2E}$ is not infinitely often in $\mathsf{SIZE}[2^n/n]$, that is $\mathsf{S_2E}$ *eventually* requires $\mathsf{SIZE}[2^n/n]$ circuit.

"We think this result reveals some fundamental difference between the small-stretch regime ($m(n) = n+1$), for which an avoidance algorithm for NC^0 implies breakthrough lower bounds, and the large-stretch regime ($m(n) = n^{1+\Omega(1)}$), for which an avoidance algorithm for NC^0 seems within reach (Theorem 3.12)."

Therefore, it is interesting and important to study the tradeoff between the stretch and the hardness for $\mathscr{C}$-Avoid when $\mathscr{C}$ is restricted (e.g., NC^0, AC^0 and ACC^0), and similarly for $\mathscr{C}$-Remote-Point as better algorithms in this case may lead to stronger average-case circuit lower bounds. In this paper, we make progress towards this direction, by establishing several new results in terms of both algorithms and hardness for $\mathscr{C}$-Avoid and $\mathscr{C}$-Remote-Point, where $\mathscr{C}$ are suitable classes of circuits.

1.1 Our Results

While as mentioned before, several previous works showed that algorithms for $\mathscr{C}$-Avoid or $\mathscr{C}$-Remote-Point with small stretch lead to circuit lower bounds, the works [Jeř04, Kor21, CHR24] remarkably showed that the converse is also true in the case where $\mathscr{C}$ is the class of unrestricted Boolean circuits. Specifically, they showed that

$$\text{Avoid} \in \mathbf{FP^{NP}} \iff \mathbf{E^{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^{o(n)}] \iff \mathbf{E^{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^n/n]^{5}$$

In particular, assuming $\mathbf{E^{NP}}$ does not have subexponential-size circuits implies an $\mathbf{FP^{NP}}$ algorithm for Avoid on unrestricted circuits. This assumption is significantly weaker than the classical hardness required in PRG-based approaches [IW97, KvM02], which assume that $\mathbf{E}$ lacks subexponential-size SAT-oracle circuits to derandomize $\mathbf{FZPP^{NP}}$.

Thus, for unrestricted Boolean circuits, algorithms for Avoid and lower bounds for $\mathbf{E^{NP}}$ are, in a precise sense, equivalent. However, such an equivalence was previously unknown for restricted circuit classes. Our first major contribution is to significantly extend previous works, by establishing (near) equivalence for certain restricted classes $\mathscr{C}$, more specifically constant depth circuits with possible augmented gates[6]. As a result, we also obtain conditional $\mathbf{FP^{NP}}$ algorithms for $\mathscr{C}$-Avoid for these circuit classes $\mathscr{C}$ with suitable smaller stretch, under much weaker assumptions than those needed for general Avoid in [Kor21]. In addition, we establish a new equivalence result between $\mathbf{FP^{NP}}$ algorithms for Remote-Point and average-case general circuit lower bound for $\mathbf{E^{NP}}$.

1.1.1 Equivalence between $\mathbf{FP^{NP}}$ $\mathscr{C}$-Avoid Algorithms and Exponential-size $\mathscr{C}$ Circuit Lower Bound against $\mathbf{E^{NP}}$

As mentioned in the above paragraphs, previous works [Kor21, RSW22] established the direction from Avoid algorithms to circuit lower bounds. In this work, we complete the equivalence by showing the converse direction for a range of natural restricted circuit classes.

Results for NC^0_4 Circuits with Small Stretch. Our first set of results concerns NC^0_4 circuits. We show that near-maximal formula lower bounds against $\mathbf{E^{NP}}$ imply efficient algorithms for NC^0_4-Avoid with small stretch:

[5] The original second equivalence obtained by [Kor21] is $\mathbf{E^{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^{o(n)}] \iff \mathbf{E^{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^n/(2n)]$, which can be strengthened by a finer encoding arguments of circuits [CHR24].

[6] Say, exact threshold gates.

Theorem 1.4. *If $\mathbf{E}^{\mathbf{NP}}$ requires near-maximum ($\Omega(2^n/n)$) size formulas*[7]*, then there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-*Avoid*$[n, 2n]$. In particular, this implies an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0_4-*Avoid*$[n, 2n]$.*

Conversely, extending ideas from [RSW22] (with the proof deferred to Appendix C), we show:

Theorem 1.5 (Strong Version of Theorem 5.8 in [RSW22])**.** *For any constant $\delta \in (0, 1)$, NC^0_4-*Avoid*$[n, n + n^\delta] \in \mathbf{FP}^{\mathbf{NP}} \implies \mathbf{E}^{\mathbf{NP}} \not\subset$ i.o.-$\mathsf{Formula}[o(2^n/n)]$.*

Together, these results nearly characterize the hardness of proving near-maximum $\mathbf{E}^{\mathbf{NP}}$ lower bounds against formulas in terms of $\mathbf{FP}^{\mathbf{NP}}$ algorithms for NC^0_4-Avoid.

Results for Constant Depth Circuit Classes Containing AC^0 with Polynomial Stretch. In the regime of polynomial stretch, we obtain tight equivalences for constant depth unbounded fan-in circuit classes $\mathscr{C}$ satisfying $\mathsf{AC}^0 \subseteq \mathscr{C}$:

Theorem 1.6. *For any constant depth unbounded fan-in circuit class $\mathscr{C}$ such that $\mathsf{AC}^0 \subseteq \mathscr{C}$ (e.g., $\mathsf{AC}^0, \mathsf{ACC}^0, \mathsf{TC}^0$), $\mathbf{E}^{\mathbf{NP}}$ requires $2^{\Omega(n)}$ size $\mathscr{C}$ circuits if and only if there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-*Avoid*$[n, n^{1+\varepsilon}]$ for any constant $\varepsilon > 0$.*

Moreover, we show analogous equivalences for $\mathbf{FQP}^{\mathbf{NP}}$[8] algorithms and $\mathbf{EXP}^{\mathbf{NP}}$ circuit lower bounds:

Theorem 1.7. *For any constant depth unbounded fan-in circuit class $\mathscr{C}$ such that $\mathsf{AC}^0 \subseteq \mathscr{C}$, $\mathbf{EXP}^{\mathbf{NP}}$ requires $2^{\Omega(n)}$ size $\mathscr{C}$ circuits if and only if there is an $\mathbf{FQP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-*Avoid*$[n, n^{1+\varepsilon}]$ for any constant $\varepsilon > 0$.*

These results represent the first equivalence theorems connecting algorithms for $\mathscr{C}$-Avoid with explicit lower bounds for $\mathbf{E}^{\mathbf{NP}}$ and $\mathbf{EXP}^{\mathbf{NP}}$ in restricted circuit classes.

We remark that the complexity-theoretic assumptions we made for Theorem 1.4 and Theorem 1.6 are consistent with our current knowledge of circuit lower bounds.

Connections to Open Problems. Our results make progress on the following open question:

Open Problem 1.8 (Open problem 2 in [Kor25])**.** *Can we reduce $\mathscr{C}$-*Avoid* to circuit lower bounds for $\mathscr{C}$ for any circuit class $\mathscr{C} \subseteq \mathbf{P}/\mathrm{poly}$?*

Specifically, Theorem 1.6 and Theorem 1.7 address Open Problem 1.8 in the stretch regime $m(n) = n^{1+\varepsilon}$, for any constant $\varepsilon > 0$, and any circuit classes containing AC^0. In addition, Theorem 1.4 and Theorem 1.5 also nearly pin down the hardness of proving $\mathbf{E}^{\mathbf{NP}}$ requires exponential size formulas in terms of NC^0_4-Avoid algorithm: proving such a lower bound should be no harder than proving NC^0_4-Avoid$[n, n + n^\delta] \in \mathbf{FP}^{\mathbf{NP}}$ for any $\delta \in (0, 1)$, but should be no easier than NC^0_4-Avoid$[n, 2n] \in \mathbf{FP}^{\mathbf{NP}}$.

[7] In a preliminary version of this paper (Revision1OfTR25-049), we claim a near equivalence regarding "exponential-size NC^1 circuits". However, exponential-size NC^1 circuits actually do not make sense because if the circuit is in NC and the depth is $O(\log n)$, then the size has to be polynomial. It only makes sense to talk about exponential size AC^i circuits.

[8] $\mathbf{FQP}$ denotes the class of functions computable in *quasi-polynomial time*, i.e., time $T(n) = n^{(\log n)^{O(1)}}$.

1.1.2 Equivalence between $\mathbf{FP}^{\mathbf{NP}}$ RPP Algorithms and Average-case Exponential-size Circuit Lower Bound against $\mathbf{E}^{\mathbf{NP}}$

Recall the definition of *good* function from [RSW22].

Definition 1.9 (Good function [RSW22]). *A function $f : \mathbb{N} \to \mathbb{N}$ is good if there is a Turing machine that, given the input n (in binary), outputs the value $f(n)$ (also in binary), and runs in time at most* $\mathrm{poly}(\log n, \log f(n))$.

The equivalence result for Avoid established in [Kor21] naturally raises the question of whether a similar equivalence holds in the average-case setting. In this paper, we answer this question affirmatively and obtain the following theorems.

Theorem 1.10. *Let $c : \mathbb{N} \to \mathbb{N}$ be a good and monotonically decreasing function which satisfies $c(O(\log n)) \geq 1/n$. Then $\mathbf{E}^{\mathbf{NP}}$ cannot be $(1/2 + c(n))$-approximated by $2^{o(n)}$-size general boolean circuits if and only if there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point$[n, n^{6+\gamma}, c(O_\gamma(\log n))]$ for some constant $\gamma > 0$.*

Theorem 1.11. *Let $c : \mathbb{N} \to \mathbb{N}$ be a good and monotonically decreasing function which satisfies $c(O(\log n)) \geq 1/n$. Then $\mathbf{EXP}^{\mathbf{NP}}$ cannot be $(1/2 + c(n))$-approximated by $2^{o(n)}$-size general boolean circuits if and only if there is an $\mathbf{FQP}^{\mathbf{NP}}$ algorithm for Remote-Point$[n, n^{6+\gamma}, c(O_\gamma(\log n))]$ for some constant $\gamma > 0$.*

1.1.3 New NC^0-Avoid Algorithms

As our second contribution, we design a new $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time algorithm for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$. This gives the first subexponential-time[9] algorithm for NC^0_k-Avoid with any super-linear stretch for any constant k.

Theorem 1.12. *For any $\varepsilon > 0$, there exists a family of $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time algorithms for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$. In addition, the algorithm can output a succinct representation of $\geq 1/2$ fraction of strings outside the range.*

Previously, the best known algorithms with comparable running time were applicable only to stretch $m(n) = \tilde{O}(n^{k/2})$ [KPI25][10], making our result the first to achieve subexponential-time performance with superlinear stretch for all k. Subsequently, the work of [GLW22] further improved the running time to $2^{n^{1-\frac{2\varepsilon}{k-3}+o(1)}}$.

Using a known connection between NC^0-Avoid and local PRGs, we show that faster Avoid algorithms would contradict plausible cryptographic assumptions.

Theorem 1.13. *Suppose Assumption 2.20 is true, there does not exist an algorithm for NC^0_k-Avoid running in time 2^{n^β} for some constant $0 < \beta < 1$ that identifies $\mathsf{negl}(n)$ fraction of strings outside the range.*

We also design an improved algorithm for the regime of minimal stretch $m = n + 1$, improving over brute-force search.

Theorem 1.14. *There exists a family of $O(n \cdot 2^{\frac{(k-2)n}{k-1}})$ time algorithms for NC^0_k-Avoid$[n, n+1]$.*

[9] There are two notions of subexponentiality in literature: $\bigcap_{c<1} 2^{O(n^c)}$ and $\bigcup_{c<1} 2^{O(n^c)}$. Here, we denote by subexponential a function that is contained in $\bigcup_{c<1} 2^{O(n^c)}$.

[10] For the special case $k = 3$, an algorithm with comparable running time was obtained in [GGNS23].

Previous and our algorithmic results are summarized in Table 1. Overall, these results expand the algorithmic landscape for $\mathscr{C}$-Avoid across both small and large stretch regimes, with implications for circuit lower bounds and local PRG security.

Problem	Algorithm	Assumption	Reference
$\textsc{Avoid}[n, n+1]$	$\mathbf{FP}^{\mathbf{NP}}$	$\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^{o(n)}]$	[Kor21]
$\textsc{Avoid}[n, n+1]$	$\mathbf{svFS_2P}$[11]	–	[CHLR23, Li24]
$\mathsf{NC}^0_k\text{-}\textsc{Avoid}[n, n^{1+\varepsilon}]$	$2^{n^{1-\frac{2\varepsilon}{k-3}+o(1)}}$	–	[GLY25]
$\mathsf{NC}^0_k\text{-}\textsc{Avoid}[n, O_k(n^{(k-1)/2}\log n)]$	$\mathbf{FP}$	–	[GLY25]
$\mathsf{NC}^0_{2t}\text{-}\mathrm{RPP}[n, O_t(n^t \log n), O(1)]$	$\mathbf{FP}$	–	[KPI25]
$\mathsf{NC}^0\text{-}\textsc{Avoid}[n, n^{1+\varepsilon}]$	$\mathbf{FP}^{\mathbf{NP}}$	Assumption 2.19	[RSW22]
$\mathsf{ACC}^0\text{-}\mathrm{RPP}[n, \mathrm{qpoly}(n), 1/\mathrm{poly}(n)]$	$\mathbf{FP}^{\mathbf{NP}}$	–	[CHLR23]
$\mathrm{RPP}[n, n^{6+\gamma}, c(O_\gamma(\log n))]$	$\mathbf{FP}^{\mathbf{NP}}$	$\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{Avg}_{c(n)}\text{-}\mathsf{SIZE}[2^{o(n)}]$	Theorem 1.6
$\mathscr{C}\text{-}\textsc{Avoid}[n, n^{1+\varepsilon}]$	$\mathbf{FP}^{\mathbf{NP}}$	$\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathscr{C}\text{-}\mathsf{SIZE}[2^{o(n)}]$	Theorem 1.6
$\mathsf{NC}^0_4\text{-}\textsc{Avoid}[n, 2n]$	$\mathbf{FP}^{\mathbf{NP}}$	$\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{Formula}[o(2^n/n)]$	Theorem 1.4
$\mathsf{NC}^0_k\text{-}\textsc{Avoid}[n, n^{1+\varepsilon}]$	$2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$	–	Theorem 1.12
$\mathsf{NC}^0_k\text{-}\textsc{Avoid}[n, n^{k-1}/\log^{k-2} n]$	$\mathbf{FP}$	Assumption 5.5	Theorem 5.6
$\mathsf{NC}^0_k\text{-}\textsc{Avoid}[n, n+1]$	$O(n2^{\frac{k-2}{k-1}n})$	–	Theorem 1.14

Table 1: Range Avoidance and Remote Point Algorithms. In the 9-th row, we assert $\mathscr{C}$ is a constant depth unbounded fan-in circuit class which contains AC^0.

1.2 Technical Overview

Equivalence between $\mathscr{C}$-Avoid$[n, n^{1+\varepsilon}] \in \mathbf{FP}^{\mathbf{NP}}$ and $\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathscr{C}\text{-}\mathsf{SIZE}[2^{o(n)}]$. For a constant depth unbounded fan-in circuit class $\mathscr{C}$, we establish a tight equivalence between the complexity of solving $\mathscr{C}$-Avoid$[n, n^{1+\varepsilon}]$ in $\mathbf{FP}^{\mathbf{NP}}$ and proving exponential lower bounds for $\mathscr{C}$ circuits against $\mathbf{E}^{\mathbf{NP}}$, generalizing the reduction of Jeřábek and Korten [Jeř04, Kor21], who proved that Avoid $\in \mathbf{FP}^{\mathbf{NP}}$ if and only if $\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^{o(n)}]$[12].

The forward direction — namely, that an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-Avoid implies exponential $\mathscr{C}$ circuit lower bounds against $\mathbf{E}^{\mathbf{NP}}$ — was largely established in [RSW22]. A key component of this argument is the *universality property* of the circuit class $\mathscr{C}$: that the truth table generator $\mathsf{TT}_{\mathscr{C}}$ can itself be computed by a circuit in $\mathscr{C}$. We strengthen and formalize this notion, showing that any

[11]We use $\mathbf{svFS_2P}$ to denote single-valued $\mathbf{FS_2P}$ algorithm.

[12]This reduction, which we refer to as *Jeřábek-Korten reduction*, was originally proved in the framework of bounded arithmetic by Jeřábek [Jeř04], and later translated to the language of computational complexity by Korten [Kor21]. Specifically, as pointed out to us by Erfan Khaniki, [Jeř04, Proposition 3.5] proved that the dual weak pigeonhole principle (dwPHP(PV)) is equivalent to the statement asserting the existence of Boolean functions with exponential circuit complexity in Buss' bounded arithmetic theory S^1_2 which captures polynomial time reasoning. An $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Avoid can be extracted from the dual weak pigeonhole principle (i.e., formalization of the totality of Avoid) in S^1_2 via the Witnessing Theorem from [Kra92].

circuit class $\mathscr{C}$ containing AC^0 satisfies this property. The intuition is that the universal circuit $\mathcal{U}$ acts as a decoder: given an encoding of a circuit C and an input x, it decodes C and evaluates it on x. Since decoding and simple simulation can be implemented in AC^0, this universality follows for all such classes.

