Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 125 (2021)

The Exact Complexity of Pseudorandom Functions and the Black-Box Natural Proof Barrier for Bootstrapping Results in Computational Complexity

Zhiyuan Fan* Jiatu Li† Tianqi Yang‡

Institute for Interdisciplinary Information Sciences
Tsinghua University, Beijing, China

June 12, 2022

Abstract

Investigating the computational resources we need for cryptography is an essential task of both theoretical and practical interests. This paper provides answers to this problem on pseudorandom functions (PRFs). We resolve the exact complexity of PRFs by proving tight upper and lower bounds for various circuit models.

- PRFs can be constructed in $2n + o(n)$ size general circuits assuming the existence of polynomial-size PRFs, simplifying and improving the $O(n)$ upper bound by Ishai, Kushilevitz, Ostrovsky, and Sahai (STOC 2008). Moreover, if PRFs exist in NC^1, we can further guarantee the depth of our construction to be $(1+\varepsilon)\log n$. We show that our construction is almost optimal by giving an unconditional $2n - O(1)$ lower bound.
- PRFs can be constructed in $\mathsf{AC}^0[2]$ circuits of $o(n)$ gates and $2n + o(n)$ wires assuming the existence of polynomial-size $\mathsf{AC}^0[2]$ PRFs. We show the optimality of our construction with a $2n + \Omega(\sqrt{n})$ wire complexity lower bound.
- PRFs can be constructed with wire complexity $n^{1+O(1.61^{-d})}$ in depth-d TC^0 circuits assuming the existence of polynomial-size TC^0 PRFs. We also present an $n^{1+\Omega(c^{-d})}$ wire complexity lower bound against depth-d TC^0 circuits for some $c > 1.61$.

As a byproduct, we prove unconditional tight upper and lower bounds for "almost" universal hash functions that are of independent interest.

Following the natural proof barrier of Razborov and Rudich (J. Comput. Syst. Sci. 1997), we observe that our exact complexity results are closely related to the "bootstrapping phenomena" in circuit complexity (such as hardness magnification and quantified derandomization). We introduce the black-box natural proof barrier and show that a large range of techniques for bootstrapping results cannot be combined with "black-box" lower bound proofs to obtain a breakthrough.

*fan-zy19@mails.tsinghua.edu.cn
†lijt19@mails.tsinghua.edu.cn
‡yangtq19@mails.tsinghua.edu.cn

ISSN 1433-8092

Contents

1 Introduction

A pseudorandom function is a fundamental primitive in cryptography, yielding, among others, simple solutions for the main problems in private key cryptography (see [BR17] for an excellent survey on this topic). The celebrated result of Goldreich, Goldwasser, and Micali [GGM84] revealed the power of such a notion by showing its equivalence to pseudorandom generators and, therefore, one-way function by later works [GL89; HILL99]. From both practical and theoretical perspectives, it is natural to study the amount of computational recourses we need to construct pseudorandom functions.

We use the usual definition of pseudorandom functions in this paper. The syntax of a pseudorandom function is defined as a collection F_n of n-input single-output Boolean functions and a sampling distribution $\mathcal{D}_n$ supported over F_n. A pseudorandom function with security $s(n)$ is a family of such collection, such that for any probabilistic $s(n)$ time adversary $\mathcal{A}$ and sufficiently large n,

$$\left| \Pr_{f \leftarrow F_n} [\mathcal{A}^f(1^n) = 1] - \Pr_{g:\{0,1\}^n \to \{0,1\}} [\mathcal{A}^g(1^n) = 1] \right| \leq 1/s(n),$$

where g is uniformly random from all n-input single-output Boolean functions. By default, we focus on PRFs with polynomial security (i.e., secure against any probabilistic polynomial time adversary). The circuit complexity of a PRF is naturally defined as the maximum complexity of functions in F_n.

Prior to our work, low-complexity PRFs have been extensively studied in cryptography. Ishai, Kushilevitz, Ostrovsky, and Sahai [IKOS08] proved that PRFs could be constructed in linear-size general circuits assuming PRFs exist. For low-depth circuits, candidates in NC^1, TC^0, or even $\mathsf{AC}^0[2]$ of polynomial-size [NRR00; NR04; BPR12; Vio15] follow from standard assumptions like Decisional Diffie-Hellman or Ring Learning-with-Error. More efficient candidates are known in TC^0 based on heuristics [MV15].

Our main contribution is to work out the *exact* complexity of pseudorandom functions in various circuit classes. We present extremely efficient constructions of pseudorandom functions in general B_2 circuits, NC^1 circuits, TC^0 circuits, and $\mathsf{AC}^0[2]$ circuits. In the meantime, we show that these constructions are almost optimal by proving matching circuit lower bounds (see Table 1). All of our lower bounds are *unconditional*, and all of our upper bounds hold under the *weakest possible assumptions*, namely the mere existence of (polynomial-size) PRFs in corresponding circuit classes. As a technical byproduct, we also obtain tight unconditional upper and lower bounds for "almost" universal hash functions in these circuit classes (see Table 2), which is of independent interest.

Apart from direct cryptographic applications, the study of low-complexity pseudorandom functions also brings insights to the study of circuit complexity. The seminal work of Razborov and Rudich [RR97] showed that a large range of circuit lower bound techniques captured by the concept of *natural proofs* are not capable of proving super-polynomial lower bounds since they can be used to break exponentially strong PRFs that are believed to exist. By constructing low-complexity PRF candidates with exponential security (see, e.g., [MV15]), we can find evidence that natural proofs may not be able to prove even weaker lower bounds, say $\mathsf{NP} \not\subseteq \mathsf{SIZE}[n \log^{10} n]$.

Following the intuition from the natural proof barrier, we observe that our result on the exact complexity of PRFs is closely related to a sequence of "bootstrapping results" in circuit complexity, which showed that mild lower bounds or derandomization results could imply major breakthroughs (see, e.g., [AK10; OS18; Tel18; OPS19; CT19; MMW19; CJW19; Che+20; CJW20]). We introduce a fine-grained variant of the natural proof barrier, which we call *black-box natural proof*

	B_2 circuits	NC^1 circuits	TC^0 circuits	$\mathsf{AC}^0[2]$ circuits
PRF upper bounds	$2n + o(n)$ size	$2n + o(n)$ size $(1+\varepsilon)\log n$ depth	$n^{1+O(\phi^{-d})}$ wires for depth d	$2n + o(n)$ wires $o(n)$ gates
PRF lower bounds	$2n - O(1)$ size		$n^{1+\Omega(c^{-d})}$ wires for depth d	$2n + \Omega(\sqrt{n})$ wires

Table 1: Summary of our results on PRFs. In this table, n refers to the input length, $\phi = \frac{1+\sqrt{5}}{2} \approx 1.618$ and $c > \phi$ is a constant. The upper bounds hold assuming the existence of polynomial-size PRFs in corresponding models (in fact, we develop *unconditional* methods to transform polynomial-size PRFs into low-complexity PRFs). See Theorem 1.1 for upper bounds and Theorem 1.2 for lower bounds.

	General circuits	$\mathsf{AC}^0[2]$ circuits	TC^0 circuits
Hash upper bounds	$2n + o(n)$ size $(1 + o(1))\log n$ depth	$2n + o(n)$ wires $o(n)$ gates	$n^{1+O(\phi^{-d})}$ wires for depth d
Techniques	High-girth graph + stacking	High-girth graph + sparse ECC	Sparse ECC
Hash lower bounds	$2n - 2m$ size	$2n + \Omega(\sqrt{n}) - 2m$ wires	$n^{1+\Omega(c^{-d})}$ wires for depth d
Techniques	Wire-counting based distinguisher		Random restriction

Table 2: Summary of our results on hash functions. In this table, n refers to the input length and m refers to the output length of the hash functions. See Theorem 1.3 for upper bounds and Theorem 1.4 for lower bounds.

barrier, for these bootstrapping results. Intuitively, it shows that several widely used bootstrapping techniques involving error-correcting codes or hash functions (see, e.g., [OPS19; Tel18; CT19; CJW19; CJW20; Che+20]) cannot be combined with "black-box" proofs of circuit lower bounds to establish a breakthrough. In particular, it explains the gap between the known lower bounds and the bootstrapping premises in recent sharp threshold results [OPS19; CT19; CJW20].

1.1 Our results

We mainly consider the complexity of PRFs in the following four circuit models: general circuits of fan-in-two gates (also named B_2 circuits), logarithmic-depth circuits (named NC^1 circuits)[1], constant-depth unbounded-fan-in circuits over basis $\{\neg, \wedge, \vee, \oplus\}$ (named $\mathsf{AC}^0[2]$ circuits) and constant-depth unbounded-fan-in circuits over linear threshold functions (named TC^0 circuits). For almost universal hash functions, our upper bounds can be implemented in a very restricted class $\mathsf{CC}^0[2]$, which consists of all constant depth unbounded-fan-in circuits with only parity gates.

[1]Each gate of B_2 or NC^1 circuits we defined here is of fan-in-two and can compute an arbitrary fan-in-two Boolean function. Note that this is different from U_2 circuits (or de Morgan circuits) in which there is no exclusive OR gate or the negation of it.

PRF upper bounds. Our first results are efficient constructions of PRFs in B_2 circuits, NC^1 circuits, $\mathsf{AC}^0[2]$ circuits and TC^0 circuits under the weakest possible assumptions.

Theorem 1.1. PRF upper bounds (informal).

B_2 circuits. If PRF exists, then there exists a PRF computable by B_2 circuits of $2n + o(n)$ size (see Corollary 5.13).

NC^1 circuits. If PRF in NC^1 exists, then for any constant $\varepsilon > 0$, there exists a PRF computable by NC^1 circuits of $2n + o(n)$ size and $(1 + \varepsilon) \log n$ depth simultaneously (see Corollary 6.2).

$\mathsf{AC}^0[2]$ circuits. If PRF in depth-d $\mathsf{AC}^0[2]$ exists, then there exists a PRF computable by $\mathsf{AC}^0[2]$ circuits of depth $d + 3$ with $o(n)$ gates and $2n + o(n)$ wires (see Corollary 6.10).

TC^0 circuits. If PRF exists in depth-d_0 TC^0, then there exists an absolute constant c, such that for any $d \geq d_0 + 4$, there exists a PRF computable by depth-d TC^0 circuits with $n^{1+c\phi^{-d}}$ wires, where $\phi = \frac{1+\sqrt{5}}{2}$ (see Corollary 6.5).

Note that the proofs of these upper bounds are all constructive: we can explicitly give a polynomial-time algorithm that takes a polynomial-size PRF in the corresponding class as input and outputs a PRF with low complexity. So if the original PRFs are uniform (i.e., samplable by a polynomial-time algorithm), our newly constructed low-complexity PRFs are uniform as well.

These upper bounds are proved by a general complexity reduction framework known as *Levin's domain extension trick* (see, e.g. [BR17], or Section 4.5). It states that we can construct a PRF by compositing an almost universal hash function (that shrinks an n-bit input into an n^δ-bit hash value) and a PRF with input length n^δ. If δ is chosen small enough, the complexity of the original PRF can be absorbed into the $o(n)$ term. Therefore the complexity of our final PRFs merely depends on the complexity of the hash function. The PRF upper bounds then follow from unconditional constructions of efficient hash functions in B_2 circuits and low-depth circuit classes.

PRF lower bounds. We also prove tight circuit lower bounds of computing PRFs in these circuit models, which shows that our constructions of PRFs are almost optimal.

Theorem 1.2. PRF lower bounds (informal).

B_2 circuits. Computing PRFs in B_2 circuits requires size at least $2n - O(1)$ (see Theorem 7.5).

NC^1 circuits. Computing PRFs in NC^1 circuits requires depth at least $\log n$ (trivial).

$\mathsf{AC}^0[2]$ circuits. Computing PRFs in $\mathsf{AC}^0[2]$ circuits requires at least $2n + \Omega(\sqrt{n})$ wires (see Theorem 7.10).

TC^0 circuits. There is an absolute constant $c > 1$ such that for all $d \geq 1$, any depth-d TC^0 circuits computing a PRF should have at least $n^{1+\Omega(c^{-d})}$ wires, where $\Omega(\cdot)$ hides an absolute constant independent of d and n (see Theorem 8.1).

All these lower bounds are unconditional and follow completely combinatorial arguments, which do not require the PRFs to be uniform. Intuitively, we show that all small-size circuits have structural properties that make them "detectable" by polynomially many oracle queries. The

lower bounds for B_2 and $\mathsf{AC}^0[2]$ circuits are proved by a simple but tricky *wire-counting* technique. The lower bound for TC^0 circuits builds upon a random restriction technique that has already been used to prove size-depth trade-off lower bounds and derandomization against linear threshold circuits [CSS18; Tel18; CT19; HHTT21]. In fact, the constant c in our theorem tightly matches the corresponding constants in the previous line of works.

Unconditional exact complexity of almost universal hash functions. As a byproduct of our analysis, we can also get upper and lower bounds for a weaker variant of universal hash functions. These results are quantitatively similar to those for PRFs but are completely unconditional.

Recall that a hash function H_n is a collection of functions mapping n-bit strings to m-bit strings. It is called *universal* if for any inputs $x \neq y \in \{0,1\}^n$, we have $\Pr_{h \leftarrow H_n}[h(x) = h(y)] = 2^{-m}$. However, in many cases, we do not need the collision probability to be exactly 2^{-m}. This motivates the definition of *almost universal hash function* with collision probability $\varepsilon(n)$, where the requirement is loosened to $\Pr_{h \leftarrow H_n}[h(x) = h(y)] \leq \varepsilon(n)$. Without further clarification we take $\varepsilon(n) = \mathsf{negl}(n)$ (i.e., $\varepsilon(n)$ vanishes faster than the inverse of any polynomial).

Our upper bounds of hash functions for TC^0 follow directly from sampling over the extremely sparse error-correcting code in TC^0 [CT19], while those for NC^1 and $\mathsf{CC}^0[2]$ require a novel construction from high-girth graphs (see, e.g., [Cha03]). These unconditional hash constructions are exactly the main technical ingredients of our PRF upper bounds.

Theorem 1.3. Almost universal hash functions with output length $m = n^{0.1}$ can be constructed by general circuits of $2n + o(n)$ size and $(1 + o(1)) \log n$ depth simultaneously (see Lemma 6.1), $\mathsf{CC}^0[2]$ circuits with $o(n)$ gates and $2n + o(n)$ wires (see Lemma 6.9), or depth-d TC^0 circuits of $n^{1+O(\phi^{-d})}$ wires for any $d \geq 4$, where $\phi = \frac{1+\sqrt{5}}{2}$, and $O(\cdot)$ hides an absolute constant independent of d and n (see Corollary 6.4). ♢

On the other hand, unconditional lower bounds for PRFs can be adapted to almost universal hash functions[2]. By the general connection between almost universal hash functions and error-correcting codes (see Proposition 4.8), these lower bounds also hold for error-correcting codes.

Theorem 1.4. There exists a constant c such that the following holds. Any almost universal hash function with output length $m = n^{0.1}$ requires B_2 (or NC^1) circuits of size at least $2n - 2m$ (see Theorem 7.6), $\mathsf{AC}^0[2]$ circuits with wire complexity at least $2n + \Omega(\sqrt{n}) - 2m$ (see Theorem 7.11), or depth-d TC^0 circuits of wire complexity at least $n^{1+\Omega(c^{-d})}$ for any depth $d \geq 1$ (see Corollary 8.6). ♢

A note on the security parameter. Although we use polynomial security throughout this paper to make it clean and readable, it is helpful to mention the concrete security parameters of our constructions. Recall that our PRFs are the composition of two parts: an almost universal hash function of collision probability $\varepsilon(n)$ which shrinks an n bits input to $m = n^{\varepsilon}$ bits, and a PRF on m inputs with security $s(m)$. By examining the proof of Levin's trick (see Section 4.5), one can see that the resulting PRF has security roughly $\min\{\varepsilon(n)^{-\Omega(1)}, s(m)^{\Omega(1)}\}$.

- The hash functions for $2n + o(n)$ size NC^1 and $\mathsf{CC}^0[2]$ in Theorem 1.3 has $\exp(-\Omega(\frac{\log^2 n}{\mathsf{poly}(\log\log n)}))$ collision probability. Therefore, assuming the existence of $\exp(\Omega(\frac{\log^2 n}{\mathsf{poly}(\log\log n)}))$ secure PRFs

[2]One may notice that assuming the existence of corresponding PRFs, such lower bounds follow directly from Levin's trick and our PRF lower bounds. Here we make them unconditional by carefully adapting the proofs.

in B_2, NC^1, or $\mathsf{AC}^0[2]$ circuits, the resulting low complexity PRFs are also $\exp(\Omega(\frac{\log^2 n}{\mathsf{poly}(\log\log n)}))$ secure. We note that the existence of such a PRF in $\mathsf{AC}^0[2]$ indeed follows from sub-exponential DDH (see Section 6.1 for a brief survey).

- The TC^0 hash function in Theorem 1.3 using efficient ECCs (see Section 6.3) can achieve $\exp(-\Omega(n^\varepsilon))$ collision probability for output length $m = n^\varepsilon$. Assuming the existence of sub-exponentially secure TC^0 PRFs (following from sub-exponential DDH), our TC^0 PRF also has sub-exponential security.

Note that the collision probability of the TC^0 hash function is already close to optimal, so a major technical open problem is to improve the one in $\mathsf{CC}^0[2]$. In particular, it is interesting to construct an almost universal hash function in $2n$ size circuits with inverse sub-exponential collision probability, which would lead to constructions of sub-exponentially secure PRFs in $2n + o(n)$ size circuits. We note that the hash function in [IKOS08] can achieve such collision probability with circuit size $O(n)$, and we make partial progress by presenting a non-uniform construction of size $3n$ in Section 5.3.

1.2 A new barrier for "bootstrapping results" in circuit complexity

Recently, a sequence of works (see, e.g., [AK10; OS18; Tel18; OPS19; CT19; MMW19; CJW19; Che+20; CJW20]) revealed a mysterious phenomenon that a mild lower bound (e.g. $\mathsf{MCSP} \notin \mathsf{SIZE}[n^{1.01}]$) or a derandomization algorithm for weak circuit classes (e.g. for TC^0-$\mathsf{SIZE}[n^{1.01}]$) would lead to a major breakthrough in computational complexity (e.g. $\mathsf{NP} \nsubseteq \mathsf{P}_{/\mathsf{poly}}$, or derandomization for the entire TC^0). These bootstrapping results are formulated in different settings, such as hardness magnification (see, e.g., [Che+20]), quantified derandomization (see, e.g., [Tel21]), and explicit obstruction (see, e.g., [CJW20]). More interestingly, the bootstrapping frontiers in some works (see, e.g., [OPS19; CT19; CJW20]) are extremely sharp in the sense that slightly weaker variants of the bootstrapping premises can actually be resolved. For instance, Chen, Jin, and Williams [CJW20] proved an $n^{2-o(1)}$ lower bound against probabilistic U_2-formulas[3] for the Minimum Circuit Size Problem (MCSP) with certain parameters while improving this lower bound to $n^{2.01}$ would imply $\mathsf{NP} \nsubseteq \mathsf{Formula}[n^k]$ for all k.

As the bootstrapping phenomenon appears broadly in circuit complexity, understanding its actual power turns out to be important. Some researchers believe that it is not a plausible approach to resolve the desired open problems, and to consolidate this pessimistic view, barriers should be set to address the weakness of current techniques. Although the bootstrapping techniques may not be limited by the celebrated *natural proof barrier* [RR97; Che+20], specific barriers are known in two major settings of the bootstrapping results. Namely, Chen et al. [Che+20] introduced the locality barrier for hardness magnification, and Tell [Tel17; Tel21] introduced a black-box barrier for quantified derandomization.

To obtain breakthroughs from bootstrapping phenomena, we need to close the gap between the bootstrapping premises and the provable lower bounds with improved bootstrapping and/or lower bound techniques. Therefore a barrier applied to the bootstrapping results needs to show that the gap is *unavoidable*: the improvement of the bootstrapping side and the lower bound side contradict each other *if we only utilize known techniques*. More precisely, we should describe a

[3]We use U_2-formulas to represent formulas with all binary connectives except for XOR and its complement, which is also known as de Morgan formulas.

tractable side that characterizes the lower bound techniques and a *bootstrapping side* that characterizes the bootstrapping techniques, and then demonstrate how they refute the improvements of each other.

This work presents a simple and general barrier, which we call *black-box natural proof barrier*, that provides a novel perspective of the bootstrapping results. The barrier is inspired by both the natural proof barrier [RR97] and our investigation on the complexity of PRFs; specifically, the tractable side of our barrier is a refinement of the natural proof barrier, and the bootstrapping side of it relates to low-complexity constructions of pseudorandom functions. It applies to the following two kinds of techniques: bootstrapping techniques using low-complexity constructions of certain combinatorial primitives that could be used to transform a polynomial-size PRF into a low-complexity PRF, giving a PRF upper bound assuming PRFs exist; and lower bounds or derandomization algorithms leading to a "property tester" that could efficiently distinguish any small-size circuit from a truly random function and subsequently yield unconditional PRF lower bounds. We will show that these two kinds of techniques should not be able to be combined to establish a breakthrough, since it will lead to conflicting upper and lower bounds for PRFs and then disprove the existence of PRFs.

We now discuss the techniques captured by our barrier on the tractable side and the bootstrapping side respectively. More details are given in Section 3.

Tractable side. Our barrier applies to the lower bound or derandomization techniques that imply efficient algorithms "recognizing" the small-size circuits in a black-box manner, i.e., distinguishing the small-size circuits from truly random functions via oracle queries.[4] These techniques are said to be *black-box natural*. Our barrier addresses the limitation of black-box natural techniques: if we want to prove a black-box natural lower bound against $s_{\mathsf{lb}}(n)$-size circuits, the distinguisher as a byproduct of it will imply an unconditional $s_{\mathsf{lb}}(n)$-size PRF lower bound. This means that if we can construct PRFs computable by $s_{\mathsf{prf}}(n)$-size circuits from well-founded assumptions, we should not hope to prove better-than-$s_{\mathsf{prf}}(n)$ lower bounds with merely black-box techniques.

Black-box natural techniques have been used as the main component to prove many low-depth circuit lower bounds [Hås86; Hås98; Tal14; CSS18] and (quantified) derandomization [GW14; IMZ19; CT19; HHTT21]. To the best of our knowledge, most of the tractable sides of sharp bootstrapping results (see, e.g., [CT19; CJW20]) essentially utilize black-box natural techniques. Still, there are many lower bound techniques in the literature not being black-box natural (at least in our eyes), such as gate elimination in proving B_2 circuit lower bounds (see, e.g., [DK11; FGHK16; LY21]), formula lower bounds for Andreev functions (see, e.g., [And87; Hås98; Tal14]), the polynomial method in proving $\mathsf{AC}^0[2]$ lower bounds (see, e.g., [Raz87; Smo87]).

Bootstrapping side. Our barrier applies to the bootstrapping techniques that essentially follow from error-correcting codes or almost universal hash functions, and whose bootstrapping thresholds[5] are determined by the complexity of these primitives. We observe that the attempt to improve the bootstrapping thresholds by constructing more efficient ECCs or hash functions cannot lead to a breakthrough if we only have a black-box natural lower bound to

[4] It is said to be *black-box* since the distinguisher has only the oracle access to the function and runs in $\mathsf{poly}(n)$ time, whereas the distinguisher in natural proof barrier can read the entire truth-table and runs in $2^{O(n)}$-time. This means that the tractable side of the black-box natural proof barrier is indeed a restricted version of the natural proof barrier.

[5] The *bootstrapping threshold* refers to the circuit size we need to work with to obtain a breakthrough. The bootstrapping result "an $n^{1.01}$-size lower bound for L will imply $\mathsf{NP} \not\subseteq \mathsf{P}_{/\mathsf{poly}}$", for instance, has bootstrapping threshold $n^{1.01}$.

combine with it. Assume that we can prove an $s_{\mathsf{lb}}(n)$ lower bound with black-box natural techniques. The key insight is that the ECCs or hash functions can also be used to construct PRFs with (roughly) the same complexity as long as polynomial-size PRFs exist.[6] As a result, when we prove a bootstrapping result with threshold $s_{\mathsf{th}}(n)$, we will also obtain a PRF computable in (roughly) $s_{\mathsf{th}}(n)$-size assuming PRFs exist. This means to obtain a breakthrough (i.e., to make $s_{\mathsf{lb}}(n) > s_{\mathsf{th}}(n)$) we will have to disprove the existence of polynomial-size PRFs (recall that an $s_{\mathsf{lb}}(n)$ black-box natural technique implies an $s_{\mathsf{lb}}(n)$ PRF lower bound).

The bootstrapping side of our barrier covers a large range of known bootstrapping techniques: the reduction-based [OS18; OPS19] and the kernelization-based [CJW19; CJW20] techniques for hardness magnification; and the error-reduction technique using seeded extractors [GW14; CT19] for quantified derandomization. We note that hardness magnification from tree-like structures of particular problems are not included (see, e.g., [AK10; MMW19]).

In summary, the black-box natural proof barrier captures the conflict between these two kinds of techniques: it is inevitable to disprove the existence of PRFs in order to get a breakthrough. In particular, this observation explains why "sharp frontiers" occur in recent bootstrapping results, such as the hardness magnification in [OPS19; CJW20] and the quantified derandomization in [CT19; CJW20] (see Section 3.5 for detailed discussion).

We borrow the following suggestion from the influential paper of Razborov and Rudich [RR97] as our advice to the readers.

> *"We do not conclude that researchers should give up on proving serious lower bounds. Quite the contrary, by classifying a large number of techniques that are unable to do the job we hope to focus research in a more fruitful direction."*

The bootstrapping phenomenon is worth further investigation, given the fact that many known lower bound techniques are not black-box natural. It remains appealing to adapt these techniques to the problems that admit the bootstrapping phenomena or establish stronger barriers extending to a broader class of lower bound techniques.

1.3 Related works

Linear-size PRFs in [IKOS08]. For B_2 circuits, Ishai, Kushilevitz, Ostrovsky, and Sahai [IKOS08] gave an $O(n)$ upper bound based on Levin's trick (see Section 4.5) and an efficient construction of pairwise-independent hash functions. We briefly compare our construction with that of [IKOS08].

- Our construction is simpler and more efficient. The multiplicative overhead of [IKOS08] is huge, making it harder to be implemented in practice. Our construction can be generalized to restricted circuit classes like TC^0 and $\mathsf{AC}^0[2]$ and can give more structural insights about the power of small-size circuits.

- The construction of [IKOS08] provides two features that have individual interests: it is a perfect pairwise-independent hash function (instead of being only almost universal), and there is a linear-size circuit computing the PRF with the key as its second input. Our constructions via almost universal hash functions do not support these features natively.

[6] This is done via the Levin's trick that is also used in our PRF upper bounds. Note that here PRF means PRF secure against any polynomial-time adversary, instead of $2^{\Omega(n)}$-secure PRF (called *pseudorandom function generator* in [RR97]) for natural proof barrier.

Low-complexity PRFs. Low complexity PRFs have been extensively studied, motivated by both cryptographic applications and the natural proof barrier [RR97].

General circuits. Since [IKOS08] proved the $O(n)$ upper bound with sub-exponential security, people have tried to construct exponentially secure candidates of small size. Miles and Viola [MV15] proposed an exponentially secure PRF candidate in quasi-linear size circuits based on the substitution-permutation network paradigm. Subsequently, Boneh, Ishai, Passelègue, Sahai, and Wu [BIPSW18] proposed the first exponentially secure candidate in $O(n)$ size based on heuristics. Constructing efficient PRFs with exponential security from well-founded assumptions remains to be an interesting and important open problem.

Low-depth circuits. Many PRF candidates with various security levels have been constructed in low-depth circuit classes such as NC^1, TC^0, and even $\mathsf{AC}^0[2]$. From well-founded assumptions such as the Decisional Diffie-Hellman, PRFs can be constructed in TC^0 [NR04; NRR00]. By strengthening the assumption to sub-exponentially secure, this construction can even be implemented in $\mathsf{AC}^0[2]$ with quasi-polynomial security [Vio15]. This quasi-polynomial security is known to be near-optimal by the natural proof in $\mathsf{AC}^0[2]$ [RR97]. In TC^0 with small wire complexity, Miles and Viola [MV15] presented a candidate of wire complexity $n^{1+O(1/d)}$ based on the substitution-permutation network paradigm. For weak PRFs (where the adversary is given a random input together with the function value), several efficient candidates are known (see, e.g. [ABGKR14; BIPSW18; Boy+21]). We refer interested readers to Table 1 of [Boy+21] for a comprehensive survey of known results.

It might be instructive to consider our results and these works as complements of each other. We focus on optimizing the circuit complexity as a function of the input length, while these works try to improve the security as a function of the key length.

Circuit lower bounds. Proving circuit lower bounds for particular functions can be unexpectedly hard. Although most people believe that $\mathsf{NP} \not\subseteq \mathsf{P}_{/\mathsf{poly}}$, the best explicit circuit lower bounds we can prove are $3.1n - o(n)$ for B_2 circuits [LY21] and $5n - o(n)$ for U_2 circuits[7] [IM02]. In a more restricted setting, we know $n^{2-o(1)}$ lower bound for B_2-formulas [Nec66], $n^{3-o(1)}$ lower bound for U_2-formulas [And87; IN93; PZ93; Hås98; Tal14] and $n^{1+\Omega(2.42^{-d})}$ lower bound for TC^0 circuits of depth d [IPS93]. Super-polynomial lower bounds can only be proved up to ACC^0 [MW20], and even $\mathsf{NEXP} \not\subseteq \mathsf{TC}^0$ remains to be open (see, e.g., [Che18; CT19]). For PRFs, prior work [RR97; KL01] only refutes the existence of pseudorandom functions in AC^0, unweighted depth-2 threshold circuits, and in $\mathsf{AC}^0[p]$ with more than quasi-polynomial security for primes p. To the best of our knowledge, there are no known impossibility results of PRF on general circuits or TC^0 circuits with large depth.

Bootstrapping results and barriers. The interpretation of our results as a barrier in circuit complexity relates to the works on *hardness magnification* and *quantified derandomization*. We provide a complete discussion of related works in Section 3.1.

1.4 Organization of the paper

We will first give some intuition on how the upper bounds and lower bounds are derived in Section 2. Then we discuss in Section 3 how our results give tight barriers on particular bootstrapping

[7] In U_2 circuits, gates can compute fan-in-two Boolean functions except for XOR and its complement.

results in circuit complexity. In Section 4, we will formally define our notations. We then give technical proofs to upper bounds for general circuits in Section 5; upper bounds for low-depth circuits in Section 6; lower bounds for general circuits and $\mathsf{AC}^0[2]$ circuits in Section 7; and finally, the lower bounds for TC^0 circuits in Section 8.

2 Proof Overview

We now give intuitions of our results. In Section 2.1, we will first introduce the intuition of our PRF upper bounds. We will discuss the PRF lower bounds against B_2 circuits and $\mathsf{AC}^0[2]$ circuits in Section 2.2, and then the lower bound against TC^0 circuits in Section 2.3.

2.1 Upper bounds in various classes

Levin's trick. Our general paradigm of proving circuit upper bounds for pseudorandom functions is a generalization of the standard *domain extension* technique called *Levin's trick*. Informally, it states that we can construct PRFs with input length n as follows: we firstly shrink an n-bit input x to an n^ε-bit hash value $h(x)$ by a (uniformly computable) *almost universal hash function*, and then feed $h(x)$ to a PRF with inputs length n^ε. Let us consider B_2 circuits as an example. Assume the original PRF has circuit complexity n^c, we can choose $\varepsilon < \frac{1}{c}$ so that the "pseudorandom kernel" has complexity $o(n)$; therefore, the complexity of the resulting PRF mainly depends on the complexity of the hash function. Through this observation, we reduce the construction of efficient PRFs to the problem of designing low-complexity almost universal hash functions with output length n^ε for all ε. Similarly, efficient PRFs in low-depth circuit classes follow from low-complexity constructions of hash functions in low-depth classes.

Efficient construction via error-correcting codes. Previously, efficient constructions of almost universal hash functions were built upon efficient error-correcting codes. Let $E : \mathbb{F}_2^n \to \mathbb{F}_2^\ell$ be an ECC with constant rate (i.e. $\ell = O(n)$) and distance. We can construct a hash function with output length m as the following distribution: randomly choose $i_1, i_2, \ldots, i_m \in [\ell]$, generate the function

$$h(x) \triangleq E(x)_{i_1} \| E(x)_{i_2} \| \ldots \| E(x)_{i_m}.$$

For any $x_1 \neq x_2$, since $E(x_1)$ and $E(x_2)$ differ by $\Omega(n)$ indices, the collision probability of this hash function is at most $\exp(-\Omega(m))$. By utilizing ECC constructions in corresponding classes, this approach has been sufficient to prove $O(n)$ PRF upper bounds in B_2 and NC^1 circuits (see [Spi96]) and our $n^{1+O(\phi^{-d})}$ upper bounds in depth-d TC^0 circuits (see [CT19]).

Taking the intermediate primitive out. Inspired by the constructions of efficient ECC (see, e.g., [GDP73; Spi96; CT19]), we observe that one can construct efficient hash functions with a much simpler primitive, which we call *1-detector*. Intuitively, a 1-detector is a linear function $D : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that for all inputs $x \neq 0$ with Hamming weight smaller than a threshold r (called the *range*), $D(x)$ is not identically zero. If we view D as a "hash function", it guarantees that the hash value of distinct pairs (x, y) with a small Hamming distance will not collide. In addition, assuming that the range r is appropriately large and m is small, we can also avoid (with high probability) the collisions between pairs (x, y) with large Hamming distance by letting the hash value to be the concatenation of the 1-detector and a randomly selected subset of the input bits.

Following this observation, it is now sufficient to construct (uniformly) 1-detectors with small circuit complexity and a nice trade-off between parameters r and m. Gelfand, Dobrushin, and Pinsker [GDP73] showed the existence of (non-uniform) 1-detectors by the standard probabilistic method, which cannot be directly made uniform.

Weakly uniform construction from probabilistic argument. Although the 1-detector induced by [GDP73] (which has circuit complexity $3n$) is not uniform, we can still sample a 1-detector with a small failure probability. By an error-reduction trick, for all integer $d > 0$, we can construct a probabilistic polynomial time algorithm $\mathcal{A}$ such that $\mathcal{A}(1^n)$ generates a 1-detector with failure probability at most n^{-d}. We call such 1-detectors (and corresponding hash functions) *weakly uniform*. We generalize Levin's trick and show that it can be adapted for weakly uniform hash functions, which leads to a $3n + o(n)$ upper bound in B_2 circuits for non-uniform PRFs. Although this construction has a larger circuit size, it can achieve sub-exponential collision probability (rather than quasi-polynomial for the $2n$ construction below); therefore, it also provides a higher security level for the corresponding PRFs. We think this is of independent interest.

High-girth-graph based (randomized) 1-detectors. To overcome the technical limitation of [GDP73], we can again slacken the requirement to construct hash functions by allowing the 1-detectors to be randomized. In particular, we define a *randomized 1-detector* as a linear function $D : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that for all $x \neq 0$ with small Hamming weight, $D(\rho(x)) \neq 0$ with high probability for a random permutation ρ of the input bits. Randomized 1-detectors are sufficient since hash functions also have access to randomness. Perhaps surprisingly, such primitive is uniformly constructible with circuit complexity only $2n$ using *graphs with large girth*[8].

Let $G = (V, E)$ be a graph with n edges and $m = \Theta(n/\log n)$ vertices with girth $g = \Theta(\log n/\log\log n)$ (see, e.g., [Cha03]). We now show that the depth-1 $\mathsf{CC}^0[2]$ circuit D whose topology is specified by the edge-vertex incident graph[9] of G computes a randomized 1-detectors with input length n, output length m, and range $r = n/2$.

We identify the n input bits and the m gates of D as the edges and vertices in G, respectively. Let S_x be the set of edges in G corresponding to the 1-indices of $x \in \mathbb{F}_2^n$. It is called *good* if at least one vertex is adjacent to odd number of edges in S_x, or equivalently, $D(x) \neq 0$. In order to prove that D is a randomized 1-detector, it suffices to show that for all x with Hamming weight at most $n/2$, $S_{\rho(x)}$ is good with high probability given random permutation ρ of input bits. This is equivalent to show that for all $0 < \ell \leq n/2$, a randomly chosen subset of edges of size ℓ is good with high probability. We prove this by considering separately the cases for $\ell < g$ and $g \leq \ell \leq n/2$.

1. In the former case, we can see that every subset of size $\ell < g$ is good. Towards a contradiction let S be such a bad subset, the induced subgraph of S contains a Eulerian cycle of size $|S| < g$, which is impossible by the high-girth property.

2. In the latter case, we claim that if all but the first $\lceil g/3 \rceil$ edges has been chosen, there is at most one choice of the remaining $\lceil g/3 \rceil$ edges such that the resulting subset is bad: if it is not the case, the symmetric difference of these two bad subsets is a bad subset of size at most $2\lceil g/3 \rceil < g$ (for large g), which is impossible by the former case.

[8] The girth of an undirected graph is the length of the smallest (simple) cycle.

[9] The n input variables and m (output) gates of D correspond to the n edges and m vertices of G, respectively. The gate corresponding to a vertex u computes the parity of variables corresponding to the edges incident to u.

Reducing the output length. It can be easily verified that our high-girth-graph based construction yields almost universal hash functions with logarithmic shrinkage (i.e., $\mathcal{H}_1 : \{0,1\}^n \to \{0,1\}^m$ for $m = \Theta(n/\log n)$) in B_2 circuits of size $2n$, and in $\mathsf{CC}^0[2]$ circuits of wire complexity $2n$. By composing this hash function with the $O(n)$ size PRF in [IKOS08] we can already obtain a $2n + o(n)$ PRF upper bound for B_2 circuits.

To obtain PRF upper bounds for low-depth classes such as NC^1, TC^0, or $\mathsf{AC}^0[2]$, we still need to reduce the output length of the $\mathsf{CC}^0[2]$ hash function to n^ε for arbitrarily small $\varepsilon > 0$. Our plan is to compose two hash functions with different features: the hash function $\mathcal{H}_1$ with logarithmic shrinkage and low complexity; and another hash function $\mathcal{H}_2 : \{0,1\}^m \to \{0,1\}^{n^\varepsilon}$ with polynomial shrinkage and slightly super-linear wire complexity, which can be constructed in $\mathsf{CC}^0[2]$ using the error-correcting codes in [CT19]. The combination of these two hash functions turns out to be a hash function with both low complexity (in NC^1, $\mathsf{AC}^0[2]$, and TC^0) and large shrinkage, which satisfies our needs.

2.2 Lower bounds in general and $\mathsf{AC}^0[2]$ circuits

The $2n - O(1)$ lower bound against B_2 circuits The proof of our lower bound against B_2 circuits consists of two steps. Firstly, we define a combinatorial property $\mathcal{P}$ about B_2 circuits such that there exists a p.p.t. algorithm $\mathcal{A}$ that distinguishes circuits with the property $\mathcal{P}$ and a truly random function. Then we prove by wire counting that all circuits with complexity $2n - O(1)$ have the property $\mathcal{P}$, which shows our algorithm $\mathcal{A}$ can break PRF candidates with wire complexity $2n - O(1)$.

Let C be a circuit and I be the set of input variables. For simplicity, we assume that each variable in C has an out-degree of at least 1. We define the *critical path* of a variable $x \in I$ in the circuit C as the set of nodes reachable from x via nodes with out-degree exactly 1. We will utilize the (black-box natural) combinatorial property: *C contains two variables with intersecting critical paths*. For a circuit C with such property, there are $x, y \in I$ and a Boolean function G, for all restrictions ρ to $I \setminus \{x, y\}$, the type[10] of $C\restriction_\rho(x, y)$ only depends on the type of G. This structure makes the circuit distinguishable from a truly random function f because the type of $f\restriction_\rho$ is independently randomly chosen for each ρ.

Now we only need to prove a lower bound for circuits without intersecting critical paths (and no variable of out-degree zero), which can be done via a standard wire-counting argument. Intuitively, the n non-intersecting critical paths should give roughly $2n$ out-wires at their terminals. By analyzing the number of wires between the critical paths and the other part of the circuit, we can show that we need about $2n$ gates to handle all these out-wires.

Wire-complexity lower bound against $\mathsf{AC}^0[2]$ circuits. Using the $2n - O(1)$ lower bound against B_2 circuits, we can obtain a slightly stronger $2n + \Omega(\sqrt{n})$ wire complexity lower bound against $\mathsf{AC}^0[2]$ circuits using a simple case analysis. Assume that we want to break a PRF candidate in $\mathsf{AC}^0[2]$ with low wire complexity, say $2n + 0.1\sqrt{n}$. If there are at least $0.2\sqrt{n}$ gates in the circuit, we can translate it into a B_2 circuit of size $2n - 0.1\sqrt{n}$ and then break it with our distinguisher for B_2 circuits. Otherwise, it has only $0.2\sqrt{n}$ gates. By considering the number of input variables with in-degree 1, 2, and ≥ 3, we can show that there exists a pair (x, y) of variables, such that

[10] We can classify Boolean functions in $\mathbb{F}_2^2 \to \mathbb{F}_2$ into four types: trivial functions that output a constant, degenerate functions that depends on only one of its input, $\oplus$-type functions that are linear over $\mathbb{F}_2$ and $\wedge$-type functions that are quadratic.

swapping them does not deviate from the functionality of the circuit. This property can be used to distinguish it from a truly random function.

2.3 Lower bounds in TC^0 circuits

Our PRF lower bound against TC^0 circuits follows the random restriction method in [CSS18], which has been a standard tool to analyze small-size TC^0 circuits (see, e.g., [Tel18; HHTT21]). Their main observation is that after a random restriction ρ and a cleverly chosen restriction σ (based on the former random restriction and the circuit), with a nice probability, we can eliminate an entire layer of linear threshold gates while keeping a moderately large fraction of variables alive. This procedure is done by considering the variables and gates with different degrees separately and analyzing the effect of the random restriction on them.

Large variables. For variables with large out-degrees, we arbitrarily fix them to a constant in the restriction σ. There will not be many such variables since each of them contributes lots of wires.

Small variables and small gates. Since variables with large out-degrees have already been removed from the circuit, we can assume that all variables have out-degrees not too large. In this case, we can choose a large subset of variables by a graph-theoretic argument, such that each gate *of small in-degree* is fed by at most one chosen variable. Then by fixing other variables in the restriction σ, we can make all these small gates depend on only one of its inputs, hence can be eliminated.

Small variables and large gates. Now only small variables and large gates remain. We argue that each gate has a nice probability of being extremely biased after the restriction ρ by anti-concentration bounds. We can hence approximate these biased gates by constants. Since only a few unbiased gates remain in expectation, we can remove them by fixing all their inputs in σ.

What [CSS18] argued is that we can carefully choose the parameters such that after the above three processes, there are still sufficiently many (say $n^{0.99}$) variables unfixed. The starting point of our argument is that the process above can be intuitively abstracted as follows.

> *Suppose that the variable set is I. We firstly take a random restriction to all variables. Let I' denote the variables kept alive by the random restriction. With nice probability, a large subset $S \subseteq I'$ exists such that randomly fixing all variables not in S would make it possible for us with high probability to eliminate an entire layer of gates and obtain a new circuit that agrees with the original circuit on most of the assignments to the unfixed variables.*

The main technical difficulty in proving our PRF lower bound is to extract a black-box natural property from this white-box argument. If we do not need to choose such S according to the circuit, we can trivialize any depth-d TC^0 circuit by repeating the process above for d times. The key to resolving this issue is to define another distinguishing procedure whose correctness is ensured by the restriction lemma. We define an input x to be good w.r.t. a TC^0 circuit C if flipping a bit of x would not change the output of C. If each sparse TC^0 circuit of depth d has a large fraction of good inputs, we can also argue according to the restriction lemma that each TC^0 circuit of depth $d+1$ has a large fraction of good inputs. By induction, we can conclude as follows.

> *Let C be any sparse TC^0 circuit. For a uniformly random input x and an input y obtained by flipping a random bit of x, $C(x) = C(y)$ with non-negligible probability.*

This property itself does not suffice to break PRF candidates in sparse TC^0 because a truly random function f also has a large fraction of good inputs (for any $x \neq y$, $f(x) = f(y)$ with probability $1/2$). To deal with this issue, we transform the PRF candidate F that we want to break into a PRF F' with output length $\log^2 n$ by expanding the last $2 \log \log n$ input bits, which would not increase its circuit complexity significantly. It is easy to verify that the property above still holds for multi-output functions. For truly random function, however, the probability that $f(x) = f(y)$ for $x \neq y$ reduces to $2^{-\log^2 n}$. This difference makes it possible to distinguish F' (and therefore F) from a truly random function.