The reverse direction, which shows that exponential $\mathscr{C}$ circuit lower bounds for functions in $\mathbf{E}^{\mathbf{NP}}$ imply that $\mathscr{C}$-Avoid $\in \mathbf{FP}^{\mathbf{NP}}$, proceeds by generalizing Korten's construction based on the GGM-tree. We illustrate the approach in the context of AC^0-Avoid$[n, n^{1+\varepsilon}]$, although the framework extends to the broader $\mathscr{C}$-Remote-Point$[n, n^{1+\varepsilon}]$ problem for any $\mathscr{C}$ containing AC^0.

We first briefly recall the $\mathbf{FP}^{\mathbf{NP}}$ reduction from circuit lower bound to Avoid in [Jeř04, Kor21] which we thereafter refer to as *Jeřábek-Korten reduction*. Given an instance of Avoid$[n, 2n]$, which we call C, one constructs a new circuit $\mathsf{GGM}[C]$ by composing C along the nodes of a perfect binary tree of height k (this construction is known as the GGM-tree construction). The resulting circuit has stretch $n \cdot 2^k$, and the output $y \in \text{Range}(\mathsf{GGM}[C])$ can be regarded as encoding the truth table of a function g, whose input is the bits used to select a path in the tree. Importantly, due to redundancy and the tree structure in $\mathsf{GGM}[C]$, this output y can be computed by a relatively small-size circuit at the cost of increasing the depth. Thus, the complexity of the function g — whose truth table is y — can be bounded in terms of the complexity of C and the structure of the GGM-tree.

We generalize this framework in the following three aspects: (1) the fan-out of the tree, denoted by q; (2) the height of the tree, denoted by k; and (3) the circuit C, which we draw from a restricted circuit class $\mathscr{C}$.

Let ℓ denote the stretch of the resulting circuit after composing C through the generalized GGM-tree, which we denote by $\mathsf{GGM}_{\ell,q,k}[C]$ (see Figure 1 for an illustration). It is easy to see that $\ell = n \cdot q^k$. To analyze the complexity of any $y \in \text{Range}(\mathsf{GGM}_{\ell,q,k}[C])$, we associate it with a function $g : \{0,1\}^{\log \ell} \to \{0,1\}$ (corresponding to the structure of the GGM-tree), whose truth table is exactly y.

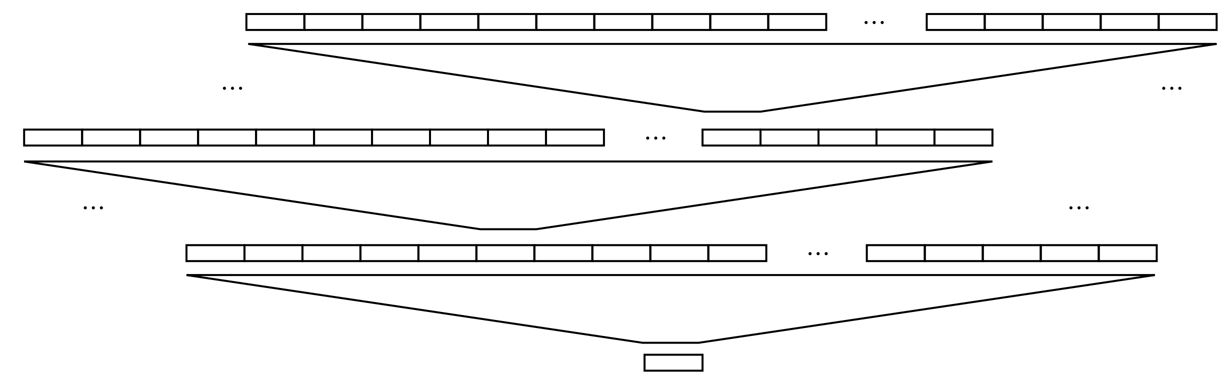

Figure 1: Generalized q-ary GGM-Tree

The circuit computing g can be constructed by composing the circuit C with k layers of multiplexing (selection) and a final indexing operation. These multiplexing and indexing subcircuits can be implemented by $O(n)$-size DNF formulas, and hence belong to any class containing DNF (such as AC^0).

Assuming $C \in \mathsf{AC}^0_d$ where AC^0_d denotes depth d AC^0 circuits, to ensure that $g \in \mathsf{AC}^0$, we must take $k = O(1)$. By setting the fan-out $q = n^{\varepsilon}$, the overall stretch becomes $\ell = n \cdot n^{k\varepsilon} = n^{1+k\varepsilon}$, and the resulting circuit g has size $O(n) + O(|C| \cdot k) = O(n^{1+\varepsilon})$.

Now suppose there exists a function $f \in \mathbf{E}^{\mathbf{NP}}$ that requires AC^0_{dk} circuits of size at least ℓ^{γ} for some constant $\gamma \in (0,1)$. Then for sufficiently large ℓ, f cannot be in the range of $\mathsf{GGM}_{\ell,q,k}[C]$, since all such y have low circuit complexity. Thus, we can use f to find a string not in Range(C) by traversing the GGM-tree with an $\mathbf{NP}$ oracle backwards. This yields an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for AC^0_d-Avoid$[n, nq]$, completing the reduction.

Altogether, this establishes a precise characterization:

$$\mathscr{C}\text{-Avoid}[n, n^{1+\varepsilon}] \in \mathbf{FP}^{\mathbf{NP}} \iff \mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathscr{C}\text{-}\mathsf{SIZE}[2^{o(n)}]$$

for any constant depth unbounded fan-in circuit class $\mathscr{C}$ containing AC^0, and where the stretch satisfies $nq = n^{1+\varepsilon}$ for any arbitrary constant $\varepsilon > 0$.

Equivalence between $\mathbf{RPP}[n, n^{6+\gamma}, c(O_\gamma(\log n))] \in \mathbf{FP}^{\mathbf{NP}}$ and $\mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{Avg}_{c(n)}\text{-}\mathsf{SIZE}[2^{o(n)}]$. We try to extend the GGM-style idea to Remote-Point. Nevertheless, the original Jeřábek-Korten reduction does not work for Remote-Point. Consider the toy case of $\mathsf{GGM}_{4n,2,2}[C]$. Assume that we have an average-case hard truth table y and are not able to find a remote point of C at relative distance ρ by traversing down the tree. Divide y into two blocks y_1, y_2, each of size $2n$. Then there exists $x, x_1, x_2 \in \{0,1\}^n$ such that $C(x_1) \approx_\rho y_1$, $C(x_2) \approx_\rho y_2$, and $C(x) \approx_\rho (x_1 \circ x_2)$, where $C(x_1), C(x_2)$, and $C(x)$ respectively achieve the maximum distance from y_1, y_2, and $x_1 \circ x_2$ among all points in Range(C). However, dividing $C(x)$ into two blocks of equal size x_1' and x_2', it is unclear how close $C(x_1')$ is to $C(x_1)$ and how close $C(x_2')$ is to $C(x_2)$. In other words, it is hard to argue about the distance between y and $\mathsf{GGM}_{4n,2,2}[C](x)$.

To solve this problem, we use an idea from [CHLR23] that reduces Remote-Point to Avoid, and incorporate an error-correcting code at each node of the GGM-tree to prevent the accumulation of errors across levels. To illustrate the core idea, consider first the simpler case of a code with unique decoding. Suppose at each node, we compose the circuit $C : \{0,1\}^n \to \{0,1\}^m$ with a unique decoder $\mathsf{Dec}_{\mathsf{uniq}} : \{0,1\}^m \to \{0,1\}^{qn}$ for a code with decoding radius ρ. If a string $y \in \{0,1\}^{qn}$ is not in Range$(\mathsf{Dec}_{\mathsf{uniq}} \circ C)$, then its encoding $\mathsf{Enc}(y)$ (under the code's natural encoding) is at least ρ-far from Range(C). This property effectively isolates the error at each node: the failure to find a preimage of y under $\mathsf{Dec}_{\mathsf{uniq}} \circ C$ directly implies that y is a remote-point for C, without the error propagating to its children. This allows the reduction to proceed similarly to the Avoid problem, by searching for a preimage on each node of the tree.

However, unique decoding limits us to a radius of $\rho \leq 1/4 - \varepsilon$, which is insufficient for our purposes. In the actual construction, we employ list-decoding to achieve a larger radius of $\rho = 1/2 - \varepsilon$. We use a list-decodable code with a decoder $\mathsf{Dec}_{\mathsf{list}} : \{0,1\}^m \to (\{0,1\}^{qn})^L$. At each node, applying $\mathsf{Dec}_{\mathsf{list}} \circ C$ produces a list of candidate values. We then apply a padding method to pad both the input and the output of C with extra $\log L$ bits. This enables us to select one candidate from this list to pass to the next level of the tree.

Hence we get a conditional $\mathbf{FP}^{\mathbf{NP}}$ for Remote-Point. Combining with a refinement of the result in [RSW22] (Theorem 2.8), it yields an equivalence between the $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point and the average-case circuit lower bound for $\mathbf{E}^{\mathbf{NP}}$.

Subexponential time NC^0-Avoid algorithm for any superlinear stretch. We present the first subexponential-time algorithm for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$, achieving runtime $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ for any $\varepsilon > 0$. Our approach exploits structural limitations of local circuits in terms of their associated bipartite graphs to identify small subcircuits with poor expansion, enabling targeted enumeration over their input-output behavior.

The algorithm is based on the following high-level idea: every $\mathsf{NC}^0_k[n, n^{1+\varepsilon}]$ circuit corresponds to a degree-k left-regular bipartite graph with n right vertices (inputs) and $m = n^{1+\varepsilon}$ left vertices (outputs). Using standard probabilistic methods, one can show that a random left-regular bipartite graph with degree k, n right vertices and $m(n) = n^{1+\varepsilon}$ left vertices is a $(K = o(n), A = 1 - o(1))$ vertex expander — meaning that for every subset of left vertices of size $\leq K$, it has $\geq KA$ neighbors. One would expect these probabilistic arguments to be actually tight. Assuming so, we would be able to find a Hall-violating subsets (i.e., a subset of outputs whose neighbors have size smaller than the subset of outputs) in any such graphs.

Luckily, the lower bound results on disperser graphs from [RTS00] can be adapted to argue that such graphs necessarily contain Hall-violating subsets of outputs of size at most $K = n^{1-\frac{\varepsilon}{k-1}+o(1)}$. This means that every such circuit contains a subcircuit of size K that maps a subset of inputs to outputs non-surjectively.

Our algorithm proceeds by brute-force search for such Hall-violating subsets $S \subseteq [m]$ of size K. Once a violating subset is found, we isolate the corresponding subcircuit C' of size K, and enumerate all strings in $\{0,1\}^{|\Gamma(S)|}$ to find those not in the image of C'. We then lift these local non-image strings to full-length output strings by assigning arbitrary values outside of S, yielding many globally valid strings not in the image of the full circuit C.

This gives the following guarantee: for every $\mathsf{NC}^0_k[n, n^{1+\varepsilon}]$ circuit, we can find (and succinctly represent) at least $2^{n^{1+\varepsilon}-1}$ strings outside the range of the circuit in time

$$O(2^{\binom{m}{K}}) = 2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}.$$

Under a conjectured tight bound on bipartite dispersers, we further refine this analysis to show that even smaller Hall-violating subsets exist, yielding improved runtimes of $2^{n^{1-\frac{\varepsilon}{k-2}+o(1)}}$. Notably, this leads to *polynomial-time* algorithms for NC^0_k-Avoid in stretch regimes as low as $m = n^{k-1}/\log^{k-2} n$, improving a prior work [GGNS23] which required larger stretch.

Finally, we connect our algorithmic result to pseudorandomness. We show that any subexponential-time Avoid algorithm capable of identifying a non-negligible fraction of non-image strings for NC^0_k circuits contradicts the existence of secure NC^0_k-based pseudorandom generators (PRGs) against subexponential-time adversary. In particular, under standard assumptions about local PRGs, our algorithm demonstrates that no such PRG with stretch $n^{1+\varepsilon}$ can be secure against 2^{n^γ}-time distinguishers for any $\gamma \geq 1 - \frac{\varepsilon}{k-1} + o(1)$, even with constant distinguishing advantage.

Improvement over brute-force for NC^0_k-Avoid$[n, n+1]$. We design a greedy, local algorithm for solving NC^0_k-Avoid$[n, n+1]$ that proceeds by iteratively fixing output bits to values that provably shrink the preimage space of the circuit. At each step, the algorithm selects an unfixed output bit y_i and assigns it a value such that the number of inputs consistent with all fixed output values decreases by at least a factor of $1/2$. This ensures that after at most $n+1$ such assignments, the preimage space collapses to an empty set, yielding a string outside the image of the circuit.

The core technical challenge lies in bounding the "decision space", i.e., the portion of the input space that must be explored to determine the effect of fixing an output bit. We analyze this by modeling the NC^0_k circuit as a bipartite dependency graph between input and output bits, and we introduce the notion of the *traversed space*: the subset of input variables affected by the fixed output bits. We show that after fixing t output bits, the maximum size of any connected component (i.e., subspace) in the traversed space is bounded by $2^{(k-2)t+1}$. This follows from structural properties of bounded-locality circuits and a case-based inductive argument.

Combining this with the observation that fixing each output bit reduces the entropy of the input space by one, we find that the decision space remains small as long as $t \leq n/(k-1)$. In particular, the algorithm only needs to examine subspaces of size at most

$$2^{(k-2)n/(k-1)},$$

leading to a total runtime of $O(n \cdot 2^{(k-2)n/(k-1)})$. Notably, when $k = 2$, the runtime becomes linear, reproducing the result of [GLW22]. For larger k, this provides a non-trivial improvement over brute force.

We also show a matching lower bound for this greedy strategy: under mild assumptions on the structure of random NC^0_k circuits (specifically, that they form good bipartite vertex expanders), any such greedy algorithm necessarily explores an exponential-sized decision space in the worst case. This demonstrates that while the algorithm performs well for $k = 2$, solving NC^0_k-Avoid efficiently in the general case may require fundamentally different techniques.

1.3 Subsequent Work

Subsequent to our work, Guruswami, Lyu, and Yuan [GLY25] presented an **FP** algorithm for NC^0_k-Avoid$[n, O_k(n^{(k-1)/2} \log n)]$, which now represents the state-of-the-art polynomial-time algorithm for NC^0-Avoid. They also obtained a $2^{n^{1-\frac{2\varepsilon}{k-3}+o(1)}}$-time algorithm for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$, offering a slight improvement over our subexponential-time algorithm. The rest of our results remain orthogonal to their work.

1.4 Paper Organization

The rest of the paper is organized as follows. In Section 2 we give some preliminary knowledge and some primitives from prior works. In Section 3 we present the generalized Jeřábek-Korten reduction, the conditional $\mathbf{FP}^{\mathbf{NP}}$ algorithm as well as the precise characterization of $\mathbf{E}^{\mathbf{NP}}$ circuit lower bound in terms of $\mathscr{C}$-Avoid problems. In Section 4 we further extend the generalized Jeřábek-Korten reduction to solve Remote-Point problems, giving a conditional $\mathbf{FP}^{\mathbf{NP}}$ algorithm as well as the precise characterization of the average-case circuit lower bound of $\mathbf{E}^{\mathbf{NP}}$. In Section 5 we present the subexponential-time NC^0-Avoid algorithm for any superlinear stretch. In Section 6 we present the non-trivial algorithm for NC^0-Avoid$[n, n+1]$. Finally, we conclude in Section 7 with some open problems.

2 Preliminaries

2.1 Notations

We use $\mathscr{C}$ to denote a circuit class, e.g., $\mathsf{NC}^0, \mathsf{AC}^0, \mathsf{ACC}^0, \mathsf{TC}^0$, etc. We use $\mathscr{C}[n, m(n)]$ to denote $\mathscr{C}$ with input length n and output length $m(n)$. We use $\mathscr{C}_1 \circ \mathscr{C}_2$ to denote the composition of circuits from $\mathscr{C}_1$ and $\mathscr{C}_2$ respectively. We use $\mathscr{C}_{n,s,d}$ to denote all the single-output $\mathscr{C}$ circuit of input length n, size s, and depth d. We use $\mathscr{C}$-Avoid$[n, m(n)]$ to denote $\mathscr{C}$-Avoid problem where the circuit $\mathscr{C}$ has input length n and output length $m(n)$. We call $m(n)$ the *stretch* of the $\mathscr{C}$-Avoid problem.

Given a circuit $C : \{0,1\}^n \to \{0,1\}^m$ where $m > n$. For a partial assignment of an m-bit string y, we use $y \notin \mathrm{Range}(C)$ to denote that any assignment consistent with y is not in the range of the circuit C.

We use $\leq_{\mathbf{FP}}$ (resp. $\leq_{\mathbf{FP}^{\mathbf{NP}}}$) to denote reduction in **FP** (resp. $\mathbf{FP}^{\mathbf{NP}}$).

For two strings $x, y \in \{0,1\}^N$, define the *relative Hamming Distance* to be the fraction of indices where x and y differ, formally $\delta(x, y) := \frac{1}{N}|\{i \in [N] : x_i \neq y_i\}|$. For a string $x \in \{0,1\}^N$ and a subset $S \subset \{0,1\}^N$, we say that x is ρ-close/far to S iff $\min_{y\in S}\delta(x,y) \le \rho/\min_{y\in S}\delta(x,y) > \rho$. When $S = \{y\}$, we also say that x is ρ-close/far to y.