3 A new barrier for bootstrapping results in circuit complexity

We now formally discuss the relationship between our exact complexity results and the bootstrapping results in circuit complexity. In Section 3.1, we briefly introduce recent bootstrapping results and known barriers. We discuss why the natural proof barrier [RR97] fails to capture the limitations of these bootstrapping results in Section 3.3. Then we introduce our *black-box natural proof barrier* in Section 3.4 and demonstrate how it captures the technical limitation of some recent bootstrapping results in [OPS19; CT19; CJW20] with "sharp thresholds" in Section 3.5.

3.1 Background: bootstrapping results in circuit complexity

A sequence of recent works in circuit complexity (see, e.g., [AK10; GW14; OS18; Tel18; OPS19; CT19; MMW19; CJW19; Che+20; CJW20]) revealed a mysterious phenomenon that a modest improvement over known circuit lower bounds or related problems may lead to a massive breakthrough in complexity theory. We briefly summarize the bootstrapping results in *hardness magnification* and *quantified derandomization* to gain more intuition.

Hardness magnification. *Hardness magnification* refers to the phenomenon that a relatively weak lower bound for some particular problems would imply a strong complexity class separation. Allender and Koucký [AK10] showed the existence of an NC^1-complete problem whose $n^{1+\varepsilon}$ lower bound against TC^0 circuits would lead to the separation of NC^1 and TC^0. It was then strengthened and generalized to many meta-complexity problems in [OS18; OPS19]. Most interestingly, [OPS19] showed that a weak lower bound for Gap-MCSP or Gap-MKtP[11] would lead to breakthroughs like $\mathsf{NP} \not\subseteq \mathsf{NC}^1$ or $\mathsf{EXP} \not\subseteq \mathsf{P}_{/\mathsf{poly}}$. McKay, Murray, and Williams [MMW19] proved that the similar magnification phenomenon holds from weak worst-case lower bounds of MCSP or MKtP using new techniques. The range of problems whose lower bounds admit such magnification is further extended by [CJW19] to all sparse NP languages, once again by a different approach. Using a derandomized restriction lemma, Chen, Jin, and Williams [CJW20] complemented the magnification result under the setting of probabilistic U_2-formulas by showing an MCSP lower bound, which nearly matches the hardness magnification threshold. Chen et al. [Che+20] gave a good survey on these results; they pointed out that in many settings, the current results can be viewed as a *sharp frontier*, in the sense that slight improvements to known lower bounds could imply big breakthroughs.

These hardness magnification results are typically proved by contrapositive. For instance, if we want to show that $n^{1+\varepsilon}$ lower bound for Gap-MCSP of particular parameters can imply

[11] Gap-MCSP$[s_1(n), s_2(n)]$ is the promise problem of MCSP where the given input has circuit complexity either greater than $s_2(n)$ or less than $s_1(n)$. Gap-MKtP is similarly defined for MKtP

$\mathsf{NP} \not\subseteq \mathsf{P}_{/\mathsf{poly}}$ [OPS19], we cleverly construct a language L which is computable by polynomial-size circuits assuming $\mathsf{NP} \subseteq \mathsf{P}_{/\mathsf{poly}}$, and show that any n-bit instance of Gap-MCSP with the parameters can be reduced to an n^{ε}-bit instance of L. If the reduction is super-efficient (say in $n^{1+\varepsilon}$-size circuits), and ε is arbitrarily small, then we can conclude with a small-size circuit computing Gap-MCSP. The most pivotal point in this argument is the reduction, which is done by error-correcting codes [OS18; OPS19], universal hash functions [CJW19; CJW20], or tree-like structures of particular problems [AK10; MMW19].

Quantified derandomization. *Quantified derandomization* is the task to derandomize probabilistic algorithms with extremely small error probabilities (say $2^{n^{0.1}-n}$). Formally, given a $\mathcal{C}$-circuit C which either accepts all but $B(n)$ inputs or rejects all but $B(n)$ inputs (say $B(n) = 2^{n^{0.1}}$), we need to design a deterministic algorithm to distinguish these two cases. Goldreich and Wigderson [GW14] first introduced this problem and revealed a surprising fact that solving the quantified derandomization for small $B(n)$ could imply the standard derandomization for $\mathsf{AC}^0[2]$ circuits. This is a bootstrapping phenomenon on the error probability of derandomization problems. Recent follow-up works [Tel18; CT19; CJW20] showed that this problem admits another interesting bootstrapping on the size of the circuit to be derandomized, which is similar to hardness magnification. These bootstrapping results on size lead to sharp phenomena in many natural classes such as TC^0 [CT19] and probabilistic formulas [CJW20]. We mainly focus on this dimension of bootstrapping in this paper.

We briefly illustrate the results in [CJW20] as an example. They showed that quantified derandomization of $n^{1.9}$ size probabilistic U_2-formulas with $B(n) = 2^{n^{0.01}}$ is solvable while improving the constant 1.9 to 2 with the same $B(n)$ would imply standard derandomization for all polynomial-size probabilistic U_2 formulas.

Typically, known quantified derandomization algorithms on the tractable side generally follow from derandomizing the standard random restriction methods. For instance, the quantified derandomization algorithm for AC^0 in [GW14] comes from derandomizing the switching lemma of [Hås86], and the one for TC^0 in [Tel18] comes from derandomizing a random restriction lemma in [CSS18]. On the other hand, the bootstrapping side is usually done by an error-reduction trick using Trevisan's extractor [Tre01; RRV02], which is essentially a clever sampling over an error-correcting code. The bootstrapping phenomena can be proved in many classes such as $\mathsf{AC}^0[2]$ circuits and TC^0_d circuits of size $n^{1+O(\phi^d)}$ by constructing error-correcting codes in corresponding classes. We refer interested readers to the survey by Tell [Tel21] for a comprehensive discussion of quantified derandomization.

3.2 Known barriers for bootstrapping results

Apart from the obvious view that these bootstrapping results shed new light towards resolving longstanding open problems, some researchers hold a more pessimistic view that the results suggest the inherent difficulty to prove the bootstrapping premise. In order to consolidate the pessimistic view, it is helpful to investigate why current techniques cannot be improved to obtain breakthroughs. In particular, a few barriers[12] have been introduced for hardness magnification

[12]Note that different from the well-known natural proof barrier [RR97] that only captures the limitation of lower bound techniques, a barrier for bootstrapping phenomena says that the bootstrapping theorems (from certain kinds of techniques) cannot be combined with explicit lower bounds or quantified derandomization (from certain kinds of techniques) to obtain a breakthrough. Therefore it should contain two sides: a *bootstrapping side* that captures the techniques for proving bootstrapping results, and a *tractable side* that captures the lower bound (or quantified derandomization)

and quantified derandomization separately.

- For hardness magnification, the *locality barrier* [Che+20] captured the conflicts between lower bound techniques and magnification techniques. The observation is as follows. On the bootstrapping side, known techniques usually imply an *unconditional* oracle circuit upper bound, which says that the problem for magnification can be computed by small-size oracle circuits with only *few oracle gates of small fan-in*. On the tractable lower bound side, it is known that an extensive range of lower bound techniques can be extended to prove lower bounds against oracle circuits with few oracle gates with small fan-in. Since the unconditional upper and lower bounds cannot conflict with each other, the magnification threshold cannot be made smaller than provable lower bounds using these two kinds of techniques.

- For quantified derandomization, Tell [Tel17; Tel21] showed that two natural "black-box" techniques for quantified derandomization, random restriction (used to design quantified derandomization algorithms) and error-reduction by extractors (used to prove bootstrapping results), cannot be combined together to derive a standard derandomization algorithm. The intuition of this barrier is that the output bits of an extractor cannot be easily trivialized with random restrictions; therefore any circuit class that can compute extractors should not have a simple restriction lemma for us to design quantified derandomization algorithms.[13]

In the remaining part of this section, we will demonstrate a new barrier, which is inspired by both the natural proof barrier [RR97] and our investigation in the complexity of PRFs, that applies to hardness magnification, quantified derandomization, and potentially other bootstrapping results with similar structures. The new barrier, which we call the *black-box natural proof barrier*, explains the limitation of the bootstrapping results by showing that the improvements of the known lower bounds (or quantified derandomization algorithms) and the bootstrapping premises contradict each other as long as we stick to certain kinds of techniques related to the complexity of PRFs. Our barrier provides a unified view on all these bootstrapping results and is arguably more general than that of Chen et al. [Che+20] and Tell [Tel17].

3.3 Natural proof barrier and its limitation

Let us first review the natural proof framework by Razborov and Rudich [RR97]. Let Γ and Λ be typical complexity classes, and $B_n \triangleq \{0,1\}^n \to \{0,1\}$ be the set of all n-bit Boolean functions. A combinatorial property $\mathcal{P} = \{P_n \subseteq B_n\}_{n\geq 1}$ over functions of input lengths n is Γ-*natural against* Λ if the following conditions holds.

(Constructivity) Given the truth table of a function $f_n \in B_n$ as input (note that the input length is 2^n), the language indicating whether f_n is in C_n is decidable in Γ.

(Smallness) There exists a constant $c > 0$ such that, for sufficiently large n, let $g_n \in B_n$ be a uniformly random function from all possible functions, we have $\Pr[g_n \in C_n] \leq 1 - 2^{-cn}$.

(Usefulness) For any language $L = \{L_n\}_{n\geq 1} \in \Lambda$, $L_n \in P_n$ for infinitely many n.

techniques.

[13] We clarify that Tell's barrier mainly considers the bootstrapping of the parameter $B(n)$ for quantified derandomization (see, e.g., [Tel21]), while our barrier focuses on the bootstrapping of circuit size. To the best of our knowledge, our barrier and Tell's barrier are technically incomparable.

As the name indicates, natural proof models the most natural way to prove explicit circuit lower bounds. Indeed, a large range of known techniques for proving unconditional circuit lower bounds fall into this region. What Razborov and Rudich [RR97] observed is that a natural proof induces an algorithm in Γ that distinguishes any function family in Λ from truly random functions given the truth tables. Hence assuming the existence of exponentially hard PRFs against Γ computable in Λ, such combinatorial properties do not exist. For a typical setting such as $\Gamma = \Lambda = \mathsf{P}_{/\mathsf{poly}}$, the existence of the desired PRF follows from the existence of exponentially hard PRG via [GGM84].

The natural proof paradigm successfully explains why proving super-polynomial circuit lower bounds is hard. By proposing candidates of exponentially-secure PRFs in low-complexity classes, we can even rule out natural proofs for lower bounds against TC^0_d circuits with $n^{1+O(1/d)}$ wires and $\widetilde{O}(n)$-size general circuits[14] assuming stronger cryptographic assumptions (see, e.g., [MV15]). This has already been sufficient to show the limitation of some bootstrapping results requiring explicit lower bounds against $n^{1+\varepsilon}$-size general circuits or TC^0 circuits.[15] However, it cannot capture many interesting bootstrapping results requiring only mild lower bounds due to our limited knowledge about the complexity of exponentially-secure PRFs.

Take the bootstrapping of quantified derandomization in [Tel18; CT19; Tel21] as an example. Their results roughly say that quantified derandomization for TC^0_d circuits of $n^{1+60^{-d}}$ wires with $B(n) = 2^{n^{1-1.61^{-d}}}$ can be solved efficiently, while improving the constant 60 to 1.61 would imply big breakthroughs such as $\mathsf{NEXP} \not\subseteq \mathsf{TC}^0$. In order to put a barrier on this bootstrapping result, the idea is to refute the existence of natural properties against $n^{1+1.61^{-d}}$ size TC^0_d circuits, so that improving the known quantified derandomization algorithm using natural proofs would be impossible. To do this, we need to propose candidates of *exponentially hard* PRFs in TC^0_d circuits of size $n^{1+1.61^{-d}}$, which seems quite hard (recall that the best known candidate in TC^0 has $n^{1+O(1/d)}$ wires [MV15]). In general, the bootstrapping thresholds are usually determined by the complexity of certain combinatorial primitives which support the magnification arguments (e.g., error-correcting codes, universal hash functions, or seeded extractors). In most cases, we do not know plausible candidates of exponentially strong PRFs with such small complexity.

3.4 A new barrier: black-box natural proof

We want to show that one cannot improve the bootstrapping thresholds and the tractable lower bounds to obtain a breakthrough with some typical techniques. Our plan is to refine the natural proof barrier (which only has the tractable side) by restricting the concept of *natural properties*, so that the PRFs required for the barrier can be constructed *from the techniques of the bootstrapping side*.

On the tractable side, we can argue that some techniques for circuit lower bounds and quantified derandomization, such as the switching lemma of AC^0 [Hås86] and the shrinkage exponent of U_2-formulas [IN93; PZ93; Hås98; Tal14], actually imply PRF distinguishers that are much more efficient than the requirements of natural proofs. Take the random restriction method as an example. If we can show that certain circuits become constants under random restrictions with nice probability, we can distinguish them from truly random functions by simulating the restriction procedure and checking whether the resulting function is constant. So it is sufficient to use stan-

[14] Recall that $\widetilde{O}$ notation hides the poly-logarithmic factors, i.e., $\widetilde{O}(T(n)) \triangleq T(n)\log^{O(1)} T(n)$.

[15] Here the natural proof barrier is interpreted as a barrier for bootstrapping results with only the tractable side: instead of saying that the bootstrapping and lower bound techniques contradict with each other, it shows that the lower bound itself is hard to prove.

dard cryptographic PRFs to refute the existence of such proofs, which can be constructed in weak classes like $2n + o(n)$ size general circuits, $n^{1.01}$ size B_2 formulas or depth-d threshold circuits of size $n^{1+O(c^{-d})}$ by our upper bounds.

On the bootstrapping side, we can see that most recent works on hardness magnification [OPS19; CJW19; CJW20] and quantified derandomization [CT19; CJW20] explicitly utilize combinatorial primitives such as error-correcting codes, universal hash functions, and seeded extractors[16]. In fact, the bootstrapping thresholds in these works tightly match the complexity of these primitives in corresponding circuit classes. According to Levin's trick (see Lemma 4.11) and our construction of almost universal hash functions from error-correcting codes (see Proposition 4.8), these primitives can be used to construct standard cryptographic PRFs with almost the same complexity.

Intuitively, these two observations explain the difficulty of obtaining a breakthrough via the bootstrapping results. Improving the bootstrapping thresholds using many techniques mentioned before requires more efficient combinatorial primitives, which can be used to obtain tighter PRF upper bounds. Improving the tractable results with specific techniques like random restrictions would imply PRF distinguishers and therefore give stronger PRF lower bounds. If we obtain a breakthrough by matching the thresholds and the tractable bounds, we will refute the existence of PRFs in that circuit class simultaneously[17], which is believed to be impossible.

Now we formally state the black-box natural proof barrier in a form similar to natural proofs. Let Λ be a typical circuit class (e.g., general circuits of size $2.01n$). We call a combinatorial property $\mathcal{P} = \{P_n\}_{n\geq 1}$ *black-box natural against* Λ if it satisfies the following properties.

(Black-box constructivity) There exists an oracle probabilistic polynomial time algorithm $\mathcal{A}^O$ such that the following conditions hold.

- For any $f_n \in P_n$, we have $\Pr_{\mathcal{A}}\left[\mathcal{A}^{f_n}(1^n) = 1\right] \geq 2/3$.
- For uniformly random function g_n, we have $\Pr_{g_n,\mathcal{A}}\left[\mathcal{A}^{g_n}(1^n) = 1\right] \leq 1/3$.

(Usefulness) For any language $L = \{L_n\}_{n\geq 1} \in \Lambda$, $L_n \in P_n$ for infinitely many n.

The usefulness in the above definition is completely the same as the one in the original natural proof. We also note that smallness is not explicitly stated here because it is implied by our definition of black-box constructivity. The probability thresholds $\delta_1 = 2/3$ and $\delta_2 = 1/3$ in black-box constructivity are taken for simplicity of presentation and can be arbitrary functions of n with a non-negligible gap. We also emphasize that we can extend the definition of black-box constructivity by allowing algorithm $\mathcal{A}$ to run in quasi-polynomial or even sub-exponential time if PRFs with stronger security can be constructed (recall the discussion in Section 1.1). Similar impossibility results still hold for typical classes that can compute sub-exponentially strong PRFs. We stick to the standard setting (i.e., p.p.t. adversary) here mainly for simplicity.

[16]Note that typical constructions of seeded extractors coming from the hardness vs. randomness paradigm [Tre01; RRV02] are essentially cleverly chosen projections over error-correcting codes.

[17]Note that this argument has a small loophole. Assume that we can construct an ECC with circuit complexity $s(n)$, then the corresponding PRF upper bound should be slightly larger than $s(n)$ (e.g. $s_{\mathsf{prf}}(n) = s(n) + n^{0.01}$). If the bootstrapping threshold could be made closer (e.g. $s_{\mathsf{th}}(n) = s(n) + O(\log n)$), it is possible to obtain a breakthrough with a black-box natural property against circuits of size $s_{\mathsf{th}}(n)$, while keeping consistent with the existence of PRFs since $s_{\mathsf{th}}(n) > s_{\mathsf{prf}}(n)$. Arguably, this loophole does not reduce the strength of our barrier since it is not so likely that the bootstrapping threshold $s_{\mathsf{th}}(n)$ can be made such close to $s(n)$. In hardness magnification results, for example, realizing the oracle gates with actual circuits could introduce an overhead which can be larger than the complexity we need to build PRFs from ECCs.

Tractable side. With the definition of the black-box natural property, the tractable side of our barrier is indeed a straightforward refinement of the standard natural proof barrier by replacing the exponentially-secure PRF with a polynomially-secure PRF.

Proposition 3.1 (The tractable side of black-box natural proof barrier). Let $\mathcal{C}$ be a circuit class. If standard cryptographic PRFs can be constructed by $s(n)$-size $\mathcal{C}$-circuits, then there is no black-box natural property against $s(n)$-size $\mathcal{C}$-circuits. ♢

Since we can obtain tight upper bounds for PRFs in various circuit classes, the tractable side itself can be considered as an individual barrier for circuit lower bound techniques in spite of being less general than the natural proof barrier. We provide some examples with the PRF upper bounds in Theorem 1.1 and lower bounds in Theorem 1.2 to demonstrate the tractable side of the black-box natural proof barrier.

- Our PRF in B_2 circuits shows that black-box natural properties against $\mathsf{SIZE}[2n + o(n)]$ do not exist, assuming PRFs exist, which directly follow from the existence of one-way functions [GGM84; HILL99]. We note that Theorem 1.2 (also see Lemma 7.4), in fact, gives a black-box natural property against $\mathsf{SIZE}[2n - O(1)]$. One may check that known better-than-$2n$ explicit circuit lower bounds [Sto77; Pau77; Blu84; DK11; FGHK16; LY21], proved by *gate elimination*, highly rely on a non-black-box procedure that eliminates all gates by introducing constraints to input bits *cleverly* according to the circuit.
- Our PRFs in NC^1 circuits show that black-box natural properties against $(1+\varepsilon)\log n$ depth circuits do not exist, assuming NC^1 PRFs exist. From the general connections between NC^1 and formula complexity, this means that black-box natural properties against $n^{1+\varepsilon}$ size B_2-formulas or $n^{2+\varepsilon}$ U_2-formulas do not exist for arbitrarily small $\varepsilon > 0$. We note that the quadratic lower bound for B_2-formulas [Nec66] and the cubic lower bounds for U_2-formulas [And87; IN93; PZ93; Hås98; Tal14] are natural in the sense of [RR97], but does not seem to be black-box natural because they utilize non-constructive counting arguments on particular functions [Nec66; And87]. However, we still note that these arguments essentially follow from the shrinkage exponent of the formula classes, which are black-box natural against probabilistic B_2-formulas of size $n^{1+\varepsilon}$ or probabilistic U_2-formulas of size $n^{2+\varepsilon}$.
- Our PRFs in TC^0 circuits show that black-box natural properties against depth-d TC^0 circuits of $n^{1+O(\phi^{-d})}$ wire complexity do not exist, assuming TC^0 PRFs exist. On the other hand, the main lemma for our TC^0 PRF lower bound (see Lemma 8.3), whose variants are also used by [CSS18; Tel18; HHTT21], can be viewed as a black-box natural property against TC^0 circuits of wire complexity $n^{1+\Omega(c^{-d})}$.

Bootstrapping side. The bootstrapping side of our barrier captures the techniques for bootstrapping results whose bootstrapping thresholds depend mostly on the complexity of error-correcting codes or (almost universal) hash functions. The following proposition is a simple consequence of Levin's trick (see Lemma 4.11) and the construction of almost universal hash functions from error-correcting codes (see Proposition 4.8).

Proposition 3.2 (The bootstrapping side of black-box natural proof barrier). Suppose that $\mathcal{C}$ is a circuit class that can compute standard cryptographic PRFs in polynomial size. If error-correcting codes with constant relative distance or almost universal hash functions with output length n^{ε} for arbitrarily small $\varepsilon > 0$ are computable with $s(n)$ size $\mathcal{C}$-circuits, then PRFs are computable with $\mathcal{C}$-circuits of size $s(n) + n^{\delta}$, where $\delta > 0$ is an arbitrarily small constant. ♢

The bootstrapping side of our barrier says that if one want to improve the bootstrapping result by constructing more efficient ECCs or hash functions, it will reduce the complexity of PRFs at the same time, which makes it harder to prove explicit lower bounds according to the tractable side. Note that most of the known techniques for bootstrapping results (see, e.g., [GW14; OS18; OPS19; CT19; CJW19; CJW20]) are indeed captured by the bootstrapping side of our barrier.

By combining the tractable side and the bootstrapping side, we can finally introduce the black-box natural proof barrier for bootstrapping results.

Theorem 3.3 (Black-box natural proof barrier). Let $\mathcal{C}$ be a circuit class that can compute PRFs in polynomial size. Suppose that we have the following two results.

(Bootstrapping side) A bootstrapping result with bootstrapping threshold $s_{\mathsf{th}}(n)$ that, as a byproduct, shows that error-correcting codes with constant relative distance or almost universal hash functions with output length n^{ε} for arbitrarily small $\varepsilon > 0$ can be computable by $\mathcal{C}$-circuits of size $s_{\mathsf{th}}(n) - n^{\Omega(1)}$.

(Tractable side) An explicit result (e.g., lower bounds or quantified derandomization) for $s_{\mathsf{tr}}(n)$-size $\mathcal{C}$-circuits that relies on a black-box natural property of $s_{\mathsf{tr}}(n)$-size $\mathcal{C}$-circuits.

Then $s_{\mathsf{th}}(n) > s_{\mathsf{tr}}(n)$, i.e., these two results cannot be combined to obtain a breakthrough. ♢

3.5 Concrete examples of the black-box natural proof barrier

From the discussion above it is clear that the concept of the black-box natural property is less general than the natural property because several practical techniques (e.g., $\mathsf{AC}^0[2]$ lower bound from polynomial method [Raz87; Smo87] and Andreev's function lower bounds from shrinkage exponent [And87; Hås98; Tal14]) do not induce such efficient PRF distinguishers.

This section discusses the strength of the black-box natural proof barrier by showing that most of the bootstrapping results with "sharp thresholds" [OPS19; CT19; CJW20] can be captured by it. Our barrier provides formal evidence that current techniques for the sharp threshold results are not likely to be simply improved in both the bootstrapping sides and the tractable sides to obtain the desired breakthroughs.

3.5.1 Chen-Jin-Williams's sharp threshold result on hardness magnification

A recent work of Chen, Jin, and Williams [CJW20] showed that probabilistic U_2-formulas admit sharp bootstrapping phenomena on hardness magnification, quantified derandomization, and explicit obstruction. We will focus on the hardness magnification results here, but similar arguments hold for the other two.

A probabilistic U_2-formula is a distribution $\mathcal{F}$ on U_2-formulas; and it is said to compute a function h if $\Pr_{f \leftarrow \mathcal{F}}[f(x) = h(x)] \geq 2/3$ for all x. Chen, Jin, and Williams [CJW20] observed that MCSP with certain parameters does not have $n^{2-o(1)}$ size probabilistic U_2-formulas, while improving the lower bound to $n^{2+\varepsilon}$ for any $\varepsilon > 0$ would imply $\mathsf{NP} \not\subseteq \mathsf{Formula}[n^k]$ for all k. This result is quite surprising, given the fact that there exists some explicit function that $n^{3-o(1)}$ lower bounds for probabilistic U_2-formulas are known, which follows from the average-case lower bounds in [KRT17; CKLM20].

Now we demonstrate how our barrier applies to this hardness magnification result. We consider both techniques used by the bootstrapping side and the tractable side of the result.

Bootstrapping side. The magnification part of Chen, Jin, and Williams [CJW20] comes from the *kernelization method* in [CJW19]. It utilizes an efficient universal hash function with a short seed length. Concisely speaking, they argue that to probabilistically decide a sparse language[18] like MCSP (with certain parameters), one can hash down the input and query an NP-oracle whether the hash value (together with the seed) covers a YES-instance. By choosing a sufficiently good hash function, we can guarantee at most one YES-instance corresponding to each hash value. By encoding the instance with an error-correcting code, we can probabilisticall check whether the current input is the YES-instance. Choosing the parameters cleverly, we can realize the NP-oracle with complexity $n^{0.01}$, assuming $\mathsf{NP} \subseteq \mathsf{Formula}[n^k]$ for some k. In such case, the dominating part of the magnification threshold is determined by the hash function.

Tractable side. The $n^{2-o(1)}$ lower bound for MCSP against probabilistic U_2-formulas in [CJW20] is proved by a derandomized version of shrinkage theorem for U_2-formulas [Hås98; Tal14], following the iterated pseudorandom restriction method [IMZ19; HS17; OPS19]. Take the shrinkage theorem of Tal [Tal14] as an example. It said that for all $p > 0$ and function f with U_2-formula complexity L, if we apply a random restriction $\rho \leftarrow \mathcal{R}_p$, the expected formula complexity of $f\restriction_\rho$ is $O(p^2L + p\sqrt{L})$. This property is black-box natural because we can easily simulate the restriction process with $p = n^{o(1)-1}$ and check whether the function shrinks to a constant to distinguish small size U_2-formulas from truly random functions.

Since the two sides of the magnification results imply the upper and lower bounds for PRFs in U_2-formulas, respectively, the gap between them cannot be closed unless there is no PRF in NC^1. To obtain a breakthrough from this bootstrapping result, one needs to either introduce a new bootstrapping method that avoids the usage of hash functions or prove a stronger circuit lower bound for MCSP with non-black-box techniques.

3.5.2 Chen-Tell's result on quantified derandomization

Another recent sharp threshold result due to Chen and Tell [CT19] is about quantified derandomization of TC^0_d circuits. They showed that there exists some constant $c > 30$ such that quantified derandomization for moderately large $B(n)$ is possible for TC^0 circuits of size $n^{1+O(c^{-d})}$, and improving the constant c to 1.61 would imply standard derandomization for all polynomial-size TC^0 circuits. Again, we consider both the bootstrapping and the tractable sides of their result.

Bootstrapping side. The bootstrapping result of Chen and Tell [CT19] follows the usual paradigm of error-reduction by extractors. The idea was first introduced by Goldreich and Wigderson [GW14] and is essentially the only way of proving bootstrapping results for quantified derandomization. Following this proof, the key ingredient is an efficient seeded extractor constructed using error-correcting codes and Trevisan's extractor [Tre01; RRV02]. Here the complexity of the error-correcting code is the only bottleneck throughout the construction, and indeed the main technical contribution of [CT19] is an extremely efficient construction of error-correcting codes in TC^0[19]. Following our general framework of proving PRF up-

[18] Sparse languages are those that have only a tiny amount of YES-instances (e.g., only 2^{n^ε} YES-instances for input lenght n). Note that a simple counting argument can show the sparsity of MCSP-like problems.

[19] The original construction of ECC in [CT19] is linear, so that its complexity is limited by the parity lower bound [IPS93]. They proposed an open problem to improve their construction with a non-linear ECC. We resolve this problem by showing that even non-linear ECC cannot be used to improve this upper bound. In fact, we can derive an *unconditional* lower bound for error-correcting codes from the hash function lower bound (see Theorem 8.5)

per bounds (see Section 2.1 and 4.5), any improvement of the error-correcting code can be translated to a corresponding PRF upper bound.

Tractable side. This part follows from a pseudorandom restriction lemma for linear threshold functions [Tel18], which is essentially the derandomized version of Lemma 8.3 (also see [CSS18]). Although this restriction lemma seems to be a white-box procedure, we can extract a black-box natural property by Lemma 8.4 and Theorem 8.1, and therefore derive a polynomial-time PRF distinguisher against TC^0_d circuits of size $n^{1+O(c_1^{-d})}$. This hints that a simple improvement of the quantified derandomization algorithm following current techniques would imply a stronger PRF lower bound at the same time.

Assume that PRFs can be constructed in TC^0 (following from standard cryptographic assumptions such as DDH, factoring, and ring learning-with-errors). Simple improvement of both sides of the result cannot lead to a breakthrough. A potential way to cross our barrier is to introduce an inherently non-black-box technique for quantified derandomization of TC^0, which can possibly work for larger TC^0 circuits.

3.5.3 Oliveira-Pich-Santhanam's hardness magnification results for MKtP

The last one we will discuss is the hardness magnification results of Oliveira, Pich, and Santhanam [OPS19]. This is an example of sharp threshold magnification that cannot be perfectly captured by our black-box natural proof barrier, but we can still gain some insights when viewing it in this way. Oliveira, Pich, and Santhanam [OPS19] investigated the sharp hardness magnification phenomena in different circuit classes for the approximate version of MKtP. Interestingly, they also gave explicit circuit lower bounds against MKtP with different parameters that tightly match the magnification threshold.

We focus on their magnification result for U_2-formulas. Following the notation of [OPS19], let $\mathsf{Gap\text{-}MKtP}[s_1(n), s_2(n)]$ be the promise problem to distinguish length-n strings with Kt complexity smaller than $s_1(n)$ with those with Kt complexity larger than $s_2(n)$.

Bootstrapping side. The magnification theorem says that for a universal constant c, if there exists an $\varepsilon > 0$ such that for arbitrarily small $\beta > 0$, $\mathsf{Gap\text{-}MKtP}[n^\beta, n^\beta + c \log n]$ cannot be computed by U_2-formulas of size $n^{3+\varepsilon}$, then $\mathsf{EXP} \not\subseteq \mathsf{NC}^1$. Their main observation is the following random compression procedure. Let E be an error-correcting code. Given a string x of length n with low (resp. high) Kt complexity, the string obtained by randomly selecting $m = n^\beta$ indices of $E(x)$ will have relatively low (resp. high) complexity. This compression procedure immediately gives a randomized reduction from Gap-MKtP to the following intermediate language

$$L = \{(a, 1^b, (i_1, \alpha_1), \ldots, (i_r, \alpha_r)) \mid \exists |x| = a, \mathsf{Kt}(x) \leq b \wedge \forall j, x_{i_j} = \alpha_j\}.$$

Assuming that $\mathsf{EXP} \subseteq \mathsf{NC}^1$, we can then construct a (one-sided error) probabilistic U_2-formula computing Gap-MKtP of size roughly $n^{2+\beta}$ (for each of the $m = n^\beta$ selected indices, we need to compute a parity of n variables). Finally, we can derandomize the probabilistic formula by $O(n)$ parallel repetition to obtain a (deterministic) U_2-formula of size roughly $n^{3+\beta}$ computing Gap-MCSP.

Tractable side. Oliveira, Pich, and Santhanam [OPS19] also give a matching $n^{3-o(1)}$ U_2-formula lower bound against Gap-MKtP *with different parameters*. The idea is quite simple: if Gap-MKtP

with appropriate parameters can be decided by $n^{3-o(1)}$-size U_2-formulas, we can distinguish strings of low Kt complexity with truly random strings, which will break the (unconditional) pseudorandom generator of Impagliazzo, Meka, and Zuckerman [IMZ19]. We note that the pseudorandom generator in [IMZ19] essentially follows from the shrinkage exponent of U_2 formulas [Hås98; Tal14].

To improve the *magnification threshold*, one can construct a more efficient ECC or utilize a better derandomization method. Our black-box natural proof barrier limits the former approach since we can prove PRF lower bound against probabilistic U_2-formulas of size $n^{2-o(1)}$. The latter does not seem to be directly limited by our barrier, though some evidence shows that derandomization requires this n overhead [CT21]. In order to improve the *tractable lower bound*, the most straightforward way is to introduce a better (i.e., low Kt complexity) pseudorandom generator against U_2-formulas, which does not necessarily prove a better PRF lower bound. Still, the current proof follows from an $n^{2-o(1)}$ probabilistic U_2 formula lower bound, which is black-box natural, so at least we can say that this part cannot be improved.

4 Preliminaries

Throughout this paper, we define $[n] \triangleq \{1, 2, \ldots, n\}$, $B_n \triangleq \mathbb{F}_2^n \to \mathbb{F}_2$ as the set of single-output Boolean functions with n inputs, and $B_{n,m} \triangleq \mathbb{F}_2^n \to \mathbb{F}_2^m$ as the set of m-output Boolean functions with n inputs. We represent Boolean AND function with $\wedge$ and Boolean XOR function with $\oplus$. For binary strings x and y of length n, the *Hamming weight* $|x|$ is defined as the number of 1-entries in x, and the *Hamming distance* $\Delta(x, y) \triangleq |x \oplus y|$ is defined as the Hamming weight of point-wise XOR of x and y. The *relative distance* $\Delta_r(x, y)$ is defined as $\Delta(x, y)/n$. We use $x \| y$ to denote the concatenation of two bit strings x and y.

All the graphs $G = (V, E)$ are undirected in default. A cycle in a graph G is a subset of vertices $\{v_0, v_1, \ldots, v_{\ell-1}\}$ such that there is an edge between v_i and $v_{(i+1) \bmod \ell}$ for all $0 \leq i < \ell$. We follow standard notations for probability and expectation, where $x \leftarrow \mathcal{D}$ represents that x is a random variable sampled according to the distribution $\mathcal{D}$. In particular, for any finite set S, $x \leftarrow S$ means that x is the random variable sampled according to uniform distribution supported on S.

Without further clarification, pseudorandom functions are meant to be secure against uniform probabilistic polynomial time (p.p.t. for short) adversary.

4.1 Circuit classes

4.1.1 B_2 circuits

A *Boolean circuit* (or B_2 *circuit*) is a directed acyclic graph where each vertex is either a *variable* of in-degree 0 or a *gate* of in-degree 2. Each variable is labeled with an index identifying its corresponding input bit, and each gate has a corresponding Boolean function out of B_2. One or more nodes are marked as output nodes, each of which is labeled with a set of indices identifying the corresponding output bits[20]. During the evaluation, we decide the output of each gate according to its corresponding function in topological order. We say a circuit C computes a function $f \in B_{n,m}$ if C contains exactly n variables and m output nodes, and it agrees with f on all inputs in $\mathbb{F}_2^n$.

According to the functionality, we can classify the 16 gates out of B_2 into four types: *trivial gates* that compute constant functions (i.e., $f(x, y) = c_1$); *degenerate gates* that only depend on one

[20] That is, a node can have more than one corresponding output bits.

of their inputs (i.e., $f(x,y) = x \oplus c_1$ or $f(x,y) = y \oplus c_2$); *$\oplus$-type gates* that compute linear functions (i.e., $f(x,y) = x \oplus y \oplus c$); and *$\wedge$-type gates* that compute quadratic functions (i.e., $f(x,y) = ((x \oplus c_1) \wedge (y \oplus c_2)) \oplus c_3$). It is easy to see that an optimal circuit computing any function f does not contain trivial and degenerate gates since we can always remove them and properly rewire the circuit while keeping the functionality of the circuit.

The *size* of a circuit is defined as the number of gates involved; and the *depth* of a circuit is defined as the number of edges in the longest path of the graph.

4.1.2 Low-depth circuit classes

In this work, we consider several circuit classes with structural restrictions.

NC^1 circuits. An NC^1_d circuit is a B_2 circuit with depth at most $d \log n$. We will simply call it an NC^1 circuit if we do not care about the exact value of the constant d. The size and depth of an NC^1 circuit are defined similarly to the B_2 circuits.

B_2-formulas. A B_2-formula is a single-output B_2 circuit whose topology is a tree rooted as the output node. The leaves of a B_2 circuit are input variables. The size of a B_2-formula is defined as the number of leaves in the tree. It is well-known that the class of polynomial-size B_2 circuits is equivalent to NC^1.

U_2-formulas. A U_2-formula is a B_2-formula which does not contain $\oplus$-type gates. The classes of polynomial size U_2-formulas and B_2-formulas are equivalent, but the B_2 and U_2 formula complexity of concrete problems could be different. For instance, the parity function $\bigoplus_n(x_1, \ldots, x_n) \triangleq x_1 \oplus \cdots \oplus x_n$ can be computed by B_2-formulas of size n, but requires U_2-formulas of size $n^{2-o(1)}$ [Hås98; Tal14].

$\mathsf{AC}^0[2]$ circuits. An $\mathsf{AC}^0_d[2]$ circuit is a depth-d circuit with $\{\wedge, \vee, \oplus\}$ gates of unbounded in-degree, and on each in-wire of the gates, it is allowed to attach a "free" $\neg$ gate (i.e., not counted in gate, wire and depth complexity). We simply call it an $\mathsf{AC}^0[2]$ circuit if $d = O(1)$ is indifferent in the context. The size of an $\mathsf{AC}^0[2]$ circuit can be measured using the number of gates or wires involved.

$\mathsf{CC}^0[2]$ circuits. A $\mathsf{CC}^0_d[2]$ circuit is a depth-d circuit with only $\oplus$ gates of unbounded in-degree. We define $\mathsf{CC}^0[2]$, depth and size similar to $\mathsf{AC}^0[2]$. Note that $\mathsf{CC}^0[2]$ is an incomplete circuit class and can only compute linear functions.

4.1.3 Threshold functions and threshold circuits

In this work, another computation model we are interested in is the *linear threshold circuit*. For notational convenience, we represent Boolean values by $\{1, -1\}$ instead of $\{0, 1\}$ when talking about linear threshold circuits (i.e., we represent `true` by -1 and `false` by 1, so that XOR is simply multiplication). We will also often omit "linear" since we will not consider any non-linear threshold functions in this paper.

Definition 4.1 (Linear threshold function). Let $w \in \mathbb{R}^m$ and $\theta \in \mathbb{R}$, a *linear threshold function* (or simply *threshold function*) corresponding to weights w and threshold θ is defined as $\mathsf{LTF}_{w,\theta}(x) \triangleq \mathrm{sgn}(\langle w, x \rangle - \theta)$, where $\langle \cdot, \cdot \rangle$ is the standard inner product of real vectors and $\mathrm{sgn}(x)$ is the sign function. $\diamondsuit$

A *linear threshold circuit* (or simply a *threshold circuit*) is a direct acyclic graph where each vertex is either a variable corresponding to an input bit or a gate of arbitrary in-degree labeled with a threshold function. Similar to B_2 circuits, one or more nodes of a threshold circuit are marked as output nodes.

The *depth* of a vertex in a threshold circuit is defined as the number of edges in the longest path from any variable to it. The *depth* of the threshold circuit is the maximum depth of all vertices. The *size* of a TC^0 circuit can be measured using the number of gates or wires, and in this paper, we only consider the wire complexity. There is usually a trade-off between the depth and the size to compute a function $f \in B_n$ with threshold circuits. For example, Paturi and Saks [PS94] shows that the parity function $\bigoplus_n(x_1, \ldots, x_n) \triangleq x_1 \oplus \cdots \oplus x_n$ can be computed by depth d threshold circuits of size $n^{1+O(1)^d}$. A matching $n^{1+\Omega(1)^d}$ lower bound is also given in Impagliazzo, Paturi, and Saks [IPS93].

4.1.4 Restrictions

To prove PRF lower bounds against B_2 and TC^0_d circuits, we need to define the notation of *restriction*. A *restriction* ρ is a mapping from input bits to $\{0, 1, \star\}$[21], where those bits mapped to $\star$, denoted by $\rho^{-1}(\star)$, are called *unfixed* or *free* bits. Let $f \in B_{n,m}$ be a Boolean function and $\rho : [n] \to \{0, 1, \star\}$ be a restriction. We can then define the restricted function $f\restriction_\rho \in B_{|\rho^{-1}(\star)|,m}$ as the function over unfixed bits obtained by fixing the i^{th} bit as $\rho(i)$ for each $i \in \rho^{-1}(\{0, 1\})$.

Definition 4.2 (Random restriction). Let n be the number of input bits. A *random p-restriction* (or *p-restriction*) is the following distribution $\mathcal{R}^n_p$ over all restrictions: independently for each input bit, we set it to $\star$ with probability p and to 0 and 1 with probability $(1 - p)/2$ each. ♢

During probabilistic arguments, it may be convenient to view a random p-restriction as a pair (S, y) of random variables, where S denotes the set of fixed bits and $y \in \mathbb{F}_2^{|S|}$ refers to the assignment to the fixed bits. Condition on a particular S, the distribution of the assignment y is uniformly chosen from $\mathbb{F}_2^{|S|}$.

4.2 Pseudorandom function

The syntax of pseudorandom functions consists of a collection of functions $F_n \subseteq B_n$ and a distribution $\mathcal{D}_n$ supported over F_n, both of which are labeled by the input length n. In this work, we assume that PRFs are defined for all input lengths. For convenience, we represent a PRF as $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$, implicitly keep the distribution $\mathcal{D}_n$ in mind and denote the sampling procedure simply by $f \leftarrow F_n$.

Definition 4.3 (Negligible functions). A function $\varepsilon(n)$ is called *negligible* if for all $c > 0$ and sufficiently large n, we have $\varepsilon(n) < n^{-c}$. We use the notation $\mathsf{negl}(n)$ to mean an arbitrary negligible functions w.r.t. n if there is no ambiguity. ♢

Definition 4.4 (Indistinguishability). Two function families $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$ and $\mathcal{G} = \{G_n \subseteq B_n\}_{n\geq 1}$ are *indistinguishable* if for all p.p.t. adversary $\mathcal{A}^O$ with oracle access to O, there exists a negligible function $\varepsilon(\cdot)$ such that

$$\left| \Pr_{f\leftarrow F_n, \mathcal{A}}[\mathcal{A}^f(1^n) = 1] - \Pr_{g\leftarrow G_n, \mathcal{A}}[\mathcal{A}^g(1^n) = 1] \right| \leq \varepsilon(n).$$

♢

[21] Or $\{1, -1, \star\}$ when we are working with threshold circuits.

One can easily verify that the indistinguishability relation is an equivalence relation, i.e., it is transitive, symmetric, and reflexive.

Definition 4.5 (Pseudorandom functions). A *pseudorandom function* (PRF) is a family $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$ that is indistinguishable from truly random function $\mathcal{B} = \{B_n\}_{n\geq 1}$. ♢

4.3 Hash functions and error-correcting codes

Let n be the input length. Similar to PRF, the syntax of a hash function is defined by a family of functions $\mathcal{H} = \{H_n \subseteq B_{n,m}\}$ and a family of distribution $\mathcal{D}_n$ supported over H_n. Again, we will omit the distribution and denote the sampling procedure by $h \leftarrow H_n$.