We use PRGs to denote pseudorandom generators. We use $\mathsf{Bip}_{n,m,D}$ to be the set of bipartite multigraphs that have m left vertices and n right vertices where $m \ge n+1$ and are D-left regular. We often use capital letters for random variables and corresponding small letters for their instantiations. Let s be an integer, $\{V_1, V_2, \cdots, V_s\}$ be a set of random variables. We use $V_{[s]}$ to denote the subset $\{V_1, \cdots, V_s\}$. For any strings y_1 and y_2, let $y_1 \circ y_2$ denote the concatenation of y_1 and y_2. Let F_2 denote the binary field.

We will adopt 0-index, e.g., the first bit of s string s is s_0, the first child of a parent in a tree is its 0-th child, etc. The height of a tree is referred to as the number of edges in the longest path from the root node to any leaf node.

2.2 Formulas, NC Circuits and AC Circuits

We use standard definitions of circuit complexity classes. A Boolean circuit is a directed acyclic graph composed of logic gates with bounded fan-in (e.g., $\wedge$, $\vee$, $\neg$) computing functions over $\{0,1\}$. A family of circuits $\{C_n\}_{n\in\mathbb{N}}$ is said to compute a function $f : \{0,1\}^* \to \{0,1\}^*$ if, for every input length n, the circuit C_n correctly computes f on inputs of length n. We use the size s of a circuit as its number of gates plus the length of output, and the depth d to denote the length of the longest path between input bits and output bits.

A formula is a specific type of circuit where the fan-out of every gate is restricted to *exactly one*. This means the output of each gate can be used as the input to *at most one* other gate, *or* it may serve as *exactly one* bit of the output.

Definition 2.1 (NC circuits [GGNS23]). *The circuit class NC^i contains multi-output Boolean circuits on n inputs of depth $O(\log^i n)$ where each gate has fan-in 2. We are particularly concerned with the following classes of circuits:*

- *For every constant $k \ge 1$, NC^0_k is the class of circuits where each output depends on at most k inputs.*
- *NC^1 is the class of circuits of depth $O(\log n)$ where all gates have fan-in 2.*
- *Linear NC^1 circuits are circuits of depth $O(\log n)$ where every gate has fan-in 2 and computes an affine function, i.e., the XOR of its two inputs or its negation.*

Proving a super-linear circuit lower bound on the size of arithmetic computing an n-output function from **FP** or even $\mathbf{FE^{NP}}$ [GGNS23, Val77, AB09, Frontier 3] is a decades-old challenge. Valiant [Val77] introduced the problem of explicitly constructing rigid matrices and showed that this would prove super-linear lower bounds on the size of (linear) NC^1 circuits.

Definition 2.2 (AC Circuits). *We denote by AC^i the class of Boolean functions computable by a family of circuits of:*

- *polynomial size,*
- *depth $O(\log^i n)$,*
- *unbounded fan-in $\wedge$ and $\vee$ gates,*
- *and $\neg$ gates allowed only at the input level and are not counted into the depth.*

We say a function f is in AC^i if it is computed by a family of AC^i circuits. The class AC is defined as the union $\mathsf{AC} = \bigcup_{i \geq 0} \mathsf{AC}^i$.

We use the notation AC^i_d to denote the family of AC^i circuits with depth at most d.

More generally, an AC^i-circuit of size $s(n)$, where $s(n)$ may be super-polynomial of n, is defined identically to an AC^i circuit but relaxing the size restriction from polynomial to $s(n)$.

For a *correlation* factor $2\gamma > 0$, we say that a circuit $C : \{0,1\}^n \to \{0,1\}$ $(1/2+\gamma)$-approximates a function $f : \{0,1\}^n \to \{0,1\}$ if $C(x) = f(x)$ for $(1/2+\gamma)$ fraction of inputs from $\{0,1\}^n$. Let $N := 2^n$, and the truth table of C be $\mathsf{TT}_C \in \{0,1\}^N$, the truth table of f be $\mathsf{TT}_f \in \{0,1\}^N$. Then the above is equivalent to $\delta(\mathsf{TT}_C, \mathsf{TT}_f) < (1/2 - \gamma)$.

For a function $f : \{0,1\}^n \to \{0,1\}$, we define $\mathsf{SIZE}(f)$ to be the minimum size of a circuit computing f exactly. Similarly, for $\gamma > 0$, we define $\mathsf{Avg}_\gamma\text{-}\mathsf{SIZE}(f)$ to be the minimum size of a circuit that $(1/2+\gamma)$-approximates f.

We use $\mathsf{SIZE}[s(n)]$ to denote the set of functions with boolean circuit complexity $s(n)$. We use $\mathscr{C}$-$\mathsf{SIZE}[s(n)]$ to denote the set of functions with $\mathscr{C}$ circuit complexity $s(n)$. We use Avg_γ-$\mathscr{C}$-$\mathsf{SIZE}[s(n)]$ to denote the set of functions that can be $(1/2+\gamma)$-approximated by $\mathscr{C}$ with circuit complexity $s(n)$.

We use $\mathsf{Formula}[s(n)]$ to denote the set of functions that can be computed by size-$s(n)$ boolean formulas.

Definition 2.3 (($\mathscr{C}$) Circuit Complexity of a String). *Given a bit string $s \in \{0,1\}^n$, we define the ($\mathscr{C}$) circuit complexity of s to be the smallest ($\mathscr{C}$) circuit whose truth table agrees with s for the first n indices. In particular, the formula complexity of s to be the smallest formula whose truth table agrees with s for the first n indices.*

2.3 Universality Property and Truth Table Generator

Definition 2.4 (Universality Property [RSW22]). *Let $\mathscr{C}$ be a circuit class. We say that $\mathscr{C}$ has the universality property if there is a constant $c \geq 1$ such that for any good function $s : \mathbb{N} \to \mathbb{N}$, there is a sequence of $\mathscr{C}$ circuits $\{U_{s,n}\}_{n \in \mathbb{N}}$ such that the following are true:*

- *The size of $U_{s,n}$ is $s(n)^c$ and it has $O(s \log s + n)$ variables.*
- *Given an input $(\langle C \rangle, x)$, where $\langle C \rangle$ is the encoding of a $\mathscr{C}$ circuit C of size s on n variables, and $x \in \{0,1\}^n$, it accepts the input iff C accepts x.*
- *The family $U_{s,n}$ is uniform: there is a Turing machine that on input $(1^s, 1^n)$, outputs the description of $U_{s,n}$ in polynomial time.*

Theorem 2.5 ([CH85]). *The class AC^0 has universality property.*

Theorem 2.6 ([Bus87]). *The class NC^1 has universality property.*

In effect, any circuit class containing AC^0 has universality property. We include in Appendix A for a detailed proof.

Definition 2.7 (Truth Table Generator). *Let $\mathsf{TT} : \{0,1\}^{O(s \log s)} \to \{0,1\}^{2^n}$ be the circuit that takes as input the description of a size-s circuit on n variables, and outputs the truth table of this circuit. Here TT denotes* truth table. *Define $\mathsf{TT}_{\mathscr{C}} : \{0,1\}^{O(s \log s)} \to \{0,1\}^{2^n}$ to be the circuit that takes as input the description of a size s $\mathscr{C}$ circuit on n variables, and outputs the truth table of this $\mathscr{C}$ circuit. It is clear that if $\mathscr{C}$ has universality property, then $\mathsf{TT}_{\mathscr{C}} \in \mathscr{C}$.*

The following modified Theorem says that solving $\mathscr{C}$-Remote-Point on $\mathsf{TT}_{\mathscr{C}}$ implies $\mathscr{C}$ circuit lower bounds with tight parameters (see Appendix C for a proof).

Theorem 2.8 (Modified Theorem 5.2 of [RSW22])**.** *Let $\mathscr{C}$ be any circuit class that has the universality property, and $c, f : \mathbb{N} \to \mathbb{N}$ be monotone functions that are good. Suppose there is an $\mathbf{FP}^{\mathbf{NP}}$ (resp. $\mathbf{FP}$, $\mathbf{FQP}^{\mathbf{NP}}$) algorithm for $\mathscr{C}$-Remote-Point$[N, f(N), c(N)]$, where each output gate has $\mathscr{C}$ circuit complexity* $\mathrm{poly}(N)$*. Then for some constant $\varepsilon > 0$, $\mathbf{E}^{\mathbf{NP}}$ (resp. $\mathbf{E}$, $\mathbf{EXP}^{\mathbf{NP}}$) cannot be $(1/2 + c(f^{-1}(2^n)))$ approximated by $\mathscr{C}$ circuits of size $\frac{\varepsilon f^{-1}(2^n)}{\log f^{-1}(2^n)}$.*

2.4 Error-correcting Code

Here we will quickly review the basic concepts from coding theory that will be needed for this work. A binary code $\mathcal{C}$ of block length n' is a subset of $\{0,1\}^{n'}$. We use $n = \log |\mathcal{C}|$ to denote the message length of $\mathcal{C}$, and the rate of $\mathcal{C}$ equals n/n'. Each string in $\mathcal{C}$ is called a codeword. The distance of $\mathcal{C}$ is defined as $\min_{x \neq x'} \delta(x, x')$ where $x, x' \in \mathcal{C}$.

A list decoding algorithm for a binary code $\mathcal{C}$ of block length n' needs to do the following. Given an error parameter $0 \leq \rho < 1$ and a received word $y \in \{0,1\}^{n'}$ the decoder needs to output all codewords $c \in \mathcal{C}$ such that $\delta(c, y) \leq \rho$. We say that a code $\mathcal{C}$ of block length n' is (ρ, L)-list-decodable, if for every such y, there are at most L codewords which satisfy $\delta(c, y) \leq \rho$.

Definition 2.9 ((n, n', ρ, L)-code)**.** *For a binary code $\mathcal{C}$ of block length n' and message length n, an encoding function for $\mathcal{C}$ is a bijection $\mathsf{Enc} : \{0,1\}^n \to \mathcal{C}$ (assume w.l.o.g that n is an integer), which can also be extended as an injection from $\{0,1\}^n$ to $\{0,1\}^{n'}$. Since $\mathcal{C}$ and Enc are essentially the same object, we will use Enc to refer to $\mathcal{C}$.*

Suppose that Enc is (ρ, L)-list-decodable, and use $\mathsf{Dec} : \{0,1\}^{n'} \to (\{0,1\}^n)^L$ to denote the list decoding algorithm for it. Then we call that $(\mathsf{Enc}, \mathsf{Dec})$ is a (n, n', ρ, L)-code, which means that Enc has message length n and block length n', as well as its list decoding algorithm Dec.

We often need to select a specific block of the list decoded from the codeword. So we define the following notation:

Definition 2.10 (Selector of list-decoding)**.** *For a (n, n', ρ, L)-code $(\mathsf{Enc}, \mathsf{Dec})$, its selector $\mathsf{Sel}_{\mathsf{Dec}} : \{0,1\}^{n'} \times [L] \to \{0,1\}^n$ outputs the z-th block of $\mathsf{Dec}(w)$ over the input $w \in \{0,1\}^{n'}$ and $z \in [L]$. W.l.o.g, assume that $\log L$ is an integer, and we also view the input domain as $\{0,1\}^{n'+\log L}$ where the first n' bits form the codeword, and the remaining $\log L$ bits represent an integer in $[L]$.*

The classic Johnson bound [Joh62] implies that *non-explicitly* a binary code of relative distance $1/2 - \varepsilon^2$ is $(1/2 - \varepsilon, 1/\varepsilon^2)$-list-decodable. When we require that both the encoding and list-decoding algorithms run efficiently, Guruswami and Rudra [GR08, Gur09] showed that:

Theorem 2.11 (Theorem 13 of [GR08])**.** *Given an integer $n > 1$ and reals $\gamma > 0$ and $0 < \varepsilon < 1/2$, there exists an explicit binary code Enc with message length n and block length at most $(1/\gamma)^{O(1)} \cdot (n^3/\varepsilon^{3+\gamma})$, which is $\left(\frac{1}{2} - \varepsilon, \left(\frac{1}{\gamma\varepsilon}\right)^{O(1/\gamma)}\right)$-list-decodable and the list decoding algorithm Dec runs in time $\left(\frac{n}{\gamma\varepsilon}\right)^{O(1/\gamma)}$.*

Specifically, there exists a $\left(n, O(n^{3(c+1)+\gamma}), 1/2 - n^{-c}, \mathrm{poly}(n)\right)$-code $(\mathsf{Enc}, \mathsf{Dec})$ for any constant $c, \gamma > 0$, where both Enc and Dec run in $\mathrm{poly}(n)$ *time.*

2.5 Bipartite Vertex Expander

Definition 2.12 (Vertex expander [Vad12]). *A digraph G is a (K, A) vertex expander if for all sets S of at most K vertices, the neighborhood $N(S) = \{u : \exists v \in S \text{ s.t. } (u, v) \in E\}$ is of size at least $A \cdot |S|$.*

Definition 2.13 (Left regular bipartite graphs [Vad12]). *Let $\mathsf{Bip}_{n,m,D}$ be the set of bipartite multigraphs that have m left vertices and n right vertices where $m \geq n+1$ and are D-left-regular, meaning that every vertex on the left has D neighbors, but vertices on the right may have varying degrees.*

We use (K, A)-$\mathsf{Bip}_{n,m,D}$ to denote $G \in \mathsf{Bip}_{n,m,D}$ that are also (K, A) vertex expander.
The following Theorem 2.14 and Theorem 2.15 are modified from [Vad12].

Theorem 2.14 (Existence of $(\Omega(n), D - 1 - \varepsilon)$-$\mathsf{Bip}_{n,m,D}$). *For every constant D, $0 < \varepsilon < 1$, there exists a constant $\alpha > 0$ such that for all n, $m = O(n)$, a uniformly random graph from $\mathsf{Bip}_{n,m,D}$ is an $(\alpha n, D - 1 - \varepsilon)$ vertex expander with probability at least $1/2$.*

Theorem 2.15 (Existence of $(o(n), 1)$-$\mathsf{Bip}_{n,m,D}$). *For every constant D and every $0 < \beta < 1$, there exists a function $A = n^{1-\beta/(D-2)}$ such that for all n, and $m = n^{1+\beta}$, a uniformly random graph from $\mathsf{Bip}_{n,m,D}$ is an $(A, 1)$ vertex expander with probability at least $1/2$.*

The following definition of Hall-violating set stems from Hall's matching theorem.

Definition 2.16 (Hall-violating set). *In a bipartite graph G with bipartite classes L and R, a set $H \subseteq L$ is a Hall-violating set if $|N(H)| < |H|$.*

Disperser graphs are special cases of bipartite expanders.

Definition 2.17 (Disperser graphs [Sip86, CW89]). *A bipartite graph $G = (V_1 = [N], V_2 = [M], E)$ is a (K, ε)-disperser graph, if for every $X \subseteq V_1$ of cardinality K, $|\Gamma(X)| > (1 - \varepsilon)M$ (i.e., every large enough set in V_1 misses less than an ε fraction of the vertices of V_2). The size of G is $|E(G)|$.*

The following theorem gives necessary conditions for G to be a disperser.

Theorem 2.18 (Lower bounds for disperser graphs [RTS00]). *Let $G = (V_1 = [N], V_2 = [M], E)$ be a (K, ε)-disperser. Denote by $\bar{D}$ the average degree of a vertex in V_1.*

1. *Assume that $K < N$ and $\lceil \bar{D} \rceil \leq \frac{(1-\varepsilon)M}{2}$ (i.e., G is not trivial). If $\frac{1}{M} \leq \varepsilon \leq \frac{1}{2}$, then $\bar{D} = \Omega(\frac{1}{\varepsilon} \cdot \log \frac{N}{K})$, and if $\varepsilon > \frac{1}{2}$, then $\bar{D} = \Omega(\frac{1}{\log(1/(1-\varepsilon))} \cdot \log \frac{N}{K})$.*
2. *Assume that $K \leq \frac{N}{2}$ and $\bar{D} \leq \frac{M}{4}$. Then, $\frac{\bar{D}K}{M} = \Omega(\log \frac{1}{\varepsilon})$.*

2.6 Local Algorithms

A local algorithm for Avoid problems probes very few bits to determine any particular output bit of the string out of the range. A local algorithm for a related problem $\mathsf{Missing\text{-}String}$ was proposed in [VW23].

2.7 Some Assumptions

Assumption 2.19 ([RSW22])**.** *For every constants $k \geq 1$ and $\varepsilon > 0$, there is an $\mathbf{FP^{NP}}$ algorithm that given any k-uniform directed hypergraph G and any predicate $P : \{0,1\}^k \to \{0,1\}$, outputs a P-sparsifier of G with error $\varepsilon = 0.5$ using $\tilde{O}(n)$ hyperedges.*

Assumption 2.20 ([JLS21])**.** *There exists a boolean function $G : \{0,1\}^n \to \{0,1\}^m$ where $m = n^{1+\tau}$ for some constant $\tau > 0$, and where each output bit computed by G depends on a constant number of input bits, such that the following computational indistinguishability holds:*

$$\{G(\sigma) \mid \sigma \leftarrow \{0,1\}^n\} \approx_c \{y \mid y \leftarrow \{0,1\}^m\}$$

The subexponential security of PRG requires the above indistinguishability to hold for adversaries of size 2^{n^β} for some constant $\beta > 0$, with negligible distinguishing advantage.

3 Generalized GGM-Tree and Conditional $\mathbf{FP^{NP}}$ Algorithms

In light of the difficulty in obtaining an unconditional $\mathbf{FP^{NP}}$ algorithm for AC^0-Avoid$[n, \mathrm{qpoly}(n)]$ and NC^0-Avoid$[n, n + o(n)]$ [RSW22], we turn our attention to exploring which assumptions might yield such an $\mathbf{FP^{NP}}$ algorithm for AC^0-Avoid and NC^0-Avoid.