Definition 4.6 (Hash functions). Let $m = m(n)$ be a function, a *hash function* is a family $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$. It is called *universal* if for all n and $x \neq y \in \mathbb{F}_2^n$,

$$\Pr_{h\leftarrow H_n}[h(x) = h(y)] = 2^{-m}.$$

It is called *almost universal* if there exists a negligible function $\varepsilon(\cdot)$ such that for all n and distinct inputs $x, y \in \mathbb{F}_2^n$,

$$\Pr_{h\leftarrow H_n}[h(x) = h(y)] \leq \varepsilon(n).$$

♢

Definition 4.7 (Error-correcting codes). Let $m = m(n) > n$ be a function. A function family $E = \{\mathrm{Enc}_n \in B_{n,m}\}_{n\geq 1}$ is called an *error-correcting code* with relative distance $\delta \in (0, 1)$ if for sufficiently large n, for all $x \neq y \in \mathbb{F}_2^n$, the relative distance $\Delta_r(\mathrm{Enc}_n(x), \mathrm{Enc}_n(y))$ is at least δ. Moreover, E is called *systematic* if the encoding function can be interpreted as $\mathrm{Enc}_n(x) = x\|\mathrm{Par}_n(x)$, where the last $m - n$ bits generated by Par_n is called the *parity-checking bits*. ♢

There is a simple construction of almost universal hash functions from error-correcting codes, which is probably folklore. Let $0 < \varepsilon < 1$, $m = \Theta(n^\varepsilon)$ be the desired output length and $\mathrm{Enc}_n \in B_{n,m'}$ be an encoding function with relative distance δ. The hash function H_n^{Enc} is defined as the collection of all functions h_S indexed by subsets $S = \{i_1, i_2, \ldots, i_m\} \subseteq [m']$ of size exactly m, such that

$$h_S(x) \triangleq \mathrm{Enc}_n(x)_{i_1}\|\mathrm{Enc}_n(x)_{i_2}\|\ldots\|\mathrm{Enc}_n(x)_{i_m}.$$

That is, the hash function $\mathcal{H}^{\mathrm{Enc}}$ selects a random m-subset of the output of $\mathrm{Enc}_n(x)$. Clearly, this construction does not increase the circuit complexity of the function since we only need to relabel the output nodes.

Proposition 4.8. $\mathcal{H}^{\mathrm{Enc}} = \{H_n^{\mathrm{Enc}}\}_{n\geq 1}$ is almost universal. ♢

Proof. Let $x \neq y$ be distinct inputs of length n. By the distance property of the error-correcting code, $\mathrm{Enc}_n(x)$ and $\mathrm{Enc}_n(y)$ have Hamming distance at least $\delta m'$. The probability that $h(x) = h(y)$ given $h \leftarrow H_n^{\mathrm{Enc}}$ can be bounded by

$$\Pr_{h\leftarrow H_n^{\mathrm{Enc}}}[h(x) = h(y)] \leq \frac{\binom{(1-\delta)m'}{m}}{\binom{m'}{m}} = \prod_{0\leq i<m} \frac{(1-\delta)m' - i}{m' - i} \leq (1-\delta)^m.$$

Recall that we take $m = \Theta(n^\varepsilon)$, so this would be negligible whenever the relative distance δ of the error-correcting code is constant. □

4.4 Circuit complexity and uniformity

Both pseudorandom functions and hash functions are defined as families of function collections $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$, hence there are several ways to define the complexity of them. For example, one can define the complexity of $\mathcal{F}$ as the complexity to sample the distribution $\mathcal{D}_n$. In this paper, however, we define the complexity of $\mathcal{F}$ just as the maximum circuit complexity of $f \in F_n$.

Definition 4.9 (Circuit complexity of a function collection). Let $\mathcal{C}$ be a circuit class (for example, B_2 or TC^0_d). For a family of function collection $F = \{F_n \subseteq B_n\}_{n\geq 1}$, the $\mathcal{C}$-circuit complexity of $\mathcal{F}$ is defined as the size function $s(n) \triangleq \max_{f\in F_n} \mathcal{C}\text{-}\mathsf{CC}(f)$, where $\mathcal{C}\text{-}\mathsf{CC}(f)$ stands for the minimum size of $\mathcal{C}$-circuits to compute f. ♢

In default, pseudorandom functions and hash functions in this paper can be *non-uniform*, i.e., there is no requirement on the complexity of generating the circuits for a function $f \in F_n$ given the corresponding key. We can also define *uniform* counterparts of these primitives.

Definition 4.10 (Uniformity of collection). Let $\mathcal{C}$ be a circuit class. A family $\mathcal{F} = \{F_n\}_{n\in\mathbb{N}}$ (with sampling distribution $\mathcal{D}_n$ over F_n) is called a *uniform-$\mathcal{C}$* family if there exists a p.p.t. algorithm $\mathcal{G}$, such that for all n and $f \in F_n$,

$$\mathcal{D}(f) = \Pr_{\mathcal{G}}[\mathcal{G}(1^n) \text{ outputs a } \mathcal{C}\text{-circuit computing } f].$$

Similarly, it is called a *weakly uniform-$\mathcal{C}$* family if for all $d \in \mathbb{N}$, there exists a p.p.t. algorithm $\mathcal{G}$ and an event $\mathcal{E}$ (denoting whether $\mathcal{G}$ successes) such that for all n,

$$\Pr_{\mathcal{G}(1^n)}[\mathcal{E}] \geq 1 - \frac{1}{n^d},$$

and for all $f \in F_n$,

$$\mathcal{D}(f) = \Pr_{\mathcal{G}}[\mathcal{G}(1^n) \text{ outputs a } \mathcal{C}\text{-circuit computing } f \mid \mathcal{E}].$$

We say a collection has *uniform complexity* (or *weakly-uniform complexity*) $s(n)$, if it is uniform (or weakly uniform), and the generated circuits is of size at most $s(n)$ for sufficiently large n. ♢

We introduce the notion of *weak uniformity* mainly due to technical reasons. We require our primitives to have a certain degree of uniformity even when we consider non-uniform PRF, since the adversary is uniform. Intuitively, a family is weakly uniform if one can sample from the family with error probability n^{-d} for arbitrarily large d.

4.5 Levin's trick for domain extension

One of the key tools to prove circuit upper bounds for PRF is Levin's trick for domain extension. It shows that one can construct a PRF $F_n \subseteq B_n$ with a PRF $F'_m \subseteq B_m$ for $m = \Omega(n^\varepsilon)$ by hiding F' behind a uniform universal hash function. We will need a generalized version of Levin's trick by allowing the hash function to be almost universal and weakly uniform.

Lemma 4.11 (Levin's trick, generalized). Let $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$ be a PRF and $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ be an almost universal hash function of polynomial weakly-uniform complexity with $m = m(n) = \Theta(n^\varepsilon)$ for some absolute constant $0 < \varepsilon < 1$. The composition of $\mathcal{F}$ and $\mathcal{H}$, i.e., $\mathcal{F}' = \{F'_n\}_{n\geq 1}$ for $F'_n \triangleq \{f \circ h \mid f \in F_m, h \in H_n\}$, is also a PRF. ♢

Proof. Let $\mathcal{B}'$ be the composition of $\mathcal{B} = \{B_m\}_{m\geq 1}$ and $\mathcal{H}$, i.e., $B'_n = \{f \circ h \mid f \in B_m, h \in H_n\}$. It is known that $\mathcal{B}'$ is indistinguishable from truly random functions $\mathcal{B} = \{B_n\}_{n\geq 1}$ (for completeness, we present a proof in Appendix A). By transitivity, it suffices to show that $\mathcal{F}'$ is indistinguishable from $\mathcal{B}'$.

Towards a contradiction we assume that $\mathcal{F}'$ and $\mathcal{B}'$ are distinguishable, then there exists a p.p.t. adversary $\mathcal{A}$ such that there exists a constant c and infinitely many bad input length, say $n \in \{n_1, n_2, \dots\}$, such that

$$\left| \Pr_{f\leftarrow F'_n,\mathcal{A}}[\mathcal{A}^f(1^n) = 1] - \Pr_{f\leftarrow B'_n,\mathcal{A}}[\mathcal{A}^f(1^n) = 1] \right| > n^{-c}.$$

Now we construct a p.p.t. adversary that breaks the original PRF on inputs $m \in \{m(n_1), m(n_2), \dots\}$. By the (weakly) uniformity of $\mathcal{H}$, there exists a p.p.t. generator $\mathcal{G}$ and an event $\mathcal{E}$ (denoting whether $\mathcal{G}$ successes) such that $\mathcal{G}(1^n)$ successfully samples a hash function conditioning on $\mathcal{E}$, and $\Pr[\mathcal{E}] \geq 1 - n^{-(c+1)}$. Our adversary $\mathcal{A}'^f(1^m)$ samples such a circuit $C \leftarrow \mathcal{G}(1^n)$ and then simulates $\mathcal{A}^{f\circ C}(1^n)$. That is, whenever $\mathcal{A}$ performs an oracle call for x, it evaluate C on x and perform an oracle call for $C(x)$. Note that

$$\begin{aligned}
&\Pr_{f'\leftarrow F'_n,\mathcal{A}}[\mathcal{A}^{f'}(1^n) = 1] \\
=& \Pr_{\substack{f\leftarrow F_m \\ h\leftarrow H_n,\mathcal{A}}}[\mathcal{A}^{f\circ h}(1^n) = 1] \\
=& \Pr_{\substack{f\leftarrow F_m \\ \mathcal{A},C\leftarrow\mathcal{G}(1^n)}}[\mathcal{A}^{f\circ C}(1^n) = 1 \mid \mathcal{E}] \\
=& \Pr_{f\leftarrow F_m,\mathcal{A}'}[\mathcal{A}'^f(1^n) = 1 \mid \mathcal{E}].
\end{aligned}$$

Similarly, we have

$$\Pr_{f'\leftarrow B'_n,\mathcal{A}}[\mathcal{A}^{f'}(1^n) = 1] = \Pr_{f\leftarrow B_m,\mathcal{A}'}[\mathcal{A}'^f(1^n) = 1 \mid \mathcal{E}].$$

Together with the fact that $\Pr[\mathcal{E}] \geq 1 - n^{-(c+1)}$, we can see that for large n,

$$\begin{aligned}
&\left| \Pr_{f\leftarrow F_m,\mathcal{A}'}[\mathcal{A}'^f(1^m) = 1] - \Pr_{f\leftarrow B_m,\mathcal{A}'}[\mathcal{A}'^f(1^m) = 1] \right| \\
>& n^{-c}\Pr[\mathcal{E}] - \left| \Pr_{f\leftarrow F_m,\mathcal{A}'}[\mathcal{A}'^f(1^m) = 1 \mid \neg\mathcal{E}] - \Pr_{f\leftarrow B_m,\mathcal{A}'}[\mathcal{A}'^f(1^m) = 1 \mid \neg\mathcal{E}] \right| \Pr[\neg\mathcal{E}] \\
\geq& n^{-c}(1 - n^{-(c+1)}) - n^{-(c+1)} \\
\geq& n^{-(c+1)} \\
\geq& \Theta(m^{-\varepsilon^{-1}(c+1)}),
\end{aligned}$$

which is non-negligible. Hence $\mathcal{A}'$ breaks the original PRF and a contradiction arises. □

4.6 Probability theory

We need to use standard Hoeffding's inequality and Chernoff bound.

Lemma 4.12 (Hoeffding's inequality). Assume that $X_1, X_2, \ldots, X_n$ are independent random variables such that $X_i \in [a_i, b_i]$. Let $X = X_1 + X_2 + \cdots + X_n$ and $\mu = \mathbb{E}[X]$, then for any $t > 0$,

$$\Pr[|X - \mu| \geq \varepsilon n] \leq 2 \exp\left(-\frac{2n^2\varepsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right). \qquad \diamondsuit$$

Lemma 4.13 (Chernoff bound). Assume that $X_1, X_2, \ldots, X_n$ are independent random variables such that $X_i \in [0, 1]$. Let $X = X_1 + X_2 + \cdots + X_n$ and $\mu = \mathbb{E}[X]$, then for any $0 \leq \delta \leq 1$,

$$\Pr[|X - \mu| > \delta\mu] \leq 2 \exp\left(-\frac{\delta^2\mu}{3}\right). \qquad \diamondsuit$$

5 A $2n + o(n)$ upper bound for B_2 circuits

In this section, we will present a construction of PRFs in $2n + o(n)$ size B_2 circuits assuming the existence of PRFs. Our construction preserves the uniformity of the original PRF, and the key ingredient is a uniform construction of almost universal hash function in $2n + o(n)$ size B_2 circuits.

To gain more intuition on our construction, we will firstly demonstrate a direct $O(n)$ upper bound using the linear-size encodable error-correcting code [Spi96] in Section 5.1. Then in Section 5.2, we show that it is sufficient to construct a primitive called *1-detector* that is much simpler than error-correcting code and prove a $3n + o(n)$ construction of it. We improve the upper bound to $2n + o(n)$ using a novel construction of almost universal hash function based on graphs with large girth in Section 5.4.

5.1 A linear upper bound

We present a simplified version of $O(n)$ upper bound from Ishai, Kushilevitz, Ostrovsky, and Sahai [IKOS08] based on the following explicit error-correcting code with linear circuit complexity.

Lemma 5.1 (Spielman [Spi96]). There exists a uniform error-correcting code $\{\mathrm{Enc}_n \in B_{n,m}\}_{n\geq 1}$ of $O(n)$ gates with constant distance and $m = O(n)$. $\diamondsuit$

Theorem 5.2. There exists a PRF (resp. a uniform PRF) computable by B_2 circuits of size $O(n)$ if polynomial-size PRFs (resp. uniform PRFs) exist. $\diamondsuit$

Proof. Assume that there exists a PRF $\mathcal{F} = \{F_n \subseteq B_n\}$ of circuit complexity n^c. By Lemma 5.1, there exists a uniform error-correcting code $\{\mathrm{Enc}_n \in B_{n,m}\}_{n\geq 1}$ with distance $\delta \in (0, 1)$ of complexity $O(n)$. Using Proposition 4.8, we can construct a linear-size uniform almost universal hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ for $m = \lceil n^{1/2c} \rceil$. By Levin's trick (see Lemma 4.11), the concatenation of $\mathcal{F}$ and $\mathcal{H}$, say $\mathcal{F} \circ \mathcal{H}$, is still a PRF. The circuit complexity of $\mathcal{F} \circ \mathcal{H}$ is at most $O(n) + O(m^c) = O(n)$. In addition, it is easy to see that $\mathcal{F} \circ \mathcal{H}$ is uniform if $\mathcal{F}$ is uniform. $\square$

5.2 Constructing hash function from 1-detector

As seen from the above proof, if we have an almost universal hash function $\{H_n \in B_{n,m}\}_{n\geq 1}$ of size cn for $m = o(n)$, we can composite the hash function and the $O(n)$ PRF in Theorem 5.2 to construct a PRF of size $cn + o(n)$. We note that the exact constant hidden in $O(\cdot)$ for the construction above

is hard to analyze because the construction in [Spi96] utilizes explicit expanders and brute-force searched good codes, whose exact parameters are not well-studied.

This hints us to explore a "low-level" primitive with smaller complexity for our application instead of directly using an ECC. Such a notion indeed exists, which we call *1-detector*. We first formally define this concept and show that it implies almost universal hash functions in this section. In the following subsections, we will show how to construct them efficiently. In particular, we will first give a $3n + o(n)$ construction and then a novel construction of $2n + o(n)$ from graphs with large girth.

Definition 5.3 (1-detector). Let $m = m(n)$ and $r = r(n)$. An (n, r, m) *1-detector* is a *linear* function $L_n \in B_{n,m}$ such that for all $x \in \mathbb{F}_2^n$ with Hamming weight $|x| \leq r$, $L_n(x) \neq 0$. A family of linear functions $\mathcal{L} = \{L_n \in B_{n,m}\}_{n\geq 1}$ is called a (r, m) 1-detector if L_n is (n, r, m) 1-detector for all n. The output bits $L_n(x)$ are called the *parity-checking bits* of x. ♢

Since 1-detectors are relatively hard to construct in small size circuits, we can slacken the definition to allow the 1-detectors to be *randomized*.

Definition 5.4 (Randomized 1-detector). Let $m = m(n)$, $r = r(n)$ and $\varepsilon = \varepsilon(n)$. An (n, r, m, ε) *randomized 1-detector* is a linear function $L_n \in B_{n,m}$ such that for all $x \in \mathbb{F}_2^n$ with Hamming weight $|x| \leq r$, $\Pr_\rho[L_n(\rho(x)) = 0] < \varepsilon(n)$ for a random permutation ρ over input bits. For simplicity, we may omit ε if it is a negligible function and simply call it (n, r, m) randomized 1-detector. A family of linear functions $\mathcal{L} = \{L_n \in B_{n,m}\}_{n\geq 1}$ is called a (r, m) randomized 1-detector if L_n is (n, r, m) randomized 1-detector for all n. The output bits $L_n(\rho(x))$ are called the *parity-checking bits* of x. ♢

Note that a (n, r, m) (deterministic) 1-detector is also an (n, r, m, ε) randomized 1-detector with $\varepsilon = 0$.

Let $\mathcal{L} = \{L_n \in B_{n,m}\}_{n\geq 1}$ be an (r, m) randomized 1-detector[22]. By the linearity, we can see that for all $x, y \in \mathbb{F}_2^n$ such that $1 \leq \Delta(x, y) \leq r$, $L_n(\rho(x)) = L_n(\rho(y))$ with only negligible probability. If $m = o(n)$ and r is moderately large, say $r = \Theta(n^\varepsilon)$, we can construct an almost universal hash function as follows. Let $\rho \in \mathcal{S}_n$ be a permutation over input bits and let $S \subseteq [n]$ be a subset of size $|S| = s = \Theta(n^{1-\varepsilon/2})$. For $S = \{i_1, i_2, \ldots, i_s\}$ we can define

$$h_{\rho,S}(x) \triangleq x_{i_1} \| x_{i_2} \| \ldots \| x_{i_s} \| L_n(\rho(x)),$$

that is the concatenation of s random bits from x and the parity-checking bits.

Lemma 5.5. Assume that $s = \omega(n \log n)/r$. If $\mathcal{L}$ is an (r, m) randomized 1-detector, the collection $\mathcal{H}^{\mathcal{L}} = \{H_n = \{h_{\rho,S} \mid \rho \in \mathcal{S}_n, |S| = s, S \subseteq [n]\}\}_{n\geq 1}$ is almost universal. ♢

Proof. Let $x, y \in \mathbb{F}_2^n$ be an arbitrary pair of distinct inputs. If $\Delta(x, y) \leq r$, the probability that the last m bits of $h_{\rho,S}(x)$ and $h_{\rho,S}(y)$ coincide is less than $\varepsilon(n)$ according to randomized 1-detector. If $\Delta(x, y) > r$, the probability that the first s bits of $h_{\rho,S}(x)$ and $h_{\rho,S}(y)$ are the same is at most

$$\frac{\binom{n-r}{s}}{\binom{n}{s}} \leq \prod_{i=0}^{s-1} \frac{n-r-i}{n-i} \leq \left(1 - \frac{r}{n}\right)^s \leq \exp\left(-\frac{rs}{n}\right),$$

which is negligible since $rs/n = \omega(\log n)$. □

[22] Indeed we will construct a (non-uniform) $3n$-gate computable deterministic 1-detector with $r = \Theta(n^\varepsilon)$ and $s = \Theta(n^{1-\varepsilon/2})$ for $\varepsilon \in (0, 1)$ in Section 5.3 and a $2n$-gate computable randomized 1-detector with $r = n/2$ and $m = \Theta(n/\log n)$ in Section 5.4.

The existence of (deterministic) 1-detectors with nice parameters has been known for a long time. Gelfand, Dobrushin, and Pinsker [GDP73] present a construction of ECC using 1-detector (although they did not name it) with slightly different parameters. A primitive called *range detector* used by Gál, Hansen, Koucký, Pudlák, and Viola [GHKPV13] is a generalized version of 1-detector. The intermediate primitive called *error-reduction code* of Spielman's ECC [Spi96] also has the property of 1-detection. However, these constructions are either not constructive or of circuit complexity larger than $3n$, therefore they are insufficient for our use.

5.3 A simple probabilistic construction

In this section, we will present a construction of 1-detectors inspired by the standard existence proof with the probabilistic method. Concretely speaking, for each positive integer k, we will give a p.p.t. algorithm $\mathcal{G}_k$ that outputs a $3n$ size circuit computing an (n, m, r) (deterministic) 1-detector with probability[23] at least $1 - n^{-\Omega(k)}$. One may see that it is sufficient for constructing PRF using Lemma 5.5 and Levin's trick.

The evaluation circuit generated by $\mathcal{G}_k$ is a $\mathsf{CC}^0_1[2]$ circuit containing m XOR gates of unbounded fan-in while each variable is of out-degree exactly $d \geq 3$, which can be transformed into a standard B_2 circuit with $d \cdot n$ gates. The underlying topology between variables and gates are defined by a bipartite graph $G = (V_1 \cup V_2, E \subseteq V_1 \times V_2)$, where $|V_1| = n$ and $|V_2| = m = m(n)$. The evaluation circuit C_G corresponding to a graph $G = (V_1 \cup V_2, E)$ is a depth-1 linear circuit: each vertex in V_1 corresponds to an input variable, each vertex in V_2 corresponds to an XOR gate, and the connection between gates and variables follows the edges of the graph. A bipartite graph G is called *good* if C_G computes an (n, r, m) 1-detector. Equivalently, a graph is good if for any subset $S \subseteq V_1$ of size at most r, at least one of variable $v' \in V_2$ connects to odd number of variables in S. For a typical choice of parameters, we assume that $r = \Theta(n^{\varepsilon})$ and $m = \Theta(n^{1-\varepsilon/2})$ for some constant $\varepsilon \in (0, 1)$.

To complete our algorithm $\mathcal{G}_k$, it suffices to describe an algorithm that generates a good graph with nice probability. By adopting the random procedure defined by [GDP73] together with an error reduction trick, we can actually design Algorithm 1 running in time $n^{O(k)}$ that generates a good graph with probability $n^{-0.1k}$, for any positive integer k.

Algorithm 1: Generating good graphs

```
for i = 1, 2, ..., t do
    Let G ← (V_1 ∪ V_2, ∅) be an empty graph;
    for v ∈ V_1, j = 1, 2, ..., d do
        Link a random edge e_{v,j} = (v, v') with v' ← V_2;
    end
    if ∀S ⊆ V_1 of size ≤ k, there exists v' ∈ V_2 connecting to odd number of vertices in S then
        return G;
    end
end
return ⊥
```

[23] On a fail execution, our algorithm may output an arbitrary circuit (or simply $\perp$) without any additional information, so there is no obvious way to amplify success probability. We note that it is similar (but not precisely equivalent) to the concept of weak uniformity: our algorithm will output a 1-detector on successful execution, but it is not guaranteed to output a particular one for all successful executions.

Lemma 5.6. Let $t = \omega(\log n)$ and $d \geq 3$. There exists a constant $\varepsilon \in (0,1)$, such that for $r = \Theta(n^{\varepsilon})$ and $m = \Theta(n^{1-\varepsilon/2})$, with probability at least $1 - n^{-0.1k}$, Algorithm 1 generates a good graph (and therefore a 1-detector) for sufficiently large n. ♢

Corollary 5.7. For any constant $\varepsilon \in (0,1)$, let $m = m(n) = \Theta(n^{\varepsilon})$, there exists an almost universal hash function $\mathcal{H} = \{H_{n,m} \subseteq B_{n,m}\}_{n \geq 1}$ with weakly uniform complexity $3n$. ♢

Since the proofs of Lemma 5.6 and Corollary 5.7 are technical and the construction in Section 5.4 will have better circuit complexity[24], we defer them to the Appendix B. The intuition of the corollary is that, if we take $d = 3$ in Lemma 5.6, the generated $\mathsf{CC}_1^0[2]$ circuit can be transformed into a B_2 circuit of size $3n$ by expanding each XOR gate independently. Then we can construct a desired hash function using Lemma 5.5. By this corollary, a $3n + o(n)$ upper bound directly follows using Levin's trick (Lemma 4.11).

Theorem 5.8. There exists a PRF of circuit complexity $3n + o(n)$ assuming PRF exists. ♢

5.4 Better 1-detectors from high-girth graphs

We will now present a randomized 1-detector which is realizable by a $\mathsf{CC}_1^0[1]$ circuit C where each variable is of out-degree exactly 2. Let $G = (V_1 \cup V_2, E \subseteq V_1 \times V_2)$ be the graph indicating the topology of C. To further improve the construction in the previous subsection, we would like to inspect graphs with fine structures instead of picking random graphs. Since each input variable is of out-degree 2, graph G can be viewed as the edge-vertex incidence relation of another undirected graph D, such that each vertex $u \in V_2$ corresponds to a vertex $\widetilde{u}$ in the new graph D and each vertex in $v \in V_1$ adjacent to $u_1, u_2 \in V_2$ corresponds to an edge connecting $\widetilde{u_1}$ and $\widetilde{u_2}$. In such case, an assignment $x \in \mathbb{F}_2^n$ of the input bits such that $C(x) = 0$ corresponds to a subset X of edges in D such that each vertex is incident to an even number of edges in X. Thus, X contains an Eulerian cycle and therefore a cycle, which means, intuitively, if we want to construct a nice 1-detector, the graph D cannot contain small cycles.

By convention, the *girth* of a graph is defined as the length of its shortest cycle. The following lemma shows the connection between good graphs for randomized 1-detector and the girth of graphs.

Lemma 5.9. Let $D = (V, E)$ be a graph of girth $g \geq 5$ and let $S \subseteq E$ be a random subset of size k. For all $1 \leq k < |E|/2$, with probability at most $(|E|/g)^{-g/3}$, every vertex is incident to an even number of edges in S. ♢

Proof. Let $D = (V, E)$ be a graph and $S \subseteq E$ be a subset of size k. If all vertices connect to an even number of edges in S, each vertex in the subgraph $D' = (V, S)$ is of even degree. In such a case, any connected component of D' consists of an Eulerian cycle, which can only happen if G contains a cycle of length no more than $|S|$. So if $k < g$, there always exists a vertex that is incident to an odd number of edges in S.

Now we consider the case when $k \geq g$. For a random subset S of size k, we can view it as first taking a random subset of size $k - \lceil g/3 \rceil$, then another random subset of size $\lceil g/3 \rceil$ in the

[24] We should also note, however, that the collision probability of this $3n$ size hash function could be much better than the construction in Section 5.4. This may make it of independent interest.

remaining edges. Let $\mathcal{L}(S)$ be the event that every vertex is incident to an even number of edges in S, then

$$\begin{aligned}\Pr_{S\subseteq E,|S|=k}[\mathcal{L}(S)] &= \mathop{\mathbb{E}}_{S_1\subseteq E,|S_1|=k-\lceil g/3\rceil}\left[\Pr_{S_2\subseteq E\setminus S_1,|S_2|=\lceil g/3\rceil}[\mathcal{L}(S_1\cup S_2)]\right] \\ &= \mathop{\mathbb{E}}_{S_1\subseteq E,|S_1|=k-\lceil g/3\rceil}\left[\binom{|E\setminus S_1|}{\lceil g/3\rceil}^{-1}\sum_{S_2\subseteq E\setminus S_1,|S_2|=\lceil g/3\rceil}[\mathcal{L}(S_1\cup S_2)]\right].\end{aligned}$$

We will show the summation in the above equation does not exceed 1, that is for any fixed $S_1 \subseteq E$ with $|S_1| = k - \lceil g/3\rceil$, there exists at most one S_2 makes $\mathcal{L}(S_1 \cup S_2)$ happens. If this is true, then we can clearly bound the above probability by

$$\Pr_{S\subseteq E,|S|=k}[\mathcal{L}(S)] \le \binom{|E|-(k-\lceil g/3\rceil)}{\lceil g/3\rceil}^{-1} \le \left(\frac{|E|}{g}\right)^{-g/3}.$$

What remains is the claim above. Towards a contradiction, assume $\mathcal{L}(S_1 \cup S_2)$ and $\mathcal{L}(S_1 \cup S_2')$ holds simultaneously for $S_2 \neq S_2'$. Consider the symmetric difference of the two sets $S_2 \oplus S_2' = (S_1 \cup S_2) \oplus (S_1 \cup S_2')$. By our definition of the event $\mathcal{L}$, each vertex is incident to even number of edges in $S_2 \oplus S_2'$, resulting in a cycle of length no more than $|S_2 \oplus S_2'| \le |S_2| + |S_2'| = 2\lceil g/3\rceil < g$. This contradicts to the fact that the girth of G is g. □

This lemma shows that if we construct a circuit C based on the edge-vertex incident graph of an undirected graph $D = (V, E)$ with $|V| = m$, $|E| = n$ and girth $g = \omega(1)$, for an assignment x with Hamming weight $|x| \le n/2$ and a random permutation ρ of input bits, $C(\rho(x)) = 0$ with only negligible probability. If the graph D is moderately dense, say $m = \Theta(n/\log n)$, we can obtain a $\mathsf{CC}^0_1[2]$ circuit computing an $(n, n/2, m)$ randomized 1-detector and therefore an almost universal hash function by Lemma 5.5.

What remains is the explicit construction of graphs with large girth. This problem has been studied in combinatorics since the 1960s, motivated by both theoretical interests and the practical issue to construct efficient low-density parity-checking (LDPC) codes. For our application, it is sufficient to use the simple construction given by Chandran [Cha03].

Lemma 5.10 ([Cha03]). There exists a polynomial-time algorithm such that given any m and $k < m/3$, constructs a graph $G = (V, E)$ of m vertices with $|E| = \lfloor mk/2\rfloor$ such that the girth of the graph is at least $g > \log_k m + O(1)$. Moreover, the degree of each vertex is $k-1, k$ or $k+1$. ♢

Corollary 5.11. For every n and m such that $\lceil 2n/m\rceil < m/3$, there exists a graph $D_{m,n}$ with m vertices and n edges where the girth $g > \frac{\log m}{\log(\lceil 2n/m\rceil)} + O(1)$. The degree of every vertex is at most $2n/m + O(1)$. Moreover, there exists a deterministic polynomial time algorithm construct $D_{m,n}$ taking m, n. ♢

Proof. Assume that we are given n and m. Let $k = \lceil 2n/m\rceil$, our algorithm firstly calls the algorithm in Lemma 5.10 to generate a graph $D = (V, E)$ with $|V| = m$, $|E| = \lfloor mk/2\rfloor \ge n$ and girth $g > \log_k(m) + O(1)$. The degree of each vertex is at most $2n/m + 2$. We can arbitrarily remove $|E| - n$ edges to obtain a desired graph. □

Now we formally describe the construction of our uniform randomized 1-detector and hash function. Take $m = \Theta(n/\log n)$. We firstly construct a graph $D_{m,n}$ with girth $g = \Omega\left(\frac{\log m}{\log(2n/m)}\right) =$

$\Omega\left(\frac{\log n}{\log\log n}\right)$. Then, we construct bipartite graph G using $D_{m,n}$ and then generate the $\mathsf{CC}^0_1[2]$ circuit C according to it. By Lemma 5.9 and previous discussion, one can easily see that C is an $(n, n/2, n/\log n)$ randomized 1-detector and can be transformed into B_2 circuit of size $2n-m$. The circuit family is hence an $(n/2, n/\log n)$ randomized 1-detector. Finally, we take $s = \Theta(\log^2 n)$ and construct an almost universal hash function with sufficient shrinkage using the sampling trick in Lemma 5.5.

Theorem 5.12. There exists a uniform almost universal hash function $\mathcal{H} = \{H_{n,m} \subseteq B_{n,m}\}_{n\geq 1}$ of circuit size $2n-m$, where $m = \Theta(n/\log n)$. ♢

We can then compose this hash function with the simple $O(n)$ PRF in Theorem 5.2 by Levin's trick (see Lemma 4.11) and give the following upper bound of PRF.

Corollary 5.13. There exists a PRF (resp. uniform PRF) of circuit complexity $2n + o(n)$ if PRF (resp. uniform PRF) exists. ♢

6 Upper bounds in low-depth classes

Besides circuit size (or running time in uniform case), circuit depth (or parallel time) is also an essential measure with theoretical and practical interests. Many efforts have been made to study the existence of PRF in low-depth circuit classes such as NC^1, TC^0, and even $\mathsf{AC}^0[2]$. However, it remains open whether we can build PRFs that are efficient in both size and depth complexity simultaneously.

In this section, we will show that some slight modifications of our PRF based on high-girth graphs would yield efficiency in both size and depth. In Section 6.1, we will first briefly present some known candidates of PRFs in low-complexity classes from well-founded cryptographic assumptions as the foundation of our constructions. In Section 6.2, we will construct an NC^1 PRF of size $2n + o(n)$ and depth $(1+\varepsilon)\log n$ for any $\varepsilon > 0$ using a simple stacking argument. In Section 6.3, we will construct an efficient TC^0 PRF using the error-correcting code of Chen and Tell [CT19]. In Section 6.4, we will construct an almost universal hash function with arbitrary polynomial shrinkage in $\mathsf{CC}^0[2]$ with depth-2 and wire complexity $2n + o(n)$ by combining our high-girth based 1-detector with the efficient error-correcting codes from [CT19], which further leads to an $\mathsf{AC}^0[2]$ PRF with $o(n)$ gates and $2n + o(n)$ wires.

6.1 Candidate PRFs in low-depth classes

Although it is unknown whether the existence of low-depth PRFs (say, in TC^0) can be based on the elementary primitives such as one-way functions, there are several constructions under standard cryptographic assumptions like factoring (of Blum-integers) or Decisional Diffie-Hellman [NR04], as well as Ring Learning-with-Error [BPR12].

We now sketch the construction of Naor and Reingold [NR04] based on the Decisional Diffie-Hellman assumption and explain how it can be implemented in NC^1, TC^0 and $\mathsf{AC}^0[2]$ circuits of polynomial size. Let n be desired input length. The key of the PRF is a tuple (p, q, g, a), where p is an n-bit prime, q is a prime dividing $p-1$, g is an element in $\mathbb{Z}_p^\times$ with order $q > 2^n$ and $a \in \mathbb{Z}_q^{n+1}$. Note that (p, q, g) is chosen over some polynomially samplable distribution and a is chosen uniformly. Let $x = x_1 \| x_2 \| \ldots \| x_n$. The output of the PRF is defined as

$$f_{p,q,g,a}(x) \triangleq (g^{a_0})^{\prod_{i=1}^n a_i^{x_i}}.$$

The DDH-assumption. The security of this PRF candidate follows from the Decisional version of Diffie-Hellman assumption, which states that it is intractable to distinguish (g^a, g^b, g^{ab}) from (g^a, g^b, g^c) for random a, b and c, where g is a generator of certain cyclic group G. In this candidate, we choose G as the multiplicative group of the finite field, in which the state-of-the-art cryptoanalysis algorithm works in $2^{O(n^{1/3})}$ time [Gor93]. It is widely believed that the DDH-assumption (in finite fields) is secure against all 2^{n^ε}-time adversaries for some $\varepsilon > 0$.

Implementation in TC^0 (and therefore NC^1). Naor and Reingold [NR04] showed that this candidate can be implemented in TC^0 (and therefore NC^1 since $\mathsf{TC}^0 \subseteq \mathsf{NC}^1$) based on the result of Reif and Tate [RT92], which shows that *multiple product* $\prod_i a_i^{x_i}$ can be evaluated in TC^0. The candidate function is computed in two steps. Firstly, we compute $a_0 \prod_i a_i^{x_i}$ with the multiple product scheme and obtain its binary form $y = \overline{y_{m-1} \dots y_1 y_0}$. Let $b_i = g^{2^i}$ be hard-wired parameters. We can again use the multiple product scheme to compute $\prod_i b_i^{y_i}$, which is exactly what we need.

Implementation in $\mathsf{AC}^0[2]$. Viola [Vio15] gives a simple construction of $\mathsf{AC}^0[2]$ PRF secure against $2^{\mathsf{polylog} n}$-time adversary from an 2^{n^ε}-time secure TC^0 PRF for some $\varepsilon > 0$, whose existence follows from the 2^{n^ε} hard DDH-assumption. Assume that such TC^0 PRF exists. We can hash down the input length to $\mathsf{polylog}(n)$ by Levin's trick (see Lemma 4.11) using a hash in $\mathsf{AC}^0[2]$[25]. By a folklore result that the majority gate of fan-in m can be realized by AC^0 circuits of size $2^{\tilde{O}(m^{1/(d-1)})}$ in depth d (see, e.g., [OSS19]), the TC^0 PRF with $\mathsf{polylog}(n)$ input length can be converted into a polynomial size $\mathsf{AC}^0[2]$ circuit.

6.2 Construction of NC^1 PRFs via stacking

We start with an uniform construction of an almost universal hash function of size $2n$ and depth $(1 + o(1)) \log n$ that shrinks n-bit input to n^ε-bit output for arbitrary $\varepsilon > 0$. This is achieved by stacking the hash functions in Theorem 5.12 for $o(\log n)$ number of times.

Lemma 6.1. For any $\varepsilon > 0$, there exists a uniform construction of almost universal hash function $\mathcal{H} = \{H_n \in B_{n,c}\}_{n \geq 1}$ of size $2n + o(n)$ and depth $(1 + o(1)) \log n$ for some $c = c(n) < n^\varepsilon$. ♢

Proof. Let $m = m(n)$ be the function in Theorem 5.12. In Section 5.4 we have present a uniform construction of $(n, n/2, m)$ randomized 1-detector based on graph with large girth. Such 1-detector can be evaluated by a $\mathsf{CC}^0_1[2]$ circuit C_n whose topology is induced from the edge-vertex incident graph of $D_{m,n} = (V, E)$ given by Corollary 5.11. Note that the degree of each vertex in $D_{m,n}$ is at most $2n/m + O(1)$, so that each XOR gate in C is of fan-in $\Theta(n/m)$. This means that C_n can be realized by a B_2 circuit of size at most $2n$ and depth $\log(n/m) + O(1)$.

Now we will construct a hash function $\mathcal{H} = \{H_n \subseteq B_{n,c}\}_{n \geq 1}$ for arbitrary $c < n^\varepsilon$ by stacking the hash function $\widetilde{\mathcal{H}} = \{\widetilde{H}_n \in B_{n,m}\}_{n \geq 1}$ in Theorem 5.12. Define $n_0, n_1, \dots, n_\ell$ by $n_0 = n$ and $n_{i+1} = m(n_i)$ for all $0 \leq i < \ell$, where ℓ is the smallest integer such that $n_\ell < n^\varepsilon$. It is easy to see that $\ell = O(\log n)$. Then we define

$$H_n \triangleq \{h_{\ell-1} \circ h_{\ell-2} \circ \cdots \circ h_0 \mid h_i \in \widetilde{H}_{n_i}\},$$

where each h_i is sampled independently from $\widetilde{H}_{n_i}$.

[25] In the work of Viola [Vio15], a simple hash function is used without optimizing its concrete complexity. We will give an efficient construction of hash function in Section 6.4 to obtain our PRF upper bounds.

Since h_i can be computed by a circuit of $2n_i$ size and $\log(n_i/n_{i+1}) + O(1)$ depth, it is easy to verify that $\mathcal{H}$ can be computed by a circuit of size $2n + o(n)$ and depth $(1 + o(1)) \log n$. Now it is sufficient to show that $\mathcal{H}$ is indeed an almost universal hash function. Because $\widetilde{\mathcal{H}}$ is almost universal, there exists a negligible function $\varepsilon(n)$ such that for all n and $x \neq y$ of length n, $\Pr[h(x) = h(y)] \leq \varepsilon(n)$ for $h \leftarrow \widetilde{H}_n$. Then we can see that for all $x \neq y$ of length n,

$$\begin{aligned}
&\Pr_{h \leftarrow H_n}[h(x) = h(y)] \\
&= \Pr_{h=(h_0,\ldots,h_\ell) \leftarrow H_n}[\exists i, h_i \text{ is the first layer with same output for } x \text{ and } y] \\
&\leq \sum_{i=1}^{\ell-1} \varepsilon(n_i),
\end{aligned}$$

which is also negligible. This completes the proof. □

Then the upper bound of PRFs in NC^1 follows directly from Levin's trick (Lemma 4.11).

Corollary 6.2. For any $\varepsilon > 0$, there exists an NC^1 PRF (resp. a uniform NC^1 PRF) of $2n + o(n)$ size and $(1 + \varepsilon) \log n$ depth assuming NC^1 PRF (resp. uniform NC^1 PRF) exists. ♢

Proof. Suppose that there exists a PRF computable by a circuit of depth $d \log n$ and size n^d. By Lemma 4.11, we can compose it with a hash function that shrinks an n-bit input to $n^{\varepsilon/(2d)}$ bits to obtain a PRF with $2n + o(n)$ size and $(1 + \varepsilon) \log n$ depth, where $\varepsilon > 0$ can be arbitrarily small. □

6.3 Construction of TC^0 PRFs from efficient ECC

To obtain an upper bound for PRFs in TC^0, we only need to construct an efficient TC^0 almost universal hash function with nice shrinkage. Fortunately, we can directly use the error-correcting code given by Chen and Tell [CT19].

Lemma 6.3 ([CT19], Proposition 10). Let $\phi = \frac{1+\sqrt{5}}{2}$ and $c > 0$ be an absolute constant. For every $d \geq 4$ there exists a family of depth-d TC^0 circuits of $O(n^{1+c\phi^{-d}})$ wires that encodes an linear error-correcting code of constant relative distance. Moreover, this circuit family is uniformly constructible in polynomial time. ♢

This immediately shows that for any $d \geq 4$, there exists a uniform almost universal hash function realizable by TC^0_d circuits of $n^{1+O(\phi^{-d})}$ size that shrinks an n-bit input to $m = \Theta(n^\varepsilon)$ bits for arbitrarily small $\varepsilon > 0$ (see Proposition 4.8). Then by Levin's trick (see Lemma 4.11), we can obtain an upper bound for PRFs.

Corollary 6.4. Let $\phi = \frac{1+\sqrt{5}}{2}$ and $c > 0$ be an absolute constant. For any $d \geq 4$ and $\varepsilon > 0$, there exists a uniform almost universal hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n \geq 1}$ computable in TC^0_d circuits of $O(n^{1+c\phi^{-d}})$ wires, where $m = m(n) = \Theta(n^\varepsilon)$. ♢

Corollary 6.5. Let $\phi = \frac{1+\sqrt{5}}{2}$ and $c > 0$ be an absolute constant. For any constant d_0, assume that there exists a PRF (resp. uniform PRF) computable in $\mathsf{TC}^0_{d_0}$, then there exists a PRF (resp. uniform PRF) computable by TC^0 circuits of $d + d_0$ depth and $O(n^{1+c\phi^{-d}})$ wires, for any $d \geq 4$. ♢

6.4 Construction of $\mathsf{AC}^0[2]$ PRFs from optimally sparse hash function in $\mathsf{CC}^0[2]$

In this section, we will construct an $\mathsf{AC}^0[2]$ PRF with $o(n)$ gates and $2n + o(n)$ wires using an extremely hash function within $\mathsf{CC}^0[2]$, the class of constant-depth circuits with only unbounded fan-in XOR gates.

Theorem 6.6. Let $0 < \varepsilon < 1$ be any constant and $m = m(n) = \Theta(n^\varepsilon)$ be the output length, there exists a uniform almost universal hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ with $2n + o(n)$ wires and $o(n)$ gates in $\mathsf{CC}^0_3[2]$. ♢

Our plan is to compose two hash functions in $\mathsf{CC}^0[2]$ with different features: the first layer is a high-girth-graph-based construction (see Section 5.4) with wire complexity $2n + o(n)$ and poly-logarithmic shrinkage, which can ensure that overall complexity; and the second layer is based on the sparse error-correcting code in [CT19] that provides polynomial shrinkage. We first introduce the error-correcting code used by the second layer of our hash function.

Theorem 6.7 ([CT19], Proposition 31). Let c be an absolute constant. There exists a family of $\mathsf{CC}^0_2[2]$ circuits with wire complexity $O(n \exp((\log\log n)^c)$ that encodes an error-correcting codes of constant relative distance. Moreover, this family is uniformly constructible in polynomial time. ♢

Corollary 6.8. Let c be the constant in Theorem 6.7. For all constant $\varepsilon \in (0,1)$ and output length $m = m(n) = \Theta(n^\varepsilon)$, there exists a uniform almost universal hash function $\mathcal{H}^1 = \{H^1_{n,m} \subseteq B_{n,m}\}_{n\geq 1}$ with $O(n \cdot \exp((\log\log n)^c))$ wires and m gates in $\mathsf{CC}^0_2[2]$. ♢

Proof. This directly follows from Theorem 6.7 and Proposition 4.8. □

To construct a linear-size almost universal hash function in $\mathsf{CC}^0[2]$, we would hide the logarithmic factor in Corollary 6.8 by composing it with a variant of construction in Section 5.4.