Korten [Kor21] observed that Avoid admits an $\mathbf{FZPP^{NP}}$ algorithm. Moreover, he, building on the work of Jeřábek [Jeř04], obtained a conditional derandomization of this algorithm under assumptions (e.g., $\mathbf{E^{NP}}$ requires circuits of size $2^{\Omega(n)}$) significantly weaker than those required by standard approaches (which typically demand, for example, that $\mathbf{E}$ requires SAT-oracle circuits of size $2^{\Omega(n)}$ [KvM02]). His approach, which we have dubbed Jeřábek-Korten reduction in the introduction, also inspired a recent breakthrough achieving near-maximal circuit lower bounds against $\mathbf{S_2E}$ [CHR24, Li24].

These developments motivate us to explore generalizations of Jeřábek-Korten reduction aimed at derandomizing the $\mathbf{FZPP^{NP}}$ algorithm for restricted circuit classes $\mathscr{C}$, specifically NC^0 and constant-depth unbounded fan-in circuit classes containing AC^0.

3.1 Generalized Jeřábek-Korten Reduction

We now define a generalized GGM-tree and demonstrate that it characterizes the feasibility of solving $\mathscr{C}$-Avoid in $\mathbf{FP^{NP}}$, even when $\mathscr{C}$ is as weak as NC^0 or AC^0. Previously, such tight correspondences were only known for unrestricted circuit classes.

Generalized GGM-tree Construction $\mathsf{GGM}_{\ell,q,k}[C]$: Given a circuit $C : \{0,1\}^n \to \{0,1\}^{qn}$, the height k and a parameter $\ell = nq^k$, construct $\mathsf{GGM}_{\ell,q,k}[C] : \{0,1\}^n \to \{0,1\}^\ell$ as follows: On the input $x \in \{0,1\}^n$, the output $\mathsf{GGM}_{\ell,q,k}[C](x)$ is defined as:

1. Build a valued perfect q-ary tree of height k. Let (i,j) denote the j-th node at level i $(0 \leq i \leq k, 0 \leq j < q^i)$.
   - For each $0 \leq i < k$, $0 \leq j < q^i$ and $0 \leq h < q$, the h-th child of node (i,j) is node $(i+1, qj+h)$.
   - The value on node (i,j) is denoted by $v_{i,j}$.
2. Set $v_{0,0} = x$.

3. At each node (i,j) with $i<k$, compute $y=C(v_{i,j})$ and assign the $(h+1)$-th block of n bits of y to $v_{i+1,qj+h}$, for any $0 \le h < q$.
4. Finally, set $\mathsf{GGM}_{\ell,q,k}[C](x) = v_{k,0} \circ \cdots \circ v_{k,q^k-1}$.

When $q=2$, we get the classic binary GGM-tree, showed in Figure 2.

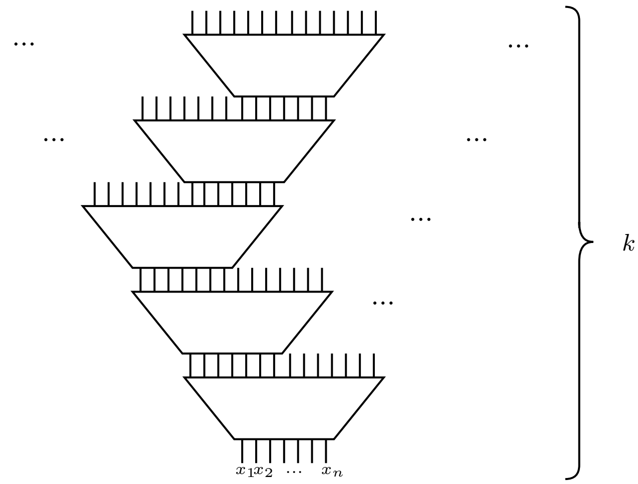

Figure 2: Apply the circuit $C:\{0,1\}^n \to \{0,1\}^{2n}$ over the GGM-tree of height k to obtain the circuit $C^*:\{0,1\}^n \to \{0,1\}^{2^k n}$.

Circuit Complexity of the Output. Like the analysis in [Kor21], we show that due to the simple repetitive nature of the GGM-tree's structure, strings produced by it all have very low circuit complexity.

Theorem 3.1. *Let $C:\{0,1\}^n \to \{0,1\}^{qn}$ be a circuit where with size s_C. Let $C^* = \mathsf{GGM}_{\ell,q,k}[C]$ have tree height k. Then for any $x \in \{0,1\}^n$, it follows that:*

1. *The circuit complexity of $C^*(x)$ is at most $O(s_C \cdot k)$*
2. *Let be $\mathscr{C}$ be a constant depth unbounded fan-in circuit class containing AC^0. If we further guarantee that C is a $\mathscr{C}$ circuit with size s_C and $k=O(1)$, then the $\mathscr{C}$ circuit complexity of $C^*(x)$ is at most $O(s_C \cdot k)$.*
3. *We can also extend the result to the case of formula complexity. Given that C can be implemented by a formula of size s'_C, the formula complexity of $C^*(x)$ is at most $O(s'_C \cdot k)$.*

Proof. We first consider the classic circuit case. Figure 3 illustrates a succinct circuit $g:\{0,1\}^{\log \ell} \to \{0,1\}$ whose truth table corresponds to a string $y = C^*(x) \in \mathrm{Range}(C^*)$.

In general, the succinct circuit simulates a root-to-leaf path. And it can be constructed as follows: It consists of k instances of the circuit C concatenated in series. Between each pair of consecutive C's, a path selector (multiplexer) is incorporated to choose the specific n-bit block corresponding to the chosen child node at that level of the tree. Finally, an output selector is added at the end to extract a single bit from the final n-bit output of the leaf node, based on a given index.

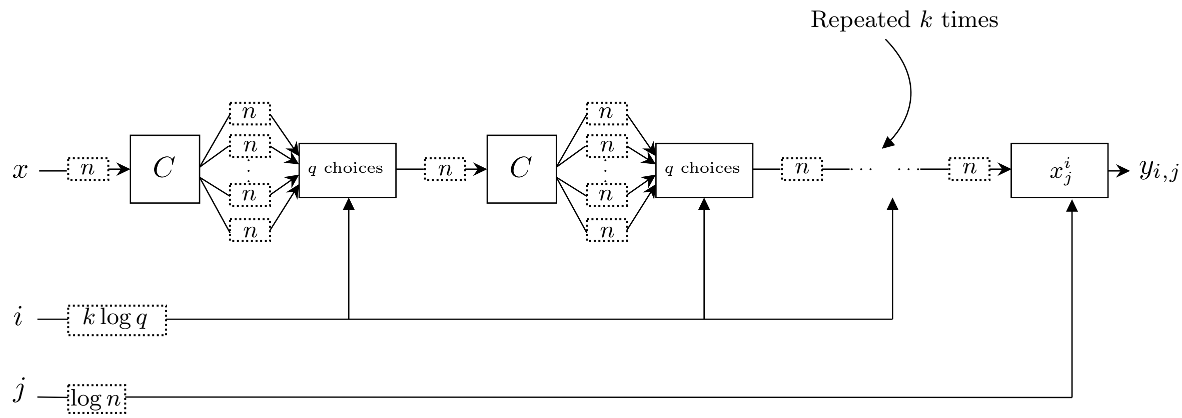

Figure 3: A succinct circuit whose truth table is $y = C^*(x)$, for any y in the range of C^* [Kor21, Figure 2]. Each C circuit has input length n and output length qn, and is overlayed by q new C circuits in the next level. As in [Kor21], dotted boxes indicate the number of bits along a wire; x is hardwired as constants/advice for any given y. The only true inputs to this circuit are i, j.

To see that g has small circuit complexity, we note that each path selector, which chooses one n-bit block from q such blocks, can be computed easily with $O(qn)$ $\wedge$, $\vee$ or $\neg$ gates. Given that s_C is at least qn, the total size of the succinct circuit is $O(s_C \cdot k)$.

For formula complexity, we just observe that the multiplexer can be also implemented by a formula with size $O(qn)$.

For $\mathscr{C}$ complexity, since $\mathscr{C}$ can use gates with unbounded fan-in, the multiplexer can be also implemented by a $\mathscr{C}$ circuit with size $O(qn)$. Additionally, the extra requirement $k = O(1)$ guarantees that the depth is still a constant, which shows that the succinct circuit still belongs to $\mathscr{C}$, as desired. $\square$

Consequently, any string $y \in \{0,1\}^\ell$ with circuit complexity exceeding $O(s_C \cdot k)$ must lie outside $\mathrm{Range}(C^*)$.

Modified Jeřábek-Korten Reduction. We give a variant of the Jeřábek-Korten reduction based on the generalized GGM-tree.

Algorithm 1: $\mathsf{Je\check{r}\acute{a}bek\text{-}Korten}''(C, f)$: Modified Jeřábek-Korten Reduction for q-ary GGM-Tree

Input: A circuit $C : \{0,1\}^n \to \{0,1\}^{qn}$, the height k and a string $f \notin \mathrm{Range}(\mathsf{GGM}_{\ell,q,k}[C])$.
Output: A string $y \notin \mathrm{Range}(C)$.

1 **for** $j \leftarrow 0$ *to* $q^k - 1$ **do**
2     $v_{k,j} \leftarrow f_{[nj,n(j+1))}$;
3 **end**
4 **for** $i = k - 1$ *to* 0 **do**
5   **for** $j = 0$ *to* $q^i - 1$ **do**
6   Use the **NP** oracle to find the lexicographically smallest $v_{i,j}$ such that $C(v_{i,j}) = v_{i+1,qj} \circ \cdots \circ v_{i+1,qj+q-1}$;
7   **if** $v_{i,j}$ *does nost exist* **then**
8   **return** $v_{i+1,qj} \circ \cdots \circ v_{i+1,qj+q-1}$;
9   **end**
10   **end**
11 **end**
12 **return** $\perp$;

This framework enables us to efficiently recover a string not in the range of C, given one outside the range of $\mathsf{GGM}_{\ell,q,k}[C]$.

Lemma 3.2. *Given that $f \notin \mathrm{Range}(\mathsf{GGM}_{\ell,q,k}[C])$, Algorithm 1 guarantees to find a $y \in \{0,1\}^{qn}$ such that $y \neq \perp$ and $y \notin \mathrm{Range}(C)$. Moreover, Algorithm 1 only needs $\mathrm{poly}(\ell) = \mathrm{poly}(n, q^k)$ calls for* **NP** *oracle.*

Proof. The running time of Algorithm 1 is trivially $\mathrm{poly}(\ell)$. For correctness, if $y \neq \perp$, the algorithm actually finds a string out of $\mathrm{Range}(C)$ and returns it in line 8.

Now assume that $y = \perp$, i.e. the algorithm returns an empty string. Then each $v_{i,j}$ exists, and therefore $C(v_{i,j}) = v_{i+1,qj} \circ \cdots \circ v_{i+1,qj+q-1}$ for any $i < k$.

This tells us $\mathsf{GGM}_{\ell,q,k}[C](v_{0,0}) = v_{k,0} \circ \cdots \circ v_{k,q^k-1} = f$, which contradicts the fact that $f \notin \mathrm{Range}(\mathsf{GGM}_{\ell,q,k}[C])$. $\square$

3.2 Conditional $\mathbf{FP}^{\mathbf{NP}}$ Algorithm for NC^0-Avoid$[n, 2n]$

In this section, we show that assuming near-maximum ($\Omega(2^n/n)$) size formula lower bound against $\mathbf{E}^{\mathbf{NP}}$, we can obtain an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-Avoid$[n, 2n]$.

Theorem 3.3. *If $\mathbf{E}^{\mathbf{NP}}$ requires near-maximum ($\Omega(2^n/n)$) size formulas, then there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-Avoid$[n, 2n]$.*

Proof. Let $C : \{0,1\}^n \to \{0,1\}^{2n}$ be a circuit in NC^0. It can be computed by a formula with size $s_C = O(n)$. Consider applying the generalized GGM-tree construction $C^* = \mathsf{GGM}_{\ell,2,k}[C]$ with height $k = t \cdot \log\log n$ for a sufficiently large constant t. Then the output length $\ell = n \cdot 2^k = n \log^t n$. For each $y \in \mathrm{Range}(C^*)$, by the third statement of Theorem 3.1, the formula complexity of y is $O(s_C \cdot k) = O(n \log\log n) = o(\ell/\log \ell)$.

Consequently, we get that any string $f \in \{0,1\}^\ell$ with formula complexity $\Omega(\ell/\log\ell)$ lies outside the range of C^*. And such a string can be found in $2^{O(\log \ell)} = \mathrm{poly}(\ell) = \mathrm{poly}(s_C)$ time by our assumption about the formula lower bound of $\mathbf{E}^{\mathbf{NP}}$. Conclusively, given such a string f, we can invoke Algorithm 1 to recover a string not in $\mathrm{Range}(C)$, thereby obtaining an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-Avoid$[n, 2n]$. $\square$

Combining with Theorem 1.5, we nearly pin down the hardness of proving $\mathbf{E}^{\mathbf{NP}}$ requires exponential size formulas in terms of NC^0_4-AVOID algorithm: proving such a lower bound should be no harder than proving $\mathsf{NC}^0_4\text{-Avoid}[n, n+n^\delta] \in \mathbf{FP}^{\mathbf{NP}}$ for any $\delta \in (0,1)$, but should be no easier than $\mathsf{NC}^0_4\text{-Avoid}[n, 2n] \in \mathbf{FP}^{\mathbf{NP}}$.

3.3 Conditional $\mathbf{FP}^{\mathbf{NP}}$ Algorithm for $\mathscr{C}$-AVOID$[n, n^{1+\varepsilon}]$

We now extend our generalized framework to establish an equivalence between lower bounds against a circuit class $\mathscr{C}$ and the existence of $\mathbf{FP}^{\mathbf{NP}}$ algorithms for $\mathscr{C}$-AVOID, under mild stretch.

Theorem 3.4. *Let $\mathscr{C}$ be a constant depth unbounded fan-in circuit class satisfying $\mathsf{AC}^0 \subseteq \mathscr{C}$. Then the following are equivalent:*

1. *$\mathbf{E}^{\mathbf{NP}}$ does not have $2^{o(n)}$-size $\mathscr{C}$ circuits;*
2. *For every constant $\varepsilon > 0$, there exists an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-AVOID$[n, n^{1+\varepsilon}]$.*

Proof. (" $\Longleftarrow$ ") This direction follows from the universality of $\mathscr{C}$, as formalized in Theorem 2.8 (just let $c(n) \equiv 1/2$, and we get the corresponding theorem for AVOID). Specifically, if $\mathsf{TT}_{\mathscr{C}}$ can be implemented within $\mathscr{C}$, then the existence of an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-AVOID implies that $\mathbf{E}^{\mathbf{NP}}$ requires exponential-size $\mathscr{C}$ circuits. See Appendix A for a detailed proof.

(" $\Longrightarrow$ ") We now show that assuming $\mathbf{E}^{\mathbf{NP}}$ requires $2^{\tau n} (= 2^{\Omega(n)})$ size $\mathscr{C}$ circuits, one can obtain an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-AVOID$[n, n^{1+\varepsilon}]$, for any constant $\varepsilon > 0$, via the generalized GGM construction.

Let $C : \{0,1\}^n \to \{0,1\}^{n^{1+\varepsilon}}$ be an instance of $\mathscr{C}$-AVOID$[n, n^{1+\varepsilon}]$, where each output bit of C is computed by a $\mathscr{C}$ circuit of size $s_C = n^t$.

Let us construct $C^* = \mathsf{GGM}_{\ell,q,k}[C]$ with parameters $q = n^\varepsilon$ and $k = O(1)$. Then the output length is $\ell = n \cdot q^k = n^{1+k\varepsilon}$. By the second property of Theorem 3.1, the circuit complexity of any $y \in \mathrm{Range}(C^*)$ is bounded by $O(s_C \cdot k) = O(n^t)$.

Now suppose there exists a string $y^* \in \{0,1\}^\ell$ with $\mathscr{C}$ circuit complexity $\geq \ell^\tau = n^{\tau(1+k\varepsilon)}$ for some constant $0 < \tau < 1$. Since $\tau(1+k\varepsilon) > t$ (by choosing k appropriately), it follows that $y^* \notin \mathrm{Range}(C^*)$.

Actually, according to our assumption of the $\mathscr{C}$ circuit lower bound of $\mathbf{E}^{\mathbf{NP}}$, such a string can be computed in $\mathrm{poly}(\ell) = \mathrm{poly}(n)$ time. Thus applying Algorithm 1 on input C and y^* allows us to find a string outside $\mathrm{Range}(C)$, and this yields an $\mathbf{FP}^{\mathbf{NP}}$ algorithm. $\square$

The above proof also extends to the setting of $\mathbf{FQP}^{\mathbf{NP}}$ algorithms and corresponding lower bounds for $\mathbf{EXP}^{\mathbf{NP}}$. Intuitively, if one can construct the truth table of a length-ℓ function in quasi-polynomial time, then the hard function lies in $\mathbf{EXP}$. Combined with Theorem 2.8, this yields the following theorems.

Theorem 3.5. *For any constant depth unbounded fan-in circuit class $\mathscr{C}$ such that $\mathsf{AC}^0 \subseteq \mathscr{C}$, $\mathbf{EXP}^{\mathbf{NP}}$ requires $2^{\Omega(n)}$ size $\mathscr{C}$ circuits if and only if there is an $\mathbf{FQP}^{\mathbf{NP}}$ algorithm for $\mathscr{C}$-AVOID$[n, n^{1+\varepsilon}]$ for any constant $\varepsilon > 0$.*

The smallest circuit class of the equivalence result is AC^0. However, it is also an intriguing question to obtain $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-AVOID$[n, n^{1+\varepsilon}]$.