Lemma 6.9. For any constant $c > 1$, there exists a uniform almost universal hash function $\mathcal{H}^2 = \{H^2_{n,m} \subseteq B_{n,m}\}_{n\geq 1}$ in $\mathsf{CC}^0_1[2]$ with at most $2n$ wires and m gates, where $m = m(n) = \Theta(n \cdot \exp(-(\log\log n)^{c+1}))$. ♢

Proof. According to Corollary 5.11, we first generate a graph $D_{m,n}$ with girth $g = \Omega\left(\frac{\log m}{\log(2n/m)}\right) = \Omega\left(\frac{\log n}{(\log\log n)^{c+1}}\right)$ and construct an bipartite graph G as the edge-vertex incident graph of $D_{m,n}$. Let C be the $\mathsf{CC}^0_1[2]$ circuit with $2n$ wires and m gates whose topology is given by the graph G. By Lemma 5.9, it is an $(n, n/2, m)$ randomized 1-detector. The circuit family is hence an $(n/2, n/\log n)$ randomized 1-detector. Finally, we take $s = \Theta(\log^2 n)$ to construct an almost universal hash function with the same number of wires and gates using Lemma 5.5. □

Proof of Theorem 6.6. Let $H^1_{n,k}$ be the hash function of input length n and output length k in Corollary 6.8, and $H^2_{k,m}$ be the hash function of input length k and output length m in Lemma 6.9, where $k = \Theta(n \cdot \exp(-(\log\log n)^{c+1}))$. We construct a hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ where

$$H_n \triangleq \{h_2 \circ h_1 | h_1 \in H^1_{n,k}, h_2 \in H^2_{k,m}\}.$$

This is clearly almost universal and uniform. Now we consider the circuit complexity. The first layer of the circuit requires $2n$ wires and $k = o(n)$ gates, and the second layer of the circuit requires

$$O(k \cdot \exp((\log\log k)^c)) = O(n \cdot \exp(-(\log\log n)^{c+1} + (\log\log k)^c)) = O(n/\log n) = o(n)$$

wires and $o(n)$ gates. Therefore, $\mathcal{H}$ can be computed by $\mathsf{CC}^0_3[2]$ circuits with $2n + o(n)$ wires and $o(n)$ gates. □

Using Levin's trick (see Lemma 4.11) with the hash function in Theorem 6.6, we get a PRF upper bound in $\mathsf{AC}^0[2]$ circuits.

Corollary 6.10. If there exist PRFs (resp. uniform PRFs) in $\mathsf{AC}^0_d[2]$, then there exists a PRF (resp. a uniform PRF) in $\mathsf{AC}^0_{d+3}[2]$ with $o(n)$ gates and $2n + o(n)$ wires. ♢

7 Lower bounds against B_2 and $\mathsf{AC}^0[2]$ circuits

In this section, we will prove several lower bounds showing that our constructions of PRFs and hash functions are near optimal (with respect to size or wire complexity) in B_2 circuits, NC^1 circuits and $\mathsf{AC}^0[2]$ circuits. In Section 7.1, we will present our $2n - O(1)$ PRF lower bound for B_2 circuits[26] using a clever distinguishing algorithm based on a wire-counting argument. In Section 7.2, we will adapt our technique to prove an unconditional lower bound for hash function. In Section 7.3, we will prove tight lower bound on the wire complexity of $\mathsf{AC}^0[2]$ in computing PRFs and hash functions.

7.1 A PRF lower bound for B_2 circuits

Our proof first studies a combinatorial structure on circuits, which we call *critical path*. We present an efficient algorithm that distinguishes circuits with intersecting critical paths and truly random functions via oracle access. Then we do a standard wire counting argument to show that $2n - O(1)$ gates are required to avoid intersecting critical paths, which leads to a circuit lower bound for PRF.

We begin by defining the combinatorial structure to be interested.

Definition 7.1 (Critical path). Let C be a circuit, and x be one of its inputs. The *critical path of* x in C is a sequence of vertices $v_0, v_1, \ldots, v_k$ satisfying the following conditions:

1. $v_0 = x$, and v_i is a descendent of v_{i-1} for all $i \geq 1$, and
2. out-degree$(v_i) = 1$ for all $0 \leq i < k$, and out-degree$(v_k) \neq 1$. ♢

Without loss of generality, we can only deal with circuits without obvious redundancy. Formally, a circuit C is called *normalized* if each gate of out-degree 0 is an output gate. Since a non-output gate of out-degree 0 can be removed, any optimal circuit computing a function f must be normalized.

Informally speaking, the *critical path of x in C* is the maximal path starting from x such that all but the last vertices have out degree exactly 1. The last vertex may have out-degree 0 or more than 2. It is obvious that the critical path is unique for each input x, so we denote it by $\mathcal{L}_C(x)$. We will be interested in the intersection of critical paths. Two critical paths are called intersecting, if they share a common vertex. We emphasize here that sharing the last vertex of out-degree not 1 is also called intersecting.

Our key observation is that if a circuit has an *isolated variable* (i.e., of out-degree 0 and is not an output node, recall that we are considering single-output functions) or *intersecting critical paths*, then it can be distinguished from truly random functions. This is presented in the following two lemmas.

[26] Because NC^1 circuits are simply B_2 circuits with depth restriction, this lower bound also applies to NC^1 circuits.

Lemma 7.2. There exists a p.p.t. oracle algorithm $\mathcal{A}$ which always accepts if it is given oracle access to a circuit C with intersecting critical path, and rejects with high probability if it is given the truly random function. $\diamondsuit$

Proof. Suppose that inputs x and y have intersecting critical paths. Let G be the gate on their first (i.e., closest to input) intersection. Under any restriction ρ to all the variables except x and y, we can see that the circuit can be viewed as computing

$$C\restriction_\rho(x,y) = f_\rho(G(g_\rho(x), h_\rho(y)))$$

for some unary functions f_ρ, g_ρ, h_ρ. By trying all possible truth tables, one can easily check that no matter what function G is, $C\restriction_\rho(x,y)$ can never compute an $\oplus$-type function for some restriction ρ_1, and compute an $\wedge$-type function for another restriction ρ_2. Indeed, when G is of $\wedge$-type, then $C\restriction_\rho(x,y)$ cannot be an $\oplus$-type function; when G is of $\oplus$-type, $C\restriction_\rho(x,y)$ cannot be an $\wedge$-type function; and when G is degenerate or trivial, $C\restriction_\rho(x,y)$ is also degenerate or trivial.

This motivates us to do the following test for each pair of inputs (x,y). Randomly sample n restrictions for all inputs except x and y. For each sampled restriction ρ, compute the truth table of $C\restriction_\rho(x,y)$, and check whether truth tables of $\oplus$-type and $\wedge$-type appear both. We accept if there exists a pair (x,y), such that either $\oplus$-type or $\wedge$-type does not appear in the truth tables of $C\restriction_\rho(x,y)$ for our n samples of ρ.

If our algorithm is given a circuit with intersecting critical paths, there always exists a pair (x,y) such that the two kinds of truth table does not appear simultaneously, so that our algorithm always accepts. Assume otherwise our algorithm is given a truly random function. Since for each pair (x,y), the n samples of restriction ρ are drawn independently, we can show by union bound that the sampled restrictions ρ are pairwisely distinct with high probability. In such case, the oracle returns independent random bits for our queries, hence both $\oplus$-type and $\wedge$-type truth tables appear for truly random functions with probability $1 - \exp(-\Omega(n))$. By union bound, our algorithm accepts with high probability. $\square$

Lemma 7.3. There exists a p.p.t. oracle algorithm $\mathcal{B}$ which accepts if it is given oracle access to a circuit C with isolated variables, and rejects with high probability if it is given the truly random function. $\diamondsuit$

Proof. Consider the following algorithm. For each variable x, we sample n restrictions for all variables except x. We accept if there exists a variable such that n sampled restrictions give the same output, and reject otherwise. The correctness of our algorithm is easy to verify. $\square$

By combining these two tests, we can distinguish a circuit with either intersecting critical paths or isolated variables. To complete the lower bound, it is sufficient to show that small circuits contain either intersecting critical paths or isolated variables. We prove this by a standard wire counting technique.

Lemma 7.4. For any *normalized* n-input m-output circuit C with no intersecting critical paths and isolated variable, the number of gates in the circuit should be at least $2n - 2m$. $\diamondsuit$

Proof. We divide all nodes (including variables and gates) in the circuit into two types: on the critical path of some variable, or outside of all critical paths. In particular, all variables fall into the first type. Suppose that there are c_1 nodes of the first type, and c_2 nodes of the second. Let l be

the number of wires connecting two nodes of the first type, and o be the number of output nodes belonging to the first type.

Since there is no isolated variables, the endpoints of critical paths must be gates or variables of out-degree at least 2. By the non-intersection property of critical paths, there should be exactly n nodes of the first type having out-degree not equal to 1, hence $(c_1 - n)$ of them have out-degree 1. Since the circuit is normalized, only output gates can have out-degree 0, so that at least $(n - o)$ nodes of first type have out-degree at least 2.

We now count different types of wires. Type $i \to$ Type j denotes the number of wires from Type i nodes to Type j nodes.

(Type 1 $\to$ Type 1) By definition this is l.

(Type 1 $\to$ Type 2) There are at least $2(n - o) + (c_1 - n)$ wires going out of Type 1 nodes, among which l wires go to Type 1, hence there should be at least $c_1 + n - 2o - l$ wires going to Type 2.

(Type 2 $\to$ Type 1) Among the c_1 nodes of Type 1, $(c_1 - n)$ of them are gates. Since each of these gates takes two wires as inputs, the number of wires going from Type 2 to Type 1 is exactly $2(c_1 - n) - l$.

(Type 2 $\to$ Type 2) Since all gates except for $(m - o)$ output nodes of Type 2 have out-degree at least 1, there is at least $c_2 - (m - o)$ wires going out of Type 2. Because $2(c_1 - n) - l$ of them goes to Type 1, there are at least $(c_2 - (m - o)) - 2(c_1 - n) + l$ remains.

Now, notice that the total number of wires going into Type 2 gates is exactly $2c_2$, so we should have the inequality

$$(c_1 + n - 2o - l) + ((c_2 - (m - o)) - 2(c_1 - n) + l) \leq 2c_2,$$

which gives us $c_1 + c_2 \geq 3n - m - o \geq 3n - 2m$. Subtracting the n input nodes from it completes the proof of the lemma. □

Combining these three lemmas, the lower bound is immediate.

Theorem 7.5. If $\mathcal{F} = \{F_n \subseteq B_n\}_{n \geq 1}$ is a PRF, then it requires at least $2n - 2$ gates to compute F_n in general circuits for sufficiently large n. ♢

7.2 An unconditional lower bound for hash function

Since we can construct efficient PRF with almost universal hash function, this lower bound also yields a lower bound for almost universal hash function assuming PRF exists. By modifying the proof a little bit, we are also able to make this lower bound unconditional.

Theorem 7.6. If $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n \geq 1}$ is an almost universal hash function, then it requires at least $2n - 2m$ gates to compute H_n in general circuits for sufficiently large n. ♢

Proof. Towards a contradiction, we assume that for infinitely many bad n, H_n can be realized by B_2 circuits with $2n - 2m$ gates. By Lemma 7.4, for each $h \in H_n$ and any normalized circuit C computing h, C contains either isolated variable or intersecting critical paths.

Let n be an arbitrary "bad" input length, we will construct a set T_n of distinct pairs of inputs such that for all $h \in H_n$, there exists a pair $(x, y) \in T_n$ with $h(x) = h(y)$.

We first consider intersecting critical paths. Suppose that x and y have their critical paths intersecting, and the intersection starts from the gate G. Then by our definition of critical paths, the outputs of the circuit should be completed determined by the output of G if all other inputs are arbitrarily fixed (e.g., fixed to be zeros). Hence there must exist a pair of assignments to x and y such that their outputs collide with all other variables fixed. We can always fix the other variables to 0, and let S_n be the set of all possible colliding pairs of inputs from above. Formally, define

$$\rho_{x,y,a,b}(v) = \begin{cases} a, & v = x \\ b, & v = y \\ 0, & \text{otherwise} \end{cases},$$

then the set S_n is

$$S_n = \{(\rho_{x,y,a,b}, \rho_{x,y,c,d}) \mid x, y \text{ are input variables, } a, b, c, d \in \{0,1\}, (a,b) \neq (c,d)\}.$$

Hence for any circuit containing an intersecting critical path, there must exist a pair in S_n making their outputs collide.

Finding collision for circuits with an isolated variable is rather simple. Let I_n be the set of pairs (x, y) such that $x = 0\|0\|\ldots\|0$ and y contains exactly one non-zero index. All circuits with an isolated variable collides on one of the pairs in I_n.

Let $T_n \triangleq I_n \cup S_n$. It is easy to see that $|T_n| = O(n^2)$. Since n is an bad input length, each $h \in H_n$ has circuit complexity smaller than $2n - 2m$ so that contains either intersecting critical paths or isolated variables, hence indeed has a collision in T_n. By previous discussion we know that

$$\Pr_{h \leftarrow H_n, (x,y) \leftarrow T_n}[h(x) = h(y)] \geq \frac{1}{|T_n|} = \Omega(n^{-2}).$$

By the averaging argument, there exists a particular pair in T_n with collision probability $\Omega(n^{-2})$. This is clearly not an almost universal hash function. □

Note that this lower bound is (almost) tight for $m = o(n)$ by the $2n$ upper bound (see Lemma 6.1). Since we can construct almost universal hash function from good ECC (see Proposition 4.8), we can also obtain an unconditional (almost) tight lower bound for ECC of the same complexity.

7.3 Lower bounds against $\mathsf{AC}^0[2]$ circuits

In Section 6.4 we have shown that almost universal hash function with polynomial shrinkage and PRFs can be constructed in $\mathsf{AC}^0[2]$ with wire complexity $2n + o(n)$. Since any $\mathsf{AC}^0[2]$ circuit with m wires can be translated to a B_2 circuit with at most m gates, we can easily obtain a $2n - O(1)$ lower bound from Theorem 7.5. In this section, we will prove slightly better $2n + \Omega(\sqrt{n})$ lower bounds for PRF and hash function with large shrinkage using a finer wire-counting argument.

Definition 7.7. A pair (x, y) of input variables in an $\mathsf{AC}^0[2]$ circuit is said to be *symmetric* if for each gate G, either none of x and y connect to G or both of x and y connect to G, and in the latter case, x connects to G via a $\neg$ gate if and only if y connects to G via a $\neg$ gate. ♢

Lemma 7.8. Let C be an m-output $\mathsf{AC}^0[2]$ circuit with at most $2n + \sqrt{n}/2 - 2m$ wires. For any $n \geq 100$, if C has no isolated variable, then either it can be converted into a B_2 circuit of size at most $2n - 2m$, or there is a symmetric pair of input variables. ♢

Proof. We prove this lemma by a simple case study.

Case 1. If there are at least $\sqrt{n}/2$ gates, we can easily convert it into an m-output B_2 circuit of size at most $2n - 2m$, since each gate in C of fan-in k can be realized by $k - 1$ gates of fan-in 2. After this case, we can assume that the number of gates is smaller than $\sqrt{n}/2$.

Case 2. If there are at least $\sqrt{n}$ variables with out-degree 1, by the pigeon hole principle, two of them must connect to the same gate in the same manner and therefore we can find a symmetric pair of variables. After this case, we assume that at most $\sqrt{n}$ variables of out-degree at most 1.

Case 3. Recall that C has wire complexity at most $2n + \sqrt{n}/2 - 2m$. Since there are at most $\sqrt{n}$ variables of out-degree at most 1, we can see that at most $3\sqrt{n}/2$ variables have out-degree larger than 2. As a result, at least $n - 5\sqrt{n}/2$ variables have out-degree 2. Because there are at most $\sqrt{n}/2$ gates, the number of different ways for an out-degree-2 variable to connect to the circuit is at most $4 \cdot (\sqrt{n}/2)^2/2 = n/2 < n - 5\sqrt{n}/2$ for $n \geq 100$. Hence there must be two variables of out-degree 2 connecting to the same pair of gates in the same manner, which forms a symmetric pair. □

Lemma 7.9. There exists a p.p.t. oracle algorithm $\mathcal{D}$ which accepts if it is given oracle access to a circuit with at least one symmetric pair of variables, and rejects with high probability if it is given the truly random function. ♢

Proof. The algorithm enumerates all pairs (x_i, x_j) of distinct input variables. For each of such pair, it randomly sample n assignments to the input variables and see whether exchanging the value of x_i and x_j would change the function value. The algorithm accepts if and only if there exists a pair (x_i, x_j) such that for all n sampled assignment, exchanging the value of x_i and x_j would not change the function value. The correctness of the algorithm is obvious. □

Combining the three algorithms in Lemma 7.2, Lemma 7.3 and Lemma 7.8, we can distinguish $\mathsf{AC}^0[2]$ circuits with small wire complexity with truly random functions. This gives the following PRF lower bound.

Theorem 7.10. If $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$ is a PRF, then it requires at least $2n + \sqrt{n}/2 - 2$ wires to compute F_n in $\mathsf{AC}^0[2]$ circuits for sufficiently large n. ♢

Similar to Section 7.2, we can also prove an unconditional $\mathsf{AC}^0[2]$ lower bounds against almost universal hash functions (and therefore a lower bound against ECC).

Theorem 7.11. If $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ is an almost universal hash function, then it requires at least $2n + \sqrt{n}/2 - 2m$ wires to compute F_n in $\mathsf{AC}^0[2]$ circuits for sufficiently large n. ♢

Proof. Recall that in the proof of Theorem 7.6 we have construct a subset $T_n \subseteq \mathbb{F}_2^n \times \mathbb{F}_2^n$ of size $O(n^2)$ such that for all B_2 circuits C with either intersecting critical paths or isolated variables, there exists a pair $(x, y) \in T_n$ such that $C(x) = C(y)$. Let $\hat{T}_n \triangleq T_n \cup \{(x, y) \mid x \neq y, |x| = |y| = 1\}$. By Lemma 7.8, for all $\mathsf{AC}^0[2]$ circuits of size at most $2n + \sqrt{n}/2 - m$, there exists a pair $(x, y) \in \hat{T}_n$ such that $C(x) = C(y)$. Following the same argument of Theorem 7.6 we immediately obtain this lower bound. □

8 A size-depth trade-off lower bound against TC^0

In this section, we consider the regime of constant-depth linear threshold circuits, and prove a slightly super-linear size-depth trade-off lower bound for pseudorandom function. Our lower bound follows from a structural result for threshold circuits used in developing average-case hardness [CSS18], quantified derandomization [Tel18] and pseudorandomness [HHTT21].

8.1 Extracting black-box property from white-box restriction

In this section, we will show that computing a PRF with depth-d threshold circuits require size at least $n^{1+\Omega(1)^d}$.

Theorem 8.1. There exist $\theta > 0$ and $c > 1$ such that the following lower bound holds. If $\mathcal{F} = \{F_n \subseteq B_n\}_{n\geq 1}$ is a PRF, then for all $d \geq 1$ and sufficiently large n, it requires at least $n^{1+\theta c^{-d}}$ wires to compute F_n in TC^0_d circuits. ♢

The technique we will use is the "white-box" random restriction method developed in previous works of average-case hardness and pseudorandomness for TC^0 circuits [CSS18; Tel18; HHTT21]. Although their results are presented in different forms, all of them essentially use the following fact about threshold circuits: for a random restriction (or pseudorandom restriction) ρ applied to a small threshold circuit C of depth d, with nice probability, there exists a properly large subset S_ρ of free variables such that for a random assignments σ to all variables but S_ρ, again with nice probability, the circuit C restricted by $\rho\sigma$ can be approxmable by a small threshold circuit of depth $d-1$. For our purpose, we formalize this fact as Lemma 8.3 and generalize it to multi-output case. We defer its proof to Section 8.3.

Definition 8.2. Let $n, m \geq 1$ and $0 \leq \varepsilon \leq 1$ be a parameter. A function $f \in B_{n,m}$ is said to be ε-approximable by a TC^0_d circuit C if for a uniformly random input x, $C(x) \neq f(x)$ with probability at most ε. A function $f \in B_{n,m}$ is said to be *transparent* if each of its output bits is a constant or only depends on exact one input variable. ♢

Lemma 8.3 (Restriction lemma for TC^0). There exist absolute constants $c > 30$ and some constant $0 < \varepsilon_0 < 1$ such that following holds. For all $\varepsilon < \varepsilon_0$ and function $\alpha_1(n)$, there exists some $\delta > 0$, such that for all depth $d \geq 1$, output length $m = m(n)$ and sufficiently large n, for any $f \in B_{n,m}$ that is $\alpha_1(n)$-approximable by a multi-output depth d threshold circuit of size $n^{1+\varepsilon}$, if we apply the restriction $\rho \leftarrow \mathcal{R}_{n^{-\delta}}$ to the function, with probability at least $1/8$ there exists a set of unfixed variables S_ρ (which depends on ρ) of size at least $n^{0.99}$, such that at least a half of the restrictions σ to *all* variables not fixed by ρ and not in S_ρ would make $f\restriction_{\rho\sigma}$ $4(\alpha_1(n) + \alpha_2(n))$-approximable by a multi-output threshold circuit of depth $d-1$ and size $|S_\rho|^{1+c\varepsilon}$ (or a transparent function if $d = 1$), where $\alpha_2(n) \triangleq 2n^{1+\varepsilon-\delta}\exp(-2n^{\delta c_1})$. ♢

Lemma 8.3 and its counterparts in [CSS18; Tel18] are considered as white-box techniques because the subset S_ρ of unfixed variables is chosen according to the circuit approximating f and the result of the random restriction ρ.[27] To prove circuit lower bound for pseudorandom functions,

[27] We note that the restriction lemma for TC^0 in [HHTT21] is indeed black-box: they argue that the circuit simplifies to a hybrid computation model called LTF decision tree with appropriate parameters after the random restriction even without selecting S_ρ. Unfortunately, we are not aware of any simple way to obtain our lower bounds with their restriction lemma.

we need to specify a black-box property of sparse TC^0_d circuits to distinguish it from truly random function with only oracle accesses. The key observation of our lower bound is that the following black-box property can be extracted from the white-box restriction lemma.

Lemma 8.4. Let c be the constant in Lemma 8.3. There exists constants $\theta > 0$ so that the following holds. For all depth $d \geq 1$, output length $m = m(n) < n^{0.99^d}$, and sufficiently large n, for any $f \in B_{n,m}$ that is $1/8^{d+1}$-approximable by a multi-output depth d threshold circuit of size less than $n^{1+\theta c^{-d}}$, for a random assignment $x \in \mathbb{F}_2^n$ to all input variables, with probability at least $1/16^{d+1}$, there exists another assignment y different from x by exact one bit such that $f(x) = f(y)$ holds. $\diamondsuit$

Proof. Assume that $\varepsilon_0, \delta, p, S_\rho, \alpha_2(n)$ follows Lemma 8.3. Take $\theta < \varepsilon_0$ (therefore $\theta c^{-d} < \varepsilon_0$ for any d). We will prove the statement by doing induction over d using Lemma 8.3. Assume that n is sufficiently large[28] and $f \in B_{n,m}$ is $1/8^{d+1}$-approximable by a multi-output TC^0_d circuit of size $n^{1+\theta c^{-d}}$. Since $\alpha_2(n) = \exp(-\mathsf{poly}(n))$, we have $\alpha_2(n) < 1/8^{d+1}$ for sufficiently large n.