Remark 3.6. *Instantiating the same framework for $\mathscr{C} = \mathsf{NC}^0$ yields that $\mathbf{E}^{\mathbf{NP}}$ requires exponential-size $(\mathsf{DNF} \circ \mathsf{NC}^0)^k \circ \mathsf{DNF}$ circuits (k being the depth of the GGM tree) $\Longrightarrow$ an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for NC^0-AVOID$[n, n^{1+\varepsilon}]$.*

4 Generalization of Jeřábek-Korten Reduction to Remote-Point

As we mentioned in the introduction, the Remote-Point problem $\mathrm{RPP}[n, m(n), c(n)]$ is the average-case analog of $\textsc{Avoid}[n, m(n)]$. Algorithms for Remote-Point imply average-case lower bound.

For example, by the work of [CHLR23], it is known that the state-of-the-art $\mathbf{FP}^{\mathbf{NP}}$ algorithm for ACC^0-Remote-Point recovers the best-known almost-everywhere average-case lower bounds[13] against ACC^0 circuits by Chen, Lyu, and Williams [CLW20].

However, it was not known whether the reverse is true. In the following, we extend the generalized Jeřábek-Korten reduction to Remote-Point, and use it to prove an equivalence between an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point and the average-case circuit lower bound for $\mathbf{E}^{\mathbf{NP}}$.

Modified GGM-tree Construction $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})]$ for Remote-Point: Given a circuit $C : \{0,1\}^n \to \{0,1\}^m$, a $(q(n + \log L), m, \delta, L)$-code $(\mathsf{Enc}, \mathsf{Dec})$, the height k and a parameter $\ell = (m + \log L) \cdot q^{k-1}$, construct $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})]$ as follows: On the input $x \in \{0,1\}^{n+\log L}$, the output $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})](x)$ is defined as:

1. Define a padded circuit $\widetilde{C} : \{0,1\}^n \times \{0,1\}^{\log L} \to \{0,1\}^{m+\log L}$: $\widetilde{C}(w, z) = C(w) \circ z$, $\forall w \in \{0,1\}^n, z \in \{0,1\}^{\log L}$.

2. Build a valued perfect q-ary tree of height $k - 1$. Let (i, j) denote the j-th node at level i ($0 \le i < k, 0 \le j < q^i$).

   - For each $0 \le i < k - 1$, $0 \le j < q^i$ and $0 \le h < q$, the h-th child of node (i, j) is node $(i + 1, qj + h)$.
   - The value on node (i, j) is denoted by $v_{i,j}$.

3. Set $v_{0,0} = x$.

4. At each node (i, j) with $i < k - 1$, we first compute $y = \mathsf{Sel}_{\mathsf{Dec}}(\widetilde{C}(v_{i,j}))$ (recall the definition of selectors in Definition 2.10, and here we view the first m bits of $\widetilde{C}(v_{i,j})$ as the codeword, while the last $\log L$ bits form an integer in $[L]$). Then assign the $(h+1)$-th block of $n + \log L$ bits of y to $v_{i+1,qj+h}$, for any $0 \le h < q$.

5. Finally, set $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})](x) = \widetilde{C}(v_{k-1,0}) \circ \cdots \circ \widetilde{C}(v_{k-1,q^{k-1}-1})$.

Comparing two constructions for Avoid and Remote-Point, for Remote-Point, we replace C on each non-leaf node by and $\mathsf{Sel}_{\mathsf{Dec}} \circ \widetilde{C}$, and each leaf node by $\widetilde{C}$. Figure 4 demonstrates the modified GGM-tree construction for $q = 2$.

And the following algorithm can be used in place of Algorithm 1 to obtain Remote-Point algorithms from a suitable average-case lower bound.

[13] Typically, a strong average-case lower bound states that certain problems cannot be $(1/2 + 1/s)$-approximated by size-s circuits [CHLR23].

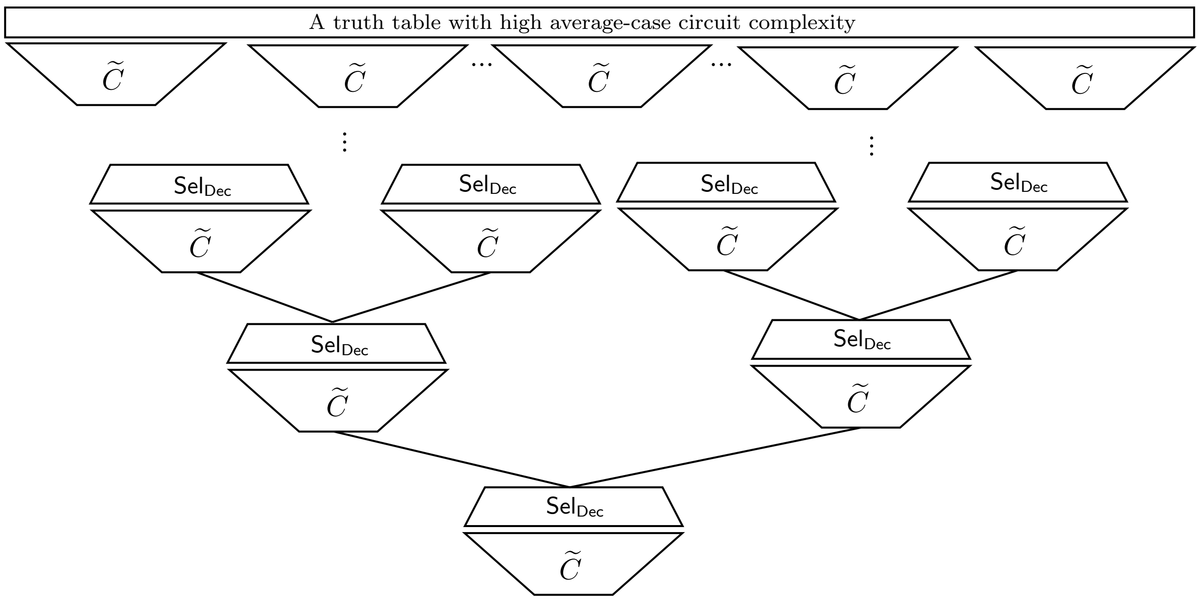

Figure 4: Modified GGM-tree Construction $\overline{\mathsf{GGM}}_{\ell,q,k}[C,(\mathsf{Enc},\mathsf{Dec})]$ for Remote-Point. We will demonstrate that a truth table with high average-case circuit complexity is a remote point for $\overline{\mathsf{GGM}}_{\ell,q,k}[C,(\mathsf{Enc},\mathsf{Dec})]$, and can be used to find a remote point for C.

Algorithm 2: $\mathsf{Je\check{r}\acute{a}bek\text{-}Korten}^{\mathsf{Avg}}(C,(\mathsf{Enc},\mathsf{Dec}),f)$: Modified Jeřábek-Korten reduction for Remote-Point

Input: A circuit $C:\{0,1\}^n\to\{0,1\}^m$, a $(q(n+\log L),m,1/2-c,L)$-code $(\mathsf{Enc},\mathsf{Dec})$, the height k, and a string $f\in\{0,1\}^\ell$ which is $(1/2-c)$-far from $\mathrm{Range}(\overline{\mathsf{GGM}}_{\ell,q,k}[C,(\mathsf{Enc},\mathsf{Dec})])$.

Output: A string $y\in\{0,1\}^m$ which is $\left(\frac{1}{2}-c-\left(\frac{1}{2}+c\right)\cdot\frac{\log L}{m}\right)$-far from $\mathrm{Range}(C)$.

```
for j ← 0 to q^{k-1} − 1 do
    Use the NP oracle to find the lexicographically smallest v_{k-1,j} such that
     δ(C̃(v_{k-1,j}), f_{[(m+log L)·j,(m+log L)·(j+1))}) ≤ 1/2 − c ;
    if v_{k-1,j} does not exist then
        return f_{[(m+log L)·j,(m+log L)·j+m)} ;
    end
end
for i ← k − 2 to 0 do
    for j ← 0 to q^i − 1 do
        Use the NP oracle to find the lexicographically smallest v_{i,j} such that
         Sel_Dec(C̃(v_{i,j})) = v_{i+1,qj} ∘ ··· ∘ v_{i+1,qj+q−1} ;
        if v_{i,j} does not exist then
            return Enc(v_{i+1,qj} ∘ ··· ∘ v_{i+1,qj+q−1}) ;
        end
    end
end
return ⊥ ;
```

The correctness of Algorithm 2 is based on the following lemma, which tells us the relation between an Avoid instance of $\mathsf{Sel}_{\mathsf{Dec}} \circ \widetilde{C}$ and a remote-point of C.

Lemma 4.1 (Modified from [CHLR23]). *For a circuit $C : \{0,1\}^n \to \{0,1\}^m$, a $(q(n+\log L), m, 1/2-c, L)$-code $(\mathsf{Enc}, \mathsf{Dec})$ and a padded circuit $\widetilde{C}$ shown in Step 1, let $C' : \{0,1\}^{n+\log L} \to \{0,1\}^{q(n+\log L)}$ be the circuit defined as $C'(x) = \mathsf{Sel}_{\mathsf{Dec}}(\widetilde{C}(x))$, $\forall x \in \{0,1\}^{n+\log L}$. If a string $y \in \{0,1\}^{q(n+\log L)}$ does not belong to $\mathrm{Range}(C')$, then $\mathsf{Enc}(y)$ is a $(1/2-c)$-far from $\mathrm{Range}(C)$.*

Proof. Assume that $\exists x \in \{0,1\}^n$ s.t. $\delta(\mathsf{Enc}(y), C(x)) \le 1/2 - c$. Then by the definition of list-decodable code, we have $y \in \mathsf{Dec}(C(x))$, and hence $\exists z \in \{0,1\}^{\log L}$ s.t. $y = \mathsf{Sel}_{\mathsf{Dec}}(C(x), z) = \mathsf{Sel}_{\mathsf{Dec}}(\widetilde{C}(x,z)) = C'(x,z)$, which leads to a contradiction. $\square$

Lemma 4.2. *Given that $f \in \{0,1\}^\ell$ is $(1/2-c)$-far from $\mathrm{Range}(\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})])$, Algorithm 2 guarantees to find a $y \in \{0,1\}^m$ such that $y \ne \perp$ and y is $\left(\frac{1}{2} - c - \left(\frac{1}{2}+c\right)\cdot\frac{\log L}{m}\right)$-far from $\mathrm{Range}(C)$. Additionally, if Enc runs in $\mathrm{poly}(n,q)$ time, Algorithm 2 only needs $\mathrm{poly}(\ell) = \mathrm{poly}(n, q^{k-1})$ calls for $\mathbf{NP}$ oracle.*

Proof. The running time is trivial. For correctness, assume that $y = \perp$, i.e. the algorithm returns an empty string. Then each $v_{i,j}$ exists, and therefore $\mathsf{Sel}_{\mathsf{Dec}}(\widetilde{C}(v_{i,j})) = v_{i+1,qj} \circ \cdots \circ v_{i+1,qj+q-1}$ for any $i < k-1$ and $\delta(\widetilde{C}(v_{k-1,j}), f_{[(m+\log L)\cdot j,(m+\log L)\cdot(j+1))}) \le 1/2 - c(n)$.

This tells us that $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})](v_{0,0}) = \widetilde{C}(v_{k-1,0}) \circ \cdots \circ \widetilde{C}(v_{k-1,q^{k-1}-1})$ is $(1/2-c)$-close to f, which contradicts the fact that f is $(1/2-c)$-far from $\mathrm{Range}(\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})])$. This shows $y \ne \perp$.

Next we prove that y is $\left(\frac{1}{2} - c - \left(\frac{1}{2}+c\right)\cdot\frac{\log L}{m}\right)$-far from $\mathrm{Range}(C)$. If $v_{i,j}$ $(i < k-1)$ does not exist, by Lemma 4.1, $\mathsf{Enc}(v_{i+1,qj} \circ \cdots \circ v_{i+1,qj+q-1})$ is $(1/2-c)$-far, and of course $\left(\frac{1}{2} - c - \left(\frac{1}{2}+c\right)\cdot\frac{\log L}{m}\right)$-far, from $\mathrm{Range}(C)$.

If $v_{k-1,j}$ does not exist, then $f_{[(m+\log L)\cdot j,(m+\log L)\cdot(j+1))}$ is $(1/2-c)$-far from $\mathrm{Range}(\widetilde{C})$. Here the algorithm deletes that last $\log L$ bits, only returns $f_{[(m+\log L)\cdot j,(m+\log L)\cdot j+m)}$. Note that $\widetilde{C}(x,z)$ is defined as $C(x) \circ z$, and hence the distance between $f_{[(m+\log L)\cdot j,(m+\log L)\cdot j+m)}$ and $\mathrm{Range}(C)$ is at least:

$$\frac{1}{m}\cdot\left(\left(\frac{1}{2}-c\right)\cdot(m+\log L) - \log L\right) = \frac{1}{2} - c - \left(\frac{1}{2}+c\right)\cdot\frac{\log L}{m},$$

as desired. $\square$

Applying the modified GGM-tree construction and Algorithm 2, we can derive an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point under the assumption of average-case circuit lower bounds.

Theorem 4.3. *Suppose the function $c : \mathbb{N} \to \mathbb{N}$ is good and monotonically decreasing, and satisfies $c(O(\log n)) \ge 1/n$. For any constant $0 < \tau < 1$, if $\mathbf{E}^{\mathbf{NP}} \not\subset \mathsf{Avg}_{c(n)}\text{-}\mathsf{SIZE}[2^{\tau n}]$, then there exists a constant $\gamma > 0$ and an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for Remote-Point$\left[n, n^{6+\gamma}, c\big(O_{\tau,\gamma}(\log n)\big)\right]$.*

Proof. We use the modified GGM-tree construction and apply the corresponding reduction from Algorithm 2. Suppose now we consider the circuit $C : \{0,1\}^n \to \{0,1\}^m$ where $m = n^{6+\gamma}$. Choose a $(2(n+O(\log n)), m, 1/2 - 1/n, \mathrm{poly}(n))$-code $(\mathsf{Enc}, \mathsf{Dec})$ from Theorem 2.11. Since Dec runs in $\mathrm{poly}(n)$ time, $\mathsf{Sel}_{\mathsf{Dec}}$ can also be implemented by $\mathrm{poly}(n)$-size circuits. Let $s_{\widetilde{C}}$ and s_{Sel} be circuit complexities of $\widetilde{C}$ and $\mathsf{Sel}_{\mathsf{Dec}}$.

Set the height $k = 2\left\lceil\frac{1}{\tau}\cdot\log(s_{\widetilde{C}} + s_{\mathsf{Sel}})\right\rceil + 1$. Then the output length of $\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})]$ is $\ell = m2^{k-1} \ge (s_{\widetilde{C}} + s_{\mathsf{Sel}})^{2/\tau}$.

Let $L \in \mathbf{E}^{\mathbf{NP}}$ be the language which cannot be $(1/2 + c(n))$-approximated by any circuit with size $2^{\tau n}$. And $y \in \{0,1\}^\ell$ is the corresponding truth table with length ℓ. Note that y cannot be $(1/2 + c(\log \ell))$-approximated by any circuit with size $\ell^\tau \geq (s_{\widetilde{C}} + s_{\mathsf{Sel}})^2$.

We claim that y is $(1/2 - c(\log \ell))$-far from $\mathrm{Range}(\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})])$. To show this, observe that if a string $y' \in \{0,1\}^\ell$ is $(1/2 - c(\log \ell))$-close to $\mathrm{Range}(\overline{\mathsf{GGM}}_{\ell,q,k}[C, (\mathsf{Enc}, \mathsf{Dec})])$, then it can be $(1/2 + c(\log \ell))$-approximated by the succinct circuit similar to Figure 3 with size $O((s_{\widetilde{C}} + s_{\mathsf{Sel}}) \cdot k)$ (here the only difference is that we replace C's with $\mathsf{Sel}_{\mathsf{Dec}} \circ \widetilde{C}$'s). The gap between $\Omega((s_{\widetilde{C}} + s_{\mathsf{Sel}})^2)$ and $O((s_{\widetilde{C}} + s_{\mathsf{Sel}}) \cdot k)$ proves our claim.

Then note that $s_{\widetilde{C}} + s_{\mathsf{Sel}} = \mathrm{poly}(n)$. Thus $\log \ell = k - 1 + \log(m + O(\log n)) = O_{\tau,\gamma}(\log n)$. By choosing γ appropriately, we have $c(\log \ell) \geq 1/n$. Therefore applying Algorithm 2 on input y gives us a string $z \in \{0,1\}^m$ which is $\left(\frac{1}{2} - c(\log \ell) - \left(\frac{1}{2} + c(\log \ell)\right) \cdot \frac{O(\log n)}{m}\right)$-far, and of course $\big(1/2 - c(O_{\tau,\gamma}(\log n))\big)$-far, from $\mathrm{Range}(C)$. The algorithm described above can be easily implemented in $\mathbf{FP}^{\mathbf{NP}}$ given that Enc runs in $\mathrm{poly}(n)$ time. $\square$

Corollary 4.4. *Let $c : \mathbb{N} \to \mathbb{N}$ be a good and monotonically decreasing function which satisfies $c(O(\log n)) \geq 1/n$. Then $\mathbf{E}^{\mathbf{NP}} \not\subset$ i.o.-$\mathsf{Avg}_{c(n)}$-$\mathsf{SIZE}[2^{o(n)}]$ if and only if there exists a constant $\gamma > 0$ and an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for* Remote-Point$\left[n, n^{6+\gamma}, c\big(O_\gamma(\log n)\big)\right]$.

Proof. One direction is proved in Theorem 4.3. For the converse direction, let $c'(n) = c(O_\gamma(\log n))$ and $f(n) = n^{6+\gamma}$. Then $c'(f^{-1}(2^n)) = c(O_\gamma(\log 2^{n/(6+\gamma)})) = c(n)$ with suitable γ, and $f^{-1}(2^n)/\log f^{-1}(2^n) = 2^{\Omega_\gamma(n)}$. Thus the proof is complete just by Theorem 2.8. $\square$

Similarly, we can also get the connection between the $\mathbf{FQP}^{\mathbf{NP}}$ algorithm for Remote-Point and the average-case circuit lower bound against $\mathbf{EXP}^{\mathbf{NP}}$:

Theorem 4.5. *Let $c : \mathbb{N} \to \mathbb{N}$ be a good and monotonically decreasing function which satisfies $c(O(\log n)) \geq 1/n$. Then $\mathbf{EXP}^{\mathbf{NP}} \not\subset$ i.o.-$\mathsf{Avg}_{c(n)}$-$\mathsf{SIZE}[2^{o(n)}]$ if and only if there exists a constant $\gamma > 0$ and an $\mathbf{FQP}^{\mathbf{NP}}$ algorithm for* Remote-Point$\left[n, n^{6+\gamma}, c\big(O_\gamma(\log n)\big)\right]$.

Discussion. Note that based on our approach, any future improvements on circuit complexity, rate or error correction radius of the list-decoder will improve the equivalence result. Especially if there exists a decoder that can be implemented in some restricted circuit classes (e.g. $\mathsf{AC}^0, \mathsf{ACC}^0, \mathsf{TC}^0$), we can potentially establish equivalence for restricted circuit classes.

5 A Family of $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ Time Algorithms for NC^0-Avoid$[n, n^{1+\varepsilon}]$

5.1 Algorithm

In this subsection, we present an improved subexponential-time algorithm for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$.

Our algorithm operates by identifying a small Hall-violating subcircuit and solving the corresponding restricted Avoid instance. Specifically, we reduce the original instance to a smaller one of the form NC^0_k-Avoid$[n' - 1, n']$ where $n' = n^{1-\frac{\varepsilon}{k-1}+o(1)}$, and then enumerate over the image of this small subcircuit. This yields a total runtime of $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$.

We begin by viewing the NC^0_k circuit $C \colon \{0,1\}^n \to \{0,1\}^m$ as a k-left-regular bipartite graph between m output bits (left side) and n input bits (right side).

The key combinatorial fact we use is the following:

Lemma 5.1 (Lower bound from [RTS00]). *Let $G = (L = [M], R = [N], E)$ be a D-left-regular bipartite graph that is a $(K, \frac{N-K+1}{N})$-disperser. Then*

$$D = \bar{D} \geq \frac{\log(M/(K-1))}{\log(1/(1-\frac{N-K+1}{N}))+1} = \frac{\log(M/(K-1))}{\log(N/(K-1))+1}.$$

In our $\mathsf{NC}^0_k[n, n^{1+\varepsilon}]$ setting, rearranging the above, we obtain:

$$(K-1)^{k-1} \leq \frac{(2n)^k}{n^{1+\varepsilon}} \quad \Rightarrow \quad K \leq K_0 := 2^{\frac{k}{k-1}} \cdot n^{1-\frac{\varepsilon}{k-1}} + 1.$$

Consequently, when $K > K_0$, any $\mathsf{NC}^0_k[n, n^{1+\varepsilon}]$ circuit cannot have the topological structure of a k-left-regular $(K, \frac{n-K+1}{n})$-disperser, and hence must contain a subset of K outputs with at most $(1 - \frac{n-K+1}{n}) \cdot n = K - 1$ distinct neighbors, violating Hall's condition. Brute-force search can find such subset and define a subcircuit C' of size K, which fails to be surjective. This leads to the following algorithm:

Algorithm 3: Improved Subexponential-Time Algorithm for NC^0-Avoid$[n, n^{1+\varepsilon}]$

Input: An NC^0_k circuit $C\colon \{0,1\}^n \to \{0,1\}^m$, with $m \geq n^{1+\varepsilon}$ for some constant $\varepsilon > 0$.
Output: A set of strings $y_1, \ldots, y_\ell \in \{0,1\}^m$ such that $y_i \notin \mathrm{Range}(C)$.

1. Search over all subsets $S \subseteq [m]$ of size $K = \lfloor K_0 \rfloor + 1$, and find one with $|\Gamma(S)| < |S|$ (guaranteed by Lemma 5.1). Let C' be the induced subcircuit.
2. Enumerate all $2^{|\Gamma(S)|}$ inputs and identify strings $y'_1, \ldots, y'_\ell \notin \mathrm{Range}(C')$.
3. For each y'_i, construct $y_i \in \{0,1\}^m$ that agrees with y'_i on S and is $*$ (representing arbitrary value) elsewhere.
4. Output $y_1, \ldots, y_\ell$.

Theorem 5.2. *Algorithm 3 runs in time $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$.*

Proof. In Step 1, we enumerate all $\binom{m}{K} \leq \left(\frac{em}{K}\right)^K = 2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ subsets. Step 2 performs $2^{n^{1-\frac{\varepsilon}{k-1}}}$ enumerations. Step 3 is linear in output size. Thus the total runtime is $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$. ☐

Corollary 5.3. *There exists a family of $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time algorithms for NC^0_k-Avoid$[n, n^{1+\varepsilon}]$. In addition, the algorithm can output a succinct representation of $\geq 1/2$ fractions of strings outside the range.*

Proof. For the subcircuit, there are more than half of the strings outside the range of the subcircuit. And since we allow $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time, we can output all such strings, and any fixing of the rest of the bits is a valid string not in the range of the larger circuit. This implies that Algorithm 3 can output a succinct representation of $\geq 1/2$ fractions of strings outside the range. ☐

Remark 5.4. *When $\varepsilon = (k-1)\left(1 - \frac{\log\log n + O(1)}{\log n}\right)$, i.e., $m = n^k / \log^{k-1} n$, the algorithm runs in polynomial time.*

Tighter Bounds via Improved Disperser Assumption. If the disperser bound of Lemma 5.1 can be improved to:

$$(K-1)^{D-2} \leq \frac{(2N)^{D-1}}{M}, \tag{5.1}$$

then setting $K = 2^{\frac{D-1}{D-2}} \cdot N^{1-\frac{\varepsilon}{D-2}} + 1$ again yields $M \leq N^{1+\varepsilon}$ (matching exactly the existence bound from Theorem 2.15), and the same algorithm applies.

Based on the above observation, we make the following assumption:

Assumption 5.5. *Let $G = (L = [M], R = [N], E)$ be a D-left-regular bipartite graph that is also a $(K, \frac{N-K+1}{N})$ disperser, then it holds that*

$$D - 1 = \bar{D} - 1 \geq \frac{\log(M/(K-1))}{\log(1/(1-\frac{N-K+1}{N})) + 1} = \frac{\log(M/(K-1))}{\log(N/(K-1)) + 1}.$$

Theorem 5.6. *Suppose Assumption 5.5 is true, there exists a family of $2^{n^{1-\frac{\varepsilon}{k-2}+o(1)}}$ time algorithms for NC_k^0-*Avoid*$[n, n^{1+\varepsilon}]$. In particular, the family of algorithms runs in polynomial time for NC_k^0-*Avoid*$[n, n^{k-1}/\log^{k-2} n]$. In addition, they output a succinct representation of $\geq 1/2$ fractions of strings outside the range.*

5.2 Implications for Local PRGs

Our subexponential-time Avoid algorithm has implications for PRG constructions in NC^0 (i.e., local PRG).

Theorem 5.7. *Suppose there exists a $\mathscr{C}$-*Avoid*$[n, m(n)]$ algorithm that, in time 2^{n^γ}, outputs a succinct representation of a non-negligible fraction of non-image strings. Then no $\mathscr{C}[n, m(n)]$-based pseudorandom generator is 2^{n^γ}-secure.*

Proof. Let $C \in \mathscr{C}$ be a PRG with output length $m(n)$. Let adversary $\mathcal{A}$ accept an input y iff $y \notin \mathrm{Range}(C)$. Since the Avoid algorithm runs in time 2^{n^γ}, this gives a distinguisher that accepts at least $2^{m(n)-1}$ non-image strings but accepts none from the PRG, violating the security of the PRG. □

Corollary 5.8. *Assuming the existence of $2^{m(n)^\beta}$-secure local PRGs in $\mathsf{NC}_k^0[n, m(n)]$, there cannot exist an algorithm for NC_k^0-*Avoid*$[n, m(n)]$ that runs in time 2^{n^γ} for any $\gamma < \beta$ and identifies a $\mathsf{negl}(n)$ fraction of non-image strings.*

6 A Faster Local Greedy Algorithm for NC_k^0-Avoid$[n, n+1]$

6.1 Algorithm

We present a simple greedy algorithm for NC_k^0-Avoid$[n, n+1]$ that runs in time

$$O\left(n \cdot 2^{\frac{(k-2)n}{k-1}}\right).$$

When $k = 2$, this yields a linear-time algorithm, matching the result of [GLW22].

Before presenting the algorithm, we need the following definition of *preimage space*.

Definition 6.1 (Preimage Space)**.** *Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a Boolean circuit. For a partial assignment $\tilde{y} \in \{0,1,*\}^m$ to the output bits, the* preimage space of $\tilde{y}$ *is defined as*

$$\mathsf{Preimage}(\tilde{y}) := \{x \in \{0,1\}^n \mid C(x) \text{ is consistent with } \tilde{y}\}.$$

In other words, $\mathsf{Preimage}(\tilde{y})$ *is the set of all valid input assignments x such that the output $C(x)$ agrees with the fixed bits of $\tilde{y}$.*

Algorithm 4: Improved Greedy Algorithm for NC^0_k-AVOID$[n, n+1]$

Input: An NC^0_k circuit $C\colon \{0,1\}^n \to \{0,1\}^m$, where $m \geq n+1$.
Output: A string $y \in \{0,1\}^m$, such that $y \notin \mathrm{Range}(C)$.

```
initially all bits of y are unassigned.
while there exists an unassigned output bit y_i and the preimage space is non-empty do
    Assign a value to y_i such that the preimage space Preimage(y) is reduced by at least a
     factor of 1/2;
end
if all output bits are assigned then
    return the assigned output string;
else
    Assign arbitrary values to unassigned bits and output the resulting string;
end
```

6.2 Analysis

Theorem 6.2. *Algorithm 4 solves NC^0_k-AVOID$[n, m]$ for $m \geq n+1$ in time $O\left(n \cdot 2^{\frac{(k-2)n}{k-1}}\right)$.*

Proof. We first argue that the algorithm always finds a valid non-image string. After fixing at most $(n+1)$ output bits, the preimage space is reduced to the empty set, so the output string obtained is guaranteed to lie outside the image of the circuit.

To analyze the running time of Algorithm 4, we model the input-output behavior of C via random variables:

- Let $X = (X_1, \ldots, X_n)$ denote i.i.d. uniform input random variables.
- and $Y = (Y_1, \ldots, Y_m)$ denote the output random variables.

Each output bit Y_i is computed as:

$$Y_i = f_i\left(X_{\sigma_i(1)}, \ldots, X_{\sigma_i(k)}\right),$$

where $f_i\colon \{0,1\}^k \to \{0,1\}$ is a Boolean function and $\sigma_i\colon [k] \to [n]$ indicates the input positions read. For each $i \in [m]$, we say that $X_{\sigma_i(1)}, \ldots, X_{\sigma_i(k)}$ are the input variables that are adjacent to Y_i.

Let $\tilde{Y}$ represent a subsequence of Y and $\tilde{y}$ the fixing of $\tilde{Y}$. Then there exists a (partial) assignment $y' \in \{\tilde{y} \circ 0 \circ \underbrace{*\cdots*}_{(m-1-|\tilde{Y}|)\text{ of } *'s}, \tilde{y} \circ 1 \circ \underbrace{*\cdots*}_{(m-1-|\tilde{Y}|)\text{ of } *'s}\}$ s.t. $y' \notin \mathrm{Range}(C)$ iff $H_\infty(X \mid \tilde{Y} = \tilde{y}) = 0$ and $m - |\tilde{Y}| > 0$. Thus, the algorithm can be viewed as a process that reduces the min-entropy of X by successively fixing bits of Y.

In the beginning of the process, X_i's are i.i.d. uniform random variables. Upon fixing an output bit, the input random variables adjacent to it become correlated. In general, fixing output bits iteratively will change the dependence.

Definition 6.3 (Connected Output Bits)**.** *We say that two fixed output variables Y_a and Y_b, where $a, b \in [m], a \neq b$ are connected if the sets of input variables adjacent to them intersect, i.e.,*

$$\left(\bigcup_{i \in [k]} X_{\sigma_a(i)} \right) \cap \left(\bigcup_{i \in [k]} X_{\sigma_b(i)} \right) \neq \varnothing.$$

Definition 6.4 (Preimage Subspace)**.** *Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a Boolean circuit, and let $\tilde{Y} \subseteq \{Y_1, \dots, Y_m\}$ be a subset of the output variables. Denote by $\tilde{X} \subseteq \{X_1, \dots, X_n\}$ the set of input variables adjacent to $\tilde{Y}$. For a fixing $\tilde{y} \in \{0,1\}^{|\tilde{Y}|}$ of $\tilde{Y}$, the* preimage subspace of $(\tilde{Y}, \tilde{y})$ *is defined as*

$$\mathsf{PreimageSub}(\tilde{Y} = \tilde{y}) := \{\, \tilde{x} \in \{0,1\}^{|\tilde{X}|} \mid \exists\, x \in \{0,1\}^n \text{ such that } C(x) \text{ is consistent with } \tilde{y} \text{ and } x|_{\tilde{X}} = \tilde{x} \,\}.$$

In words, $\mathsf{PreimageSub}(\tilde{Y} = \tilde{y})$ *is the set of all assignments of the input variables adjacent to $\tilde{Y}$ that are consistent with fixing $\tilde{Y}$ to $\tilde{y}$.*

Remark 6.5 (Preimage Space vs. Preimage Subspace)**.** *We emphasize the distinction between the two notions introduced above. The* preimage space $\mathsf{Preimage}(\tilde{y}) \subseteq \{0,1\}^n$ *refers to the set of all full input assignments consistent with a partial output fixing $\tilde{y}$. In contrast, the* preimage subspace $\mathsf{PreimageSub}(\tilde{Y} = \tilde{y}) \subseteq \{0,1\}^{|\tilde{X}|}$ *only records the assignments to those input variables $\tilde{X}$ that are adjacent to $\tilde{Y}$. Thus, the subspace is a projection of the full preimage space onto the relevant coordinates, ignoring the inputs that play no role in determining $\tilde{Y}$. This distinction will be crucial in our running-time analysis later.*

Lemma 6.6 (Decomposition of Preimage Subspace)**.** *Given a subset of t output random variables denoted by $\tilde{Y}^{(t)}$ that are fixed to $\tilde{y}^{(t)}$. Then* $\mathsf{PreimageSub}(\tilde{Y}^{(t)} = \tilde{y}^{(t)})$ *can be decomposed into disjoint subspaces $T_1, T_2, \dots, T_s$ where $s \geq 1$.*

Proof. Classify $\tilde{Y}^{(t)}$ into s connected components according to Definition 6.3. Denote the connected components by $\tilde{Y}_1^{(t)}, \dots, \tilde{Y}_s^{(t)}$. For $i \in [s]$, let $\tilde{y}_i^{(t)}$ be the fixing of $\tilde{Y}_i^{(t)}$. Each component $\tilde{Y}_i^{(t)}$ corresponds to a disjoint preimage subspace $T_i := \mathsf{PreimageSub}(\tilde{Y}_i^{(t)} = \tilde{y}_i^{(t)})$. Furthermore, it holds that $\{T_1, \dots, T_s\}$ are determined by mutually independent input random variables, since by Definition 6.3, their adjacent output variables are not connected. $\square$

Hence, upon fixing r output bits, the input variables adjacent to the r output bits can be viewed as clusters of mutually independent random variables. The clusters are independent of each other, but internally correlated. Each cluster corresponds to a connected component of the fixed output bits.

Preimage subspace of a subset of output bits corresponding to a fixing: Let r be the number of fixed output bits. Denote the fixed output bits as $Y_{t_1}, Y_{t_2}, \cdots, Y_{t_r}$ and their fixing as $y_{t_1}, y_{t_2}, \cdots, y_{t_r}$. The preimage subspace of $Y_{t_1}, Y_{t_2}, \cdots, Y_{t_r}$ corresponding to $y_{t_1}, y_{t_2}, \cdots, y_{t_r}$ is the set of valid fixings of the random variables

$$\bigcup_{i \in [r]} \{X_{\sigma_{t_i}(1)}, \cdots, X_{\sigma_{t_i}(k)}\}.$$

Let $\tilde{Y}^{(t)}$ denotes the concatenation of the t output random variables that have been fixed, $\tilde{y}^{(t)}$ be its fixing, and $\tilde{X}^{(t)}$ denotes the concatenation of the input random variables adjacent to $\tilde{Y}^{(t)}$. Let us define the following useful notion of *traversed decision space*.