For simplicity, we call an assignment x *good* if there exists another assignment y that differs on exactly one bit from x such that $f(x) = f(y)$. What we want to show is that $\Pr[x \text{ is good}] \geq 1/16^{d+1}$ for uniformly random x. For a random assignment to all input variables, we consider it as a pair of a random restriction $\rho \leftarrow \mathcal{R}_p$ and a random assignment β to the leftover variables. Clearly, it is sufficient to show that

$$\Pr_{\rho \leftarrow \mathcal{R}_p, \beta \leftarrow \{-1,1\}^{|\rho^{-1}(\star)|}} [(\rho, \beta) \text{ is good}] \geq \frac{1}{16^{d+1}}. \tag{1}$$

We call a random restriction $\rho \leftarrow \mathcal{R}_p$ *good* if S_ρ in Lemma 8.3 exists. By Lemma 8.3, $\Pr[\rho \text{ is good}] \geq 1/8$. Then it is sufficient to show that for any fixed good ρ,

$$\Pr_{\beta \leftarrow \{-1,1\}^{|\rho^{-1}(\star)|}} [(\rho, \beta) \text{ is good}] \geq \frac{8}{16^{d+1}}. \tag{2}$$

Now we fix any good ρ (so that there exists S_ρ satisfying Lemma 8.3) and prove Equation (2). In such case, the assignment β to the leftover variables can be further considered as the composition of a random restriction σ to variables not in S_ρ and another random assignment τ to the remaining variables S_ρ. Identify $\beta = (\sigma, \tau)$ as discussed above. We call an assignment σ *good* if $f\restriction_{\rho\sigma}$ can be approximable by a TC^0_{d-1} circuit of size $|S_\rho|^{1+\theta c^{-(d-1)}}$ (or a transparent function) with error $4(1/8^{d+1} + \alpha_2(n)) < 1/8^d$. By Lemma 8.3, at least a half of the assignments σ are good. This means to prove Equation (2), it is sufficient to show that for any fixed good ρ and good σ,

$$\Pr_{\tau \leftarrow \{-1,1\}^{|S_\rho|}} [(\rho, \sigma, \tau) \text{ is good}] \geq \frac{1}{16^d}. \tag{3}$$

Firstly we consider $d = 1$. In this case, $f\restriction_{\rho\sigma}$ can be $1/8$ approximated by a transparent function such that each output bit depends on at most one input variable. Since the output length $m < n^{0.99}$ is smaller than the input length $|S_\rho| \geq n^{0.99}$, the transparent function is independent from some input variable v. By union bound, for uniformly chosen τ and the corresponding τ' only differ on

[28] Rigorously speaking, we will take $n > n_0$, where n_0 satisfies the following three constraints: (1) Lemma 8.3 holds for depth d; (2) induction hypothesis holds for depth $d-1$; and (3) the inequality $\alpha_2(n) < 1/8^{d+1}$ holds.

input variable v, $\Pr[f\restriction_{\rho\sigma}(\tau) = f\restriction_{\rho\sigma}(\tau')] \geq 3/4$. By averaging argument, for at least $1/2$ fraction of τ, there exists some τ' such that $f\restriction_{\rho\sigma}(\tau) = f\restriction_{\rho\sigma}(\tau')$. This immediately shows that

$$\Pr_{\tau \leftarrow \{-1,1\}^{|S_\rho|}}[(\rho, \sigma, \tau) \text{ is good}] \geq \frac{1}{2} \geq \frac{1}{16^d}.$$

When $d > 1$, there exists a multi-output threshold circuit of depth $d-1$ and size less than $|S_\rho|^{1+\theta c^{-(d-1)}}$ that $1/8^d$-approximates $f\restriction_{\rho\sigma}$, where $|S_\rho|$ is the input length of the new circuit. Since $|S_\rho| \geq n^{0.99}$, this circuit is of output length $m < n^{0.99^d} \leq |S_\rho|^{0.99^{d-1}}$ which satisfies the induction condition. According to induction hypothesis, for at least $1/16^d$ fraction of τ, there exists τ' such that $f\restriction_{\rho\sigma}(\tau) = f\restriction_{\rho\sigma}(\tau')$. Then Equation (3) immediately follows since for each of such τ, (ρ, σ, τ) is good. This completes the induction. □

This lemma shows that finding a collision is easy for multi-output functions approximable by sparse threshold circuits. Since it is hard to find a collision for truly random functions, we can then distinguish sparse threshold circuits from truly random functions.

Proof of Theorem 8.1. We will prove this result by presenting a distinguisher which separates functions computable by sparse TC^0_d circuits and truly random functions. Suppose that the set of input variables is I, and the algorithm is given oracle access to the function f. Our algorithm is quite simple.

- Let S be the subset of first $\ell(n) = 2\lceil \log \log n \rceil$ input variables.
- Randomly choose an assignment $x \leftarrow \mathbb{F}_2^{|I \setminus S|}$ for other variables.
- Randomly flip an input variable of x to obtain y.
- Accept if $f(z\|x) = f(z\|y)$ holds for all assignment $z \in \mathbb{F}_2^{|S|}$.

Since there are in total $2^{\ell(n)} = \Theta(\log^2 n)$ different assignments to S, the algorithm queries $\Theta(\log^2 n)$ separated pairs of assignments which can be done in polynomial time. For truly random functions, each pair of assignments outputs the same result with probability $1/2$ independently. Thus, the algorithm would accept with probability $2^{-\Theta(\log^2 n)}$, which is negligible.

We then show the algorithm would accept functions computable by sparse TC^0 circuits with non-negligible probability. Firstly, f can be viewed as a multi-output function $g \in B_{n-\ell(n),2^{\ell(n)}}$ such that z-th output bit of $g(x)$ is $f(z\|x)$. Clearly, our algorithm accept if and only if $g(x) = g(y)$ holds. Since each output bit of g compute f under some restriction, g can be computed by a TC^0_d circuit of output length $\Theta(\log^2(n))$ and wire complexity less than $\Theta(\log^2(n)) \cdot n^{1+\theta c^{-d}}$. Let θ be some constant moderately smaller than the parameter required in Lemma 8.4. When n is sufficiently large, both the output length and the wire complexity would satisfy the requirement in Lemma 8.4. Thus, there exists some y such that $g(x) = g(y)$ with probability at least $1/16^{d+1}$. Since there are only n different ways to flip one bit, the algorithm can hit the proper y with probability at least $1/n$. This shows the algorithm can accept the original TC^0 circuit with probability at least $1/(16^{d+1}n)$, which is non-negligible.

Notice that we can easily boost up the gap between the acceptance probability of the two cases by repeating the test polynomial times. So the algorithm can effectively distinguish two cases, which concludes that any PRF cannot be computed by threshold circuits with less than $n^{1+\theta c^{-d}}$ wires if the depths of the circuits are bounded by d. □

8.2 A lower bound for hash functions against TC^0

Similar to Theorem 7.6, we can modify the proof for the PRF lower bound to obtain an unconditional circuit lower bound for almost universal hash functions (and also for error-correcting codes with exactly the same wire complexity).

Theorem 8.5. Let $c > 30$ be the constant in Lemma 8.3. There exists a constant $\theta > 0$ so that the following holds. Let $m = m(n) < n^{0.99^d}$. If $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ is an almost universal hash function, then for all $d \geq 1$ and sufficiently large n, it requires at least $n^{1+\theta c^{-d}}$ wires to compute H_n in $\mathsf{TC}^0_d[2]$ circuits. ♢

Proof. Let θ be the constant in Lemma 8.4. Towards a contradiction assume that $\mathcal{H}$ can be computed by TC^0_d circuits of size $n^{1+\theta c^{-d}}$ for infinitely many length n. Consider the following distribution $\mathcal{D}_n$ supported over pairs of distinct n-bit inputs: we first choose an input x uniformly, choose an index $i \leftarrow [n]$, and generate the pair (x, y) where y differs from x only on the i^{th} bit. By Lemma 8.4, for any TC^0_d circuit C of size $n^{1+\theta c^{-d}}$, we have

$$\Pr_{(x,y)\leftarrow\mathcal{D}_n}[C(x) = C(y)] \geq \frac{1}{16^{d+1}n}.$$

This means that for those bad n,

$$\Pr_{(x,y)\leftarrow\mathcal{D}_n, h\leftarrow H_n}[h(x) = h(y)] \geq \frac{1}{16^{d+1}n}.$$

By averaging argument, for each of the bad n, there exists a pair (x, y) of distinct inputs such that $h(x) = h(y)$ with non-negligible probability. This contradicts the almost universality of $\mathcal{H}$. □

Note that here we need to assume $m < n^{0.99^d}$ to apply Lemma 8.4. In fact, we can remove this requirement to make our lower bounds more general for large d.

Corollary 8.6. Let $c > 30$ be the constant in Lemma 8.3 and $\hat{\theta} \in (0, 1)$ be an absolute constant. For any constant $\varepsilon \in (0, 1/3)$ and output length $m = m(n) = \Theta(n^\varepsilon)$, if $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ is an almost universal hash function, then it requires at least $n^{1+\hat{\theta}c^{-d}}$ wires to compute H_n in TC^0_d circuits for all $d \geq 1$ and sufficiently large n. ♢

Proof. Let θ be the constant in Theorem 8.5. Set $\hat{\theta} \triangleq 0.99 \cdot \theta \cdot c^{-2}$. Towards a contradiction we assume that there exists an almost universal hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ for $m = \Theta(n^\varepsilon)$ that is computable by TC^0_d circuits of $n^{1+\hat{\theta}c^{-d}}$ wires for infinitely many n. If $\varepsilon < 0.99^d$ we can immediately derive a contradiction by Theorem 8.5.

Now we assume that $\varepsilon \geq 0.99^d$. By Corollary 6.4, we can construct a linear almost universal hash function $\hat{\mathcal{H}} = \{\hat{H}_m \subseteq B_{m,k}\}_{m\geq 1}$ for $k = k(m) = \Theta(m^{0.99^d/(2\varepsilon)})$. Since the parity function of m variables can be computed by a TC^0_2 circuit of wire complexity $O(m^2)$ (see, e.g., [PS90]), $\hat{H}_m$ can be realized by TC^0_2 circuits of wire complexity $O(km^2)$. The composition of $\hat{\mathcal{H}}$ and $\mathcal{H}$ turns out to be an almost universal hash function with output length $k(m) < n^{0.99^d}$ computable in TC^0_{d+2} with wire complexity

$$n^{1+\hat{\theta}c^{-d}} + O(km^2) \leq n^{1+\hat{\theta}c^{-d}} + o(n) \leq n^{1+\theta c^{-(d+2)}}$$

for sufficiently large n. This leads to a contradiction with Theorem 8.5. □

8.3 Proof of the restriction lemma

The technique we use to prove the restriction lemma follows the standard method based on anti-concentration of threshold functions, which has been used to prove average-case hardness [CSS18], quantified derandomization [Tel18], and pseudorandom generator [HHTT21] for sparse TC^0 circuits. There is essentially nothing new in the proof, but we need to carefully check that the previous proofs can be adapted to our statement. The proof mainly involves three steps.

1. We notice that after a random restriction, most gates in depth-1 connected to many variables become highly "imbalanced" to be approximable by constants. This procedure is done by a structural lemma from [CSS18].

2. Then we count the number of variables feeding a gate that is both "balanced" and large fan-in. It can be shown that the number of such variables is $o(n)$ with nice probability. We will not include these variables in S_ρ, so that fixing all variables outside S_ρ will make all such gates (i.e., both balanced and of large fan-in) become constants.

3. Finally, we only need to consider gates with small fan-in. It is easy to see that it can select sufficiently many variables into S_ρ, such that for each small gate, at most one of its input variables is in S_ρ. By fixing all variables but S_ρ, all gates in depth-1 can be approximable by constants.

We start by defining what "balance" means to a threshold function.

Definition 8.7 (t-balance). A function $\mathsf{LTF}_{w,\theta}(x)$ is called *t-balanced* if $|\theta| \leq t \cdot \|w\|_2$. Otherwise, it is called *t-imbalanced*. ♢

By Hoeffding's inequality (Lemma 4.12), a t-imbalanced LTF function has a large fraction of its input being constant.

Proposition 8.8. Let $\mathsf{LTF}_{w,\theta}(x)$ be a t-imbalanced function, then

$$\Pr_{x\in\{-1,1\}^n}[\mathsf{LTF}_{w,\theta}(x) = \mathrm{sgn}(\theta)] \leq 2\exp(-2t^2). \qquad ♢$$

Proof. We suppose, without loss of generality, that $\theta > 0$. Then,

$$\begin{aligned}
\Pr_{x\in\{-1,1\}^n}[\mathsf{LTF}_{w,\theta}(x) = 1] &= \Pr_{x\in\{-1,1\}^n}[\langle w, x\rangle \geq t\|w\|_2] && (t\text{-imbalance})\\
&\leq 2\exp\left(-\frac{2t^2\|w\|_2^2}{\sum_{i=1}^n w_i^2}\right) && (\text{Lemma 4.12})\\
&= 2\exp(-2t^2). && \square
\end{aligned}$$

We will make use of the following anti-concentration lemma for linear threshold functions.

Lemma 8.9 ([CSS18], Lemma 4.4; or [Tel18], Proposition 5.8). There exist absolute constants $p_0 < 1, c_1, c_2 > 0$, such that for any $p \in [0, p_0]$, any LTF function Φ over n input variables satisfies

$$\Pr_{\rho\leftarrow\mathcal{R}_p}\left[\Phi\restriction_\rho \text{ is } p^{-c_1}\text{-balanced}\right] \leq O(p^{c_2}). \qquad ♢$$

This lemma tells us that after a random p-restriction, any LTF function will become extremely biased with nice probability. If a function is indeed imbalanced after random restriction, we can then approximate it using a constant by Proposition 8.8.

We are now ready to prove the main restriction lemma, following the intuition above.

Reminder of Lemma 8.3. There exist absolute constants $c > 1$ and some constant $0 < \varepsilon_0 < 1$ such that following holds. For all $\varepsilon < \varepsilon_0$ and function $\alpha_1(n)$, there exists some $\delta > 0$, such that for all depth $d \geq 1$, output length $m = m(n)$, and sufficiently large n, for any $f \in B_{n,m}$ that is $\alpha_1(n)$-approximable by a multi-output depth d threshold circuit of size $n^{1+\varepsilon}$, if we apply the restriction $\rho \leftarrow \mathcal{R}_{n^{-\delta}}$ to the function, with probability at least $1/8$ there exists a set of unfixed variables S_ρ (which depends on ρ) of size at least $n^{0.99}$, such that at least a half of the restrictions σ to *all* variables not fixed by ρ and not in S_ρ would make $f\restriction_{\rho\sigma}$ $4(\alpha_1(n) + \alpha_2(n))$-approximable by a multi-output threshold circuit of depth $d-1$ and size $|S_\rho|^{1+c\varepsilon}$ (or a transparent function if $d = 1$), where $\alpha_2(n) \triangleq 2n^{1+\varepsilon-\delta}\exp(-2n^{\delta c_1})$. ♢

Proof. Let $f \in B_{n,m}$ be a function that can be $\alpha_1(n)$-approximated by a depth d threshold circuit C of size $n^{1+\varepsilon}$. Assume that $\phi_1, \phi_2, \ldots, \phi_t$ are the gates of depth 1 (i.e., directly fed by inputs). A gate ϕ_i is called *small* if its in-degree is at most n^δ, where δ is a constant to be chosen later; and is called *large* otherwise. We use Φ_s and Φ_l to denote the set of small and large gates of depth 1, respectively. A variable x_i is called *small* if its out-degree is at most $n^{2\varepsilon}$, and is called *large* otherwise. We use X_s and X_l to denote the set of small and large variables, respectively. Since the circuit has only $n^{1+\varepsilon}$ wires, $|\Phi_l| \leq n^{1+\varepsilon-\delta}$ and $|X_l| \leq n^{1-\varepsilon}$.

Assume that the circuit C is randomly restricted with $\rho \leftarrow \mathcal{R}_p$ for $p = n^{-\delta/2}$. For simplicity, we identify the restriction ρ as a pair (I, y) where $I \subseteq [n]$ represents the fixed variables and $y \in \{-1,1\}^{|I|}$ represents the assignment. A restriction $\rho = (I, y)$ is called *generic for a large gate* $\phi_i \in \Phi_l$ of in-degree k if the number of free variables after the restriction is in the range $[kp/2, 3kp/2]$, and a restriction $\rho = (I, y)$ (or simply an I) is called *generic* if $|X_s \cap \rho^{-1}(\star)| \geq pn/2$, and it is generic for all large gates. Let $\mathcal{G}$ be the event that ρ is generic. Note that $|X_s| \geq n - n^{1-\varepsilon} = n - o(n)$. Clearly,

$$
\begin{aligned}
&\Pr_{\rho\leftarrow\mathcal{R}_p}[\neg\mathcal{G}] \\
\leq &\Pr_{\rho\leftarrow\mathcal{R}_p}[|X_s \cap \rho^{-1}(\star)| < pn/2] + \sum_{\phi\in\Phi_l}[\rho \text{ is not generic for } \phi] && \text{(Union bound)} \\
\leq &\exp(-\Omega(pn)) + \sum_{\phi\in\Phi_l}[\rho \text{ is not generic for } \phi] && \text{(Chernoff bound)} \\
\leq &\exp(-\Omega(pn)) + |\Phi_l| \cdot \exp(-\Omega(p \cdot \text{in-degree}(\phi))) && \text{(Chernoff bound)} \\
\leq &\exp(-\Omega(n^{1-\delta/2})) + n^{1+\varepsilon-\delta}\exp(-\Omega(n^{\delta/2})) \\
\leq &\mathsf{negl}(n).
\end{aligned}
$$

Hence we can stick to the case when the restriction is generic from now on.

Recall that our goal is to find a large subset S_ρ of unfixed variables such that at least a half of the restrictions that fixes all variables not in S_ρ would make f approximable by a small TC^0_{d-1} circuit. Let F_ρ be the set of unfixed variables not in S_ρ. Now will we define the set F_ρ and S_ρ by the following three-phase procedure.

Large variables. Let $n_1 = |X_s \cap I|$, which is the number of unfixed small variables. If ρ is generic, we know that $n_1 \geq pn/2 \geq n^{1-\delta/2}/2$, which is sufficiently large for small δ. In this phase, we simply put all unfixed large variables into F_ρ.

Large gates. Let $\phi_i \in \Phi_l$ be a large gate. By Lemma 8.9, we know that $\Pr_{\rho \leftarrow \mathcal{R}_p}[\phi_i \text{ is } p^{-c_1}\text{-balanced}] \leq p^{c_2}$ for absolute constants c_1 and c_2, which means that for large n,

$$\Pr_{\rho \leftarrow \mathcal{R}_p}[\phi_i \text{ is } p^{-c_1}\text{-balanced} \mid \mathcal{G}] \leq \frac{p^{c_2}}{\Pr_{\rho \leftarrow \mathcal{R}_p}[\mathcal{G}]} \leq \frac{p^{c_2}}{1 - \mathsf{negl}(n)} \leq 2p^{c_2}.$$

By Proposition 8.8, the large gates that become imbalanced would output a constant value with high probability, so that they will not bother us. Hence in this phase, we will put all the inputs of p^{-c_1}-*balanced* large gates into F_ρ.

To analyze the number of inputs that will be put into F_ρ, we define a random variable Y_i for each $\phi_i \in \Phi_l$, which is 0 if ϕ_i is p^{-c_1}-imbalanced, and is the in-degree of ϕ_i after the restriction if ϕ_i is p^{-c_1}-balanced. Let $Y = \sum_{\phi_i \in \Phi_l} Y_i$ be an upper bound of the number of variables to be put into F_ρ. Clearly,

$$\mathop{\mathbb{E}}_{\rho \leftarrow \mathcal{R}_p}[Y_i \mid \mathcal{G}] \leq \frac{3p}{2} \cdot \text{in-degree}(\phi_i) \Pr_{\rho \leftarrow \mathcal{R}_p}[\phi_i \text{ is } p^{-c_1}\text{-balanced} \mid \mathcal{G}] \leq 3p^{1+c_2} \cdot \text{in-degree}(\phi_i),$$

hence

$$\mathop{\mathbb{E}}_{\rho \leftarrow \mathcal{R}_p}[Y \mid \mathcal{G}] = \sum_{\phi_i \in \Phi_l} \mathop{\mathbb{E}}_{\rho \leftarrow \mathcal{R}_p}[Y_i \mid \mathcal{G}] \leq 3p^{1+c_2} \cdot \sum_{\phi_i \in \Phi_l} \text{in-degree}(\phi_i) \leq 3p^{1+c_2} n^{1+\varepsilon}.$$

Let $\mu \triangleq 3p^{1+c_2} n^{1+\varepsilon}$. By Markov's inequality, we know that

$$\Pr_{\rho \leftarrow \mathcal{R}_p}[Y > 4\mu \mid \mathcal{G}] \leq \frac{1}{4}.$$

Let $\mathcal{L}$ be the event that $Y \leq 4\mu$, i.e., only 4μ variables are put into F_ρ in this phase. If ρ is generic and we set δ to be moderately large, say $\delta = 3\varepsilon/c_2$, with probability at least $3/4$, $\mathcal{L}$ will happen, in which case we will put only $O(n^{1+\varepsilon-(1+c_2)\delta/2}) = o(n_1)$ variables into F_ρ.

Small variables feeding small gates. Now we deal with small gates in depth 1. We will put (roughly) all but $n_1^{1-(2\varepsilon+\delta)}$ variables into F_ρ, so that for each small gate $\phi_j \in \Phi_s$, at most one of its input variables is free. Consider the undirected graph $G = (V, E)$ where each node represents a variables that is neither fixed nor put in F_ρ in the above two cases, and two nodes are connected if both of them feed a small gate