Definition 6.7 (traversed decision space $\mathcal{T}(t)$)**.** *After $\tilde{Y}^{(t)}$ is fixed to $\tilde{y}^{(t)}$, according to Lemma 6.6, let the decomposition of the corresponding preimage subspace of the $\tilde{Y}^{(t)}$ be $T_1, \ldots, T_s$, each over disjoint subsets of $\tilde{X}^{(t)}$. Define:*

$$\mathcal{T}(t) := \{T_1, \ldots, T_s\}, \quad w(\mathcal{T}(t)) := 2^{k-s} \cdot \prod_{i \in [s]} |T_i|$$

where $|T_i|$ denotes the cardinality, or the number of possible assignments of the input random variables associated with the space T_i.

Intuitively, the function $w(\mathcal{T}(t))$ upper bounds the effective search space we need to track after t output bits are fixed — the preimage space of the next output bit to be fixed intersects at most k subsets in $\mathcal{T}(t)$ while any set in $\mathcal{T}(t)$ has size ≥ 2.

The following claim says that the search space is non-trivially bounded.

Claim 6.8. *For all t, we have $w(\mathcal{T}(t)) \leq 2^{(k-2)t+k}$.*

Proof. We proceed inductively.

Base Case: When $t = 1$, it holds that there are only one input space corresponding to the fixed output bit, and it holds that

$$w(\mathcal{T}(1)) = 2^{k-1} \cdot |T_1| \leq 2^{k-2+k}.$$

For the inductive case, assume that $w(\mathcal{T}(h)) \leq 2^{(k-2)h+k}$ for $h = (t-1)$, we prove in the following that $w(\mathcal{T}(t)) \leq 2^{(k-2)t+k}$ also holds. There are two cases to consider:

Case 1: The inputs adjacent to the new output bit are disjoint from the inputs of all previously traversed output bits. In this case, the decision of which boolean value to assign to the current output bit only depends on a constant-sized space of 2^k values. Let s_t denote $|\mathcal{T}(t)|$. Then the new subspace added to $\mathcal{T}(t)$ is T_{s_t}. It holds that

$$w(\mathcal{T}(t)) \leq 2^{-1} \cdot 2^{(k-2)(t-1)+k} \cdot |T_{s_t}| = 2^{-1} \cdot 2^{(k-2)(t-1)k1} \cdot 2^{k-1} \leq 2^{(k-2)t+k}$$

by induction.

Case 2: Suppose $\ell \in (0, k]$ of the inputs adjacent to the new output bit intersects with $1 \leq r$ subspaces $T_{t_1}, \ldots, T_{t_r}$ in $\mathcal{T}(t-1)$. Since $T_{t_1}, \ldots, T_{t_r}$ are disjoint, it holds that $r \leq \ell$. Then, fixing this output bit merges the intersected subspaces. Moreover, since only $(k-\ell)$ new random variables are introduced into $\tilde{X}^{(t)}$, the growth from the product of the sizes of the spaces $T_{t_1}, \ldots, T_{t_r}$ to the size of the merged subspace is bounded by a factor of at most $2^{k-\ell}$. On the other hand, our choice to fix the output bit always reduces the preimage size by at least half. Hence, the net increase is bounded by:

$$w(\mathcal{T}(t)) \leq 2^{-r+1} \cdot w(\mathcal{T}(t-1)) \cdot 2^{k-\ell-1} \leq 2^{(k-2)t+k}$$

by induction.

□

Remark 6.9. *The following is true*

$$H_\infty(\tilde{X}^{(t)} \mid \tilde{Y}^{(t)}) = \log\left(\prod_{i \in [s]} |T_i|\right) = \log\left(|\mathsf{PreimageSub}(\tilde{Y}^{(t)} = \tilde{y}^{(t)})|\right).$$

Proof. Initially, every assignment of $\tilde{X}^{(t)}$ has equal probability density $1/2^{|\tilde{X}^{(t)}|}$, since $\tilde{X}^{(t)}$ forms the minimal sufficient set of random variables determining $\tilde{Y}^{(t)}$.

Because the partial assignments in each T_i for $i \in [s]$ are independent, the product $\prod_{i\in[s]} |T_i|$ exactly counts the number of valid assignments consistent with fixing $\tilde{Y}^{(t)} = \tilde{y}^{(t)}$.

Moreover, each valid assignment has the same probability density, and this property remains invariant as we fix the bits of $\tilde{Y}^{(t)}$ sequentially. □

By a similar proof, we have

Remark 6.10. *The following is true*

$$|\mathsf{Preimage}(\tilde{y})| = 2^{H_\infty(X|\tilde{Y})}.$$

Finally, we use a "meet-in-the-middle" argument to analyze the running time. By Claim 6.8, the size of the traversed decision space grows at most as

$$|\mathcal{T}(t)| \leq 2^{(k-2)t}.$$

On the other hand, fixing t output bits shrinks the preimage space — equivalently, the quantity $2^{H_\infty(X|\tilde{Y}^{(t)})}$ — to size at most 2^{n-t}. Thus, when determining the t-th output bit, the algorithm needs to examine a subspace of size at most

$$\min\{2^{(k-2)t+k},\, 2^{n-t}\} \;\leq\; 2^{\frac{k+(k-2)n}{k-1}}.$$

Since the algorithm performs at most $(n+1)$ steps and each step inspects a space of size at most $2^{\frac{k+(k-2)n}{k-1}}$, the overall running time is

$$O\left(n \cdot 2^{\frac{(k-2)n}{k-1}}\right).$$

□

6.3 Lower Bound

The following result shows that Algorithm 4 has exponential worst-case runtime, giving evidence of the intrinsic hardness of NC^0_k-Avoid$[n, O(n)]$.

Theorem 6.11. *Algorithm 4 runs in exponential time in the worst case for* NC^0_k*-Avoid*$[n, O(n)]$.

Proof. By Theorem 2.14, a random $\mathsf{NC}^0_k[n, O(n)]$ circuit is an $(\Omega(n), k-1-\varepsilon)$-bipartite expander with probability at least $1/2$, where ε is constant arbitrarily close to 0. Fix such a circuit. For an arbitrary subset of output bits of size $\Omega(n)$, the induced subgraph on inputs and outputs is nearly a tree, with only $O(1)$ cycles. This is the worst-case scenario in the above case analysis of Algorithm 4:

- there will be only a major single subspace in $\mathcal{T}(t)$;
- there are almost no cycles in the subcircuit, there is no means to additively reduce the size of $\mathcal{T}(t)$.

These essentially imply that the upper bound on $w(\mathcal{T}(t))$ could be tight if at each step of the fixing we reduce the input space by roughly $1/2$. This happens in the following instances.

Assuming each predicate f_i is a random Boolean function (say, implemented by resilient functions), then when we iteratively fix each output bit, no matter which bit value we assign to the next unfixed bit, with high probability, the traversed decision space increases by a factor of 2^{k-2}. Thus, the number of configurations to track grows exponentially, and the traversed decision space size necessarily reaches $2^{\Omega(n)}$.

From the output string's perspective, this means that every $\Omega(n)$-bit projection of the image is nearly uniform. Hence, no partial assignment over $\Omega(n)$ output bits can efficiently help identify a non-image string, and the algorithm explores exponentially many paths. □

Remark 6.12. *Note that no unconditional exponential-time lower bound can be shown for any NC^0-Avoid algorithms in the constant-stretch regime. Indeed, since $\mathsf{NC}^0\text{-Avoid} \in \mathbf{F\Sigma}_2$ [Kor21], it follows that if $\mathbf{P} = \mathbf{NP}$, then $\mathsf{NC}^0\text{-Avoid} \in \mathbf{FP}$. Thus, an unconditional exponential-time lower bound would imply $\mathbf{NP} \neq \mathbf{P}$.*

7 Conclusion and Open Problems

Open Problem 1.

- **(Hardness)** Improve the stretch for the hardness of NC^0-Avoid problem: by [CL24], we know that $\mathsf{NC}^1\text{-Avoid}[n, n+1] \notin \mathbf{SearchNP}$. Under randomized encoding techniques [RSW22], this also implies that $\mathsf{NC}^0_4\text{-Avoid}[n, n+1] \notin \mathbf{SearchNP}$. Can we prove that under plausible assumptions $\mathsf{NC}^0\text{-Avoid}[n, O(n)] \notin \mathbf{SearchNP}$, or even for some small constant ε, $\mathsf{NC}^0_k\text{-Avoid}[n, n^{1+\varepsilon}] \notin \mathbf{SearchNP}$ when k is large.

- **(Algorithms)** In the work, we show that there is a $2^{n^{1-\frac{\varepsilon}{k-1}+o(1)}}$ time algorithm for $\mathsf{NC}^0_k\text{-Avoid}[n, n^{1+\varepsilon}]$. Does there exist a $2^{n^{o(1)}}$ time algorithm for $\mathsf{NC}^0_k\text{-Avoid}[n, n^{1+\varepsilon}]$ for some $\varepsilon > 0$? If so, then assuming ETH (Exponential Time Hypothesis) [IPZ98, IP01], $\mathsf{NC}^0_k\text{-Avoid}[n, n^{1+\varepsilon}] \in \mathbf{SearchNP}$.

Open Problem 2. In this work, we only prove equivalence results for polynomial stretch. Can we extend such equivalence to quasipolynomial stretch? Ideally, we would be able to prove the following conjecture.

Conjecture 1. *$\exists \delta$ s.t., $\mathbf{E}^{\mathbf{NP}}$ requires 2^{n^δ} size ACC^0 circuit complexity if and only if there is an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathsf{AC}^0\text{-Avoid}[n, \mathrm{qpoly}(n)]$, where each output bit is computed by a $\mathrm{qpoly}(n)$ size ACC^0 circuit.*

Assuming Conjecture 1 is true and leveraging on existing ACC^0 circuit lower bound against $\mathbf{E}^{\mathbf{NP}}$ [Wil14, CLW20], the reduction directly yields an $\mathbf{FP}^{\mathbf{NP}}$ algorithm for $\mathsf{ACC}^0\text{-Avoid}[n, \mathrm{qpoly}(n)]$ where each output bit is computed by a $\mathrm{qpoly}(n)$-size ACC^0 circuit.

We remark that the technique in this paper seems to fall short of achieving this, as to condense a hard function of large quasi-polynomial stretch using Jeřábek-Korten's reduction, one seems to need the depth of the tree to be super-constant.

Open Problem 3. Recall that [Jeř04, Kor21, CHR24] proved the following equivalence result.

$$\textsc{Avoid} \in \mathbf{FP}^{\mathbf{NP}} \iff \mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^{o(n)}] \iff \mathbf{E}^{\mathbf{NP}} \not\subset i.o.\text{-}\mathsf{SIZE}[2^n/n].$$

The second equivalence is a hardness amplification result.

1. Is there such a similar amplification result for restricted circuit classes? Given Theorem 1.6 and that AC^0-Avoid algorithm for smaller stretch implies stronger lower bounds according to Theorem 2.8, the answer could be negative.

2. Is there such an average-case to average-case hardness amplification pheonomemon, possibly by proving reduction between different instances of Remote-Point? It is unclear how to generalize the $\mathbf{FP}^{\mathbf{NP}}$ reduction of Avoid from any polynomial stretch to minimal stretch to Remote-Point.

Acknowledgements

We thank the anonymous ITCS 2026 reviewers for their thoughtful comments, the ITCS 2025 reviewers for their helpful feedback on a preliminary version of this work, and the FOCS 2025 reviewers for identifying bugs in the original proofs of the equivalence related to the Remote-Point problem and the equivalence involving exponential-size circuits for NC^1.

We would also like to thank Hanlin Ren for helpful discussions on range avoidance and Kuan Cheng for discussions on decoding in AC^0. We are grateful to Erfan Khaniki for pointing out a historical inaccuracy in an earlier version of this manuscript — the arguments underlying what we had referred to as "Korten's reduction" already appeared in Jeřábek's earlier work, as carefully credited in [Kor21]. Accordingly, referring to it solely as "Korten's reduction" is not historically accurate.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, USA, 1st edition, 2009. 11

[BHPT24] Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular pcps. *SIAM Journal on Computing*, 53(2):480–523, 2024. doi:10.1137/22M1495597. 37

[Bus87] S. R. Buss. The boolean formula value problem is in alogtime. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 123–131, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.28409. 12

[CH85] Stephen A. Cook and H. James Hoover. A depth-universal circuit. *SIAM Journal on Computing*, 14(4):833–839, 1985. doi:10.1137/0214058. 12, 35

[Che24] Lijie Chen. Nondeterministic quasi-polynomial time is average-case hard for ACC circuits. *SIAM Journal on Computing*, 0(0):FOCS19–332–FOCS19–397, 2024. doi:10.1137/20M1321231. 1

[CHLR23] Yeyuan Chen, Yizhi Huang, Jiatu Li, and Hanlin Ren. Range avoidance, remote point, and hard partial truth table via satisfying-pairs algorithms. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, page 1058–1066, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3564246.3585147. 1, 2, 6, 8, 20, 22

[CHR24] Lijie Chen, Shuichi Hirahara, and Hanlin Ren. Symmetric exponential time requires near-maximum circuit size. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 1990–1999, New York, NY, USA, 2024. Association for Computing Machinery. 2, 3, 15, 31

[CL23] Eshan Chattopadhyay and Jyun-Jie Liao. Hardness against linear branching programs and more. In *Proceedings of the Conference on Proceedings of the 38th Computational Complexity Conference*, CCC '23, Dagstuhl, DEU, 2023. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. 1

[CL24] Yilei Chen and Jiatu Li. Hardness of range avoidance and remote point for restricted circuits via cryptography. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 620–629, New York, NY, USA, 2024. Association for Computing Machinery. 1, 2, 30, 37

[CLW20] Lijie Chen, Xin Lyu, and R. Ryan Williams. Almost-Everywhere Circuit Lower Bounds from Non-Trivial Derandomization . In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1–12, Los Alamitos, CA, USA, November 2020. IEEE Computer Society. doi:10.1109/FOCS46700.2020.00009. 1, 20, 30

[CR22] Lijie Chen and Hanlin Ren. Strong average-case circuit lower bounds from nontrivial derandomization. *SIAM Journal on Computing*, 51(3):STOC20–115–STOC20–173, 2022. doi:10.1137/20M1364886. 1

[CW89] A. Cohen and A. Wigderson. Dispersers, deterministic amplification, and weak random sources. In *30th Annual Symposium on Foundations of Computer Science*, pages 14–19, 1989. doi:10.1109/SFCS.1989.63449. 14

[Ebe84] W. Eberly. Very fast parallel matrix and polynomial arithmetic. In *25th Annual Symposium onFoundations of Computer Science, 1984.*, pages 21–30, 1984. 40

[Erd47] Paul Erdös. Some remarks on the theory of graphs. *Bulletin of the American Mathematical Society*, 53:292–294, 1947. URL: https://api.semanticscholar.org/CorpusID:14215209. 1

[GGNS23] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. Range avoidance for constant depth circuits: Hardness and algorithms. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*, volume 275 of *LIPIcs*, pages 65:1–65:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 2, 5, 9, 11, 37

[GLW22] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. Range Avoidance for Low-Depth Circuits and Connections to Pseudorandomness. In Amit Chakrabarti and Chaitanya

Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*, volume 245 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:21, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 2, 5, 10, 25, 37

[GLY25] Venkatesan Guruswami, Xin Lyu, and Weiqiang Yuan. Cell-probe lower bounds via semi-random csp refutation: Simplified and the odd-locality case. *arXiv preprint arXiv:2507.22265*, 2025. 6, 10

[GR08] Venkatesan Guruswami and Atri Rudra. Soft decoding, dual bch codes, and better list-decodable e-biased codes. In *2008 23rd Annual IEEE Conference on Computational Complexity*, pages 163–174, 2008. doi:10.1109/CCC.2008.13. 13

[Gur09] Venkatesan Guruswami. List decoding of binary codes–a brief survey of some recent results. In *International Conference on Coding and Cryptology*, pages 97–106. Springer, 2009. 13

[ILW23] Rahul Ilango, Jiatu Li, and R. Ryan Williams. Indistinguishability obfuscation, range avoidance, and bounded arithmetic. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, page 1076–1089, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3564246.3585187. 2

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. 30

[IPZ98] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*, pages 653–662, 1998. doi:10.1109/SFCS.1998.743516. 30

[IW97] Russell Impagliazzo and Avi Wigderson. P = bpp if e requires exponential circuits: derandomizing the xor lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 220–229, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258590. 3

[Jeř04] Emil Jeřábek. Dual weak pigeonhole principle, boolean complexity, and derandomization. *Annals of Pure and Applied Logic*, 129(1):1–37, 2004. doi:10.1016/j.apal.2003.12.003. 3, 6, 7, 15, 31

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 60–73, New York, NY, USA, 2021. Association for Computing Machinery. 2, 15

[Joh62] S. Johnson. A new upper bound for error-correcting codes. *IRE Transactions on Information Theory*, 8(3):203–207, 1962. doi:10.1109/TIT.1962.1057714. 13

[KKMP21] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos 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. doi:10.4230/LIPIcs.ITCS.2021.44. 1

[Kor21] Oliver Korten. The hardest explicit construction. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 433–444, Los Alamitos, CA, USA, 2021. IEEE Computer Society. 1, 3, 5, 6, 7, 15, 16, 17, 30, 31, 39

[Kor25] Oliver Korten. Range avoidance and the complexity of explicit constructions. *The Computational Complexity Column by Michal Koucky, Bulletin of the European Association for Theoretical Computer Science*, 145:94–134, 2025. 4

[KPI25] O. Korten, T. Pitassi, and R. Impagliazzo. Stronger cell probe lower bounds via local prgs. In *Electron. Colloquium Comput. Complex.*, 2025. 2, 5, 6

[Kra92] Jan Krajíček. No counter-example interpretation and interactive computation. In Yiannis N. Moschovakis, editor, *Logic from Computer Science*, pages 287–293, New York, NY, 1992. Springer New York. 6

[KS25] N. Kuntewar and J. Sarma. Range avoidance in boolean circuits via turan-type bounds. In *Electron. Colloquium Comput. Complex.*, 2025. 2

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652. 3, 15

[Li23] Xin Li. Two Source Extractors for Asymptotically Optimal Entropy, and (Many) More . In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1271–1281, Los Alamitos, CA, USA, November 2023. IEEE Computer Society. doi:10.1109/FOCS57990.2023.00075. 39

[Li24] Zeyong Li. Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 2000–2007, New York, NY, USA, 2024. Association for Computing Machinery. 2, 6, 15

[LZ24] Xin Li and Yan Zhong. Explicit Directional Affine Extractors and Improved Hardness for Linear Branching Programs. In Rahul Santhanam, editor, *39th Computational Complexity Conference (CCC 2024)*, volume 300 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:14, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2024.10. 1

[MVW99] Peter Bro Miltersen, N. V. Vinodchandran, and Osamu Watanabe. Super-polynomial versus half-exponential circuit size in the exponential hierarchy. In *Proceedings of the 5th Annual International Conference on Computing and Combinatorics*, COCOON'99, page 210–220, Berlin, Heidelberg, 1999. Springer-Verlag. 2

[RSW22] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 640–650, 2022. doi:10.1109/FOCS54457.2022.00067. 1, 2, 3, 4, 5, 6, 8, 12, 13, 15, 30, 37, 38, 39, 40

[RTS00] Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM Journal on Discrete Mathematics*, 13(1):2–24, 2000. doi:10.1137/S0895480197329508. 9, 14, 24

[Sip86] M Sipser. Expanders, randomness, or time versus space. In *Proc. of the Conference on Structure in Complexity Theory*, page 325–329, Berlin, Heidelberg, 1986. Springer-Verlag. 14

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010. 14

[Val77] Leslie G Valiant. Graph-theoretic arguments in low-level complexity. In *International Symposium on Mathematical Foundations of Computer Science*, pages 162–176. Springer, 1977. 11

[VW23] Nikhil Vyas and Ryan Williams. On Oracles and Algorithmic Methods for Proving Lower Bounds. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 99:1–99:26, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.99. 14

[Wil14] Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1), January 2014. doi:10.1145/2559903. 30

A Universality Property of Low-Depth Circuits

The following theorem is implicit in [CH85].

Theorem A.1. *Any circuit class containing* AC^0 *has the universality property.*

Proof. We show that for any circuit $C \in \mathscr{C}_{n,s,d}$, where $\mathscr{C}$ is any circuit class containing AC^0, there exists a circuit $U_{n,s,d} \in \mathscr{C}$ that satisfies the three conditions of the universality property as defined in Definition 2.4.

We first need the following definition about the succinct encoding of C.

Definition A.2 (Encoding Format (Size $O(s \log s)$))**.** *Let the circuit* C *have* n *inputs,* m *gates,* s *wires (i.e., total* `fan-in` *across all gates is* s*), and depth* d*. We encode the circuit as a list of gates: Each gate descriptor includes:*

- **Gate type:** 2–3 *bits.*
- **List of fan-in wires:** *each wire is indexed by a* $\log s$*-bit value pointing to: either an input* x_i*, or another gate* g_j*.*

Note that the number of bits for the gate is:

$$O(1 + (\text{fan-in}) \cdot \log s)$$

Summing over all gates:

$$\sum_{\text{gates}} \text{fan-in}(g) = s \quad \Longrightarrow \quad \text{Total encoding size} = O(s \log s)$$

Then the following universal circuit construction applies.

General Universal Circuit Construction for $\mathscr{C} \supseteq \mathsf{AC}^0$. Consider the following set-up of parameters:

- Input size: n
- Wire bound: s
- Depth bound: d (can be constant or more, depending on the class)

Let C be any circuit in $\mathscr{C}$ with those bounds. We construct a *universal circuit* $U_{n,s,d}$ with the following properties:

Inputs:

- $x_1, \dots, x_n$: regular inputs
- $\langle C \rangle$: an encoding of a circuit C of size (wires) $\leq s$, depth $\leq d$, using a total of $O(s \log s)$ bits

Outputs:

- The output(s) of the simulated circuit $C(x)$

Universal Gate Module. For each gate in the simulated circuit, the universal circuit will include a *universal gate module* that:

- **Reads** the gate type from the encoding
- **Selects** the inputs using a list of $\log s$-bit selectors
- **Evaluates** the function ($\wedge$, $\vee$, $\neg$) as per the encoding

Input selection is done via a *selector tree* or *multiplexer* circuit using control bits from the encoding. This works in any class that can simulate a selector (e.g., AC^0).

Layered Construction (Depth-Universal Simulation). For a depth-d circuit C, simulate it layer-by-layer:

- Build d layers in the universal circuit
- Each layer contains $O(s)$ universal gate modules
- Layer i reads inputs from layer $i - 1$ or from the original inputs

This preserves depth:

- If $\mathscr{C}$ has constant depth, depth remains constant
- If $\mathscr{C}$ allows polylog-depth, so does $U_{n,s,d}$

Final Construction: Universal Circuit $U_{n,s,d}$. Let $\mathscr{C}$ be any circuit class containing AC^0, and let s and d be polynomially bounded functions of n.

Then we can construct a uniform family of universal circuits $\{U_{n,s,d}\}$ such that:

- Each $U_{n,s,d}$ has:
  - n regular inputs
  - $O(s \log s)$ encoding inputs
  - $O(s)$ auxiliary gates
  - Depth $O(d)$
- For any circuit $C \in \mathscr{C}$ with n inputs, $\leq s$ wires, and depth $\leq d$, and for any input $x \in \{0,1\}^n$, we have:

$$U_{n,s,d}(x, \langle C \rangle) = C(x)$$

This universal circuit simulates *any circuit from* $\mathscr{C}$ with specified resource bounds, given only its *succinct encoding* and input. □

B Reductions Between Avoid Instances via Direct-Sum

In this section, we present a reduction between instances of $\mathscr{C}$-Avoid, focusing on how to relate instances with varying input/output lengths.

We present a direct-sum-type reduction that improves upon prior reductions in the literature.

Theorem B.1. *For any constant $\delta \in (0,1)$ and any circuit class $\mathscr{C}$, it holds that*

$$\mathscr{C}\text{-Avoid}[n, n+1] \leq_{\mathbf{FP^{NP}}} \mathscr{C}\text{-Avoid}[n, n+n^{\delta}].$$

Specializing to $\mathscr{C} = \mathsf{NC}^0_k$, this reduction yields several consequences when combined with results from [RSW22, GLW22, GGNS23].

For instance, [GGNS23] showed that explicitly constructing rigid matrices sufficient for Valiant's program reduces to NC^0_3-Avoid$[n, n+n^{2/3}]$. Moreover, improving the current $\mathbf{FP^{NP}}$ constructions of rigid matrices [BHPT24] would follow from an $\mathbf{FP^{NP}}$ algorithm for NC^0_3-Avoid$[n, n+n^{12/17-\varepsilon}]$ for any constant $\varepsilon > 0$.

By Theorem B.1, we obtain that even solving NC^0_3-Avoid$[n, n+n^{\delta}]$ for any constant $\delta \in (0,1)$ is already sufficient to yield such constructions — though this suggests that doing so is likely as hard as solving the hardest case which has the minimum stretch NC^0_3-Avoid$[n, n+1]$, a stretch regime believed to lie beyond **SearchNP** [CL24].[14]

This reduction also applies to other explicit construction problems reducible to small-stretch NC^0_k-Avoid, including:

- constructing binary linear codes approaching the Gilbert–Varshamov bound,
- list-decodable codes achieving list-decoding capacity,
- optimal Ramsey graphs.[15]

[14] Precisely speaking, [CL24] only shows that it is likely that $\mathsf{NC}^0_4[n, n+1]$-Avoid $\notin$ **SearchNP**.

[15] While we are not aware of a formal reduction for Ramsey graphs in the literature, we provide one in Appendix D.

Hence, this result is both a positive and negative message: on the one hand, it shows the potential power of solving small-stretch AVOID instances; on the other hand, it aligns with the growing evidence that these instances are unlikely to be in **SearchNP**.

In the following, we present the proof of Theorem B.1.

Proof of Theorem B.1. Construct $s = n^{d/(d+1)}$ copies of $\mathcal{C} \in \mathscr{C}$ of input size $n^{1/(d+1)}$, each with stretch $n^{1/(d+1)} + 1$. Concatenating them yields a circuit $\mathcal{C}'$ with input size n and output size $n + n^{d/(d+1)}$. Given $y \notin \text{Range}(\mathcal{C}')$, we can partition y into s equal-sized blocks and use an **NP**-oracle to find a block not in Range($\mathcal{C}$) in time $O(s)$. $\square$

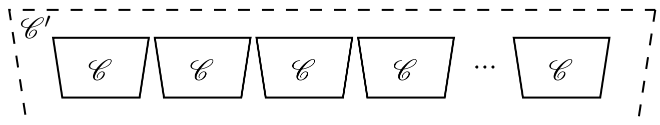

Figure 5: Concatenating small instances (circuits) with small stretch to a larger instance (circuit) with larger stretch.

C Missing Proofs

C.1 Proof of Theorem 2.8

We restate and prove Theorem 2.8 here, which is a version of the implication of $\mathscr{C}$-AVOID algorithms to circuit lower bounds based on *universality property* of the circuit classes from [RSW22], with tightened parameters.

Theorem C.1 (Refinement of Theorem 5.2 from [RSW22])**.** *Let $\mathscr{C}$ be any circuit class that has the universality property, and $f : \mathbb{N} \to \mathbb{N}$ be a monotone function that is good. Suppose there is an* $\mathbf{FP^{NP}}$ *(resp.* $\mathbf{FP}$*,* $\mathbf{FQP^{NP}}$*) algorithm for $\mathscr{C}$-REMOTE-POINT$[N, f(N), c(N)]$, where each output gate has $\mathscr{C}$ circuit complexity* poly(N)*. Then for some constant $\varepsilon > 0$,* $\mathbf{E^{NP}}$ *(resp.* $\mathbf{E}$*,* $\mathbf{EXP^{NP}}$*) cannot be $(1/2 + c(f^{-1}(2^n)))$-approximated by $\mathscr{C}$ circuits of size $\frac{\varepsilon f^{-1}(2^n)}{\log f^{-1}(2^n)}$.*

Proof. Consider the truth table mapping:

$$\mathsf{TT}_{\mathscr{C}} \colon \{0,1\}^N \to \{0,1\}^{2^n},$$

which maps the encoding $\langle C \rangle$ of a single-output $\mathscr{C}$ circuit of size $s = s(n)$ to its truth table. By the universality of $\mathscr{C}$, there exists a constant c such that $N = O(s \log s)$. In particular,

$$N \le f^{-1}(2^n) \cdot \left(1 - \frac{\log\log f^{-1}(2^n)}{\log f^{-1}(2^n)}\right) < f^{-1}(2^n),$$

for sufficiently large n.

Thus, the output length 2^n satisfies:

$$2^n > f(N).$$

Moreover, each output bit of $\mathsf{TT}_{\mathscr{C}}$ can be computed by a $\mathscr{C}$ circuit of size poly(N), since evaluating C on any input is efficient by assumption.

Applying the $\mathbf{FP^{NP}}$ algorithm for $\mathscr{C}$-Avoid$[N, f(N)]$, we can find a string $y \notin \mathrm{Range}(\mathsf{TT}_{\mathscr{C}})$. This string represents the truth table of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ that cannot be computed by any $\mathscr{C}$ circuit of size s. Since the Avoid algorithm runs in $\mathbf{FP^{NP}}$, the function f is in $\mathbf{FE^{NP}}$.

By the definition of $\mathscr{C}$-Remote-Point$[N, f(N), c(N)]$, the output of the algorithm on the instance C, which we call y, has relative hamming distance $\geq 1/2 - c(N)$ from Range(C). Then it holds that Range(C) and y agrees on $\leq 1/2 + c(f^{-1}(2^n))$ fraction of inputs. □

C.2 Proof of Theorem 1.5

We reproduce Theorem 1.5 here for convince:

Theorem C.2. *For any constant $\delta \in (0,1)$, an $\mathbf{FP^{NP}}$ algorithm for NC^0_4-Avoid$[n, n+n^\delta]$ implies that $\mathbf{E^{NP}}$ requires $\Omega(2^n/n)$-size formulas.*

Proof. By Theorem B.1, an $\mathbf{FP^{NP}}$ algorithm for NC^0_4-Avoid$[n, n+n^\delta]$ implies an $\mathbf{FP^{NP}}$ algorithm for NC^0_4-Avoid$[n, n+1]$. By instantiating Theorem C.1 with formulas, we have $\mathsf{Formula}$-Avoid$[n, n+1] \iff \in \mathbf{FP^{NP}} \implies \mathbf{E^{NP}} \not\subseteq \mathsf{Formula}[o(2^n/n)]$. By [RSW22], we have NC^0_4-Avoid$[n, n+1] \in \mathbf{FP} \implies \mathsf{Formula}$-Avoid$[n, n+1] \in \mathbf{FP}$. Hence, it holds that NC^0_4-Avoid$[n, n+1] \in \mathbf{FP^{NP}} \implies \mathbf{E^{NP}} \not\subseteq \mathsf{Formula}[o(2^n/n)]$ (tighted version of [RSW22, Theorem 5.8]). Combining the above we have, for any constant $1 > \delta > 0$, it holds that NC^0_4-Avoid$[n, n+n^\delta] \in \mathbf{FP^{NP}} \implies \mathbf{E^{NP}} \not\subset \mathsf{Formula}[o(2^n/n)]$. □

D Reducing Explicit Construction of Optimal Ramsey Graphs to NC^0_4-Avoid

The current state-of-the-art explicit construction of a $(\log^{O(1)} n)$-Ramsey graph is due to [Li23]. It is well-known that an explicit construction of a two-source extractor with parameters $(\log n + 2\log(1/\varepsilon(n)) + 3, \varepsilon(n))$ and constant error $\varepsilon(n) = O(1)$ would imply an explicit $O(\log n)$-Ramsey graph.

In this section, we show that constructing such two-source extractors can be reduced in polynomial time to the problem of finding strings outside the range of circuits in the class NC^0_4-Avoid. Our approach closely follows the strategy of [Kor21], who constructed circuits for Avoid instances.

Theorem D.1. *Let $\varepsilon(n)$ be any efficiently computable function satisfying $1/n^c < \varepsilon(n) < 1/2$ for some constant $c > 0$ and sufficiently large n. Then, the problem of explicitly constructing a $(\log n + 2\log(1/\varepsilon(n)) + 3, \varepsilon(n))$-two-source extractor reduces in polynomial time to NC^0_4-Avoid.*

Proof. The high-level idea is to encode a partial truth table of a candidate extractor on "bad" sources, i.e., sources on which the extractor fails to produce an ε-biased output. We then build a circuit that takes this partial truth table as input and computes the coefficients of a polynomial that interpolates exactly the points in the bad source. Any string outside the image of this circuit corresponds to a set of coefficients whose polynomial disagrees with every such bad source, thereby certifying the extractor as valid.

Consider the function $f : \{0,1\}^n \to \{0,1\}^n$ defined as:

$$f(x) = \sum_{i=1}^{2^{2k}} \alpha_i x^{i-1},$$

and define $g(x) = f(x) \bmod 2$, where arithmetic is over a suitable extension field.

The input to the circuit consists of:

1. The two sources X, Y, each of size 2^k, where each element is an n-bit string. These require $2 \cdot 2^k \cdot n = 2^{k+1}n$ bits.

2. A single bit $b \in \{0, 1\}$ indicating the biased output value.

3. The coefficients β_i for encoding the outputs on bad sources, which require $2^{2k}(2n - 1)$ bits.

4. A string $S \in \{0,1\}^{2^{2k}}$ of Hamming weight $(1/2 - \varepsilon) \cdot 2^{2k}$, specifying the support of the bad outputs. This can be encoded using at most $2^{2k}(1 - \varepsilon^2) + \log(2^{2k})$ bits (via standard entropy bounds).

The total number of *input bits* is:

$$2^{k+1}n + 1 + 2^{2k}(2n - 1) + 2^{2k}(1 - \varepsilon^2) + 2k.$$

The number of *output bits* is:

$$2^{2k} \cdot n,$$

corresponding to the full truth table of $f(x)$.

By choosing parameters such that:

$$2^{2k}\varepsilon^2 - 2k - 1 - 2^{k+1}n > 0,$$

we ensure that the number of inputs is strictly less than the number of outputs, making the construction amenable to the AVOID framework.

Computing the coefficients α_i from the evaluations of $f(x)$ can be done via polynomial interpolation, specifically by inverting a Vandermonde matrix. This procedure is known to be in NC^1 [Ebe84]. Finally, by applying the known reduction from NC^1-AVOID to NC^0_4-AVOID given in [RSW22], we conclude that explicitly constructing optimal two-source extractors (and thus optimal Ramsey graphs) reduces to NC^0_4-AVOID. $\square$

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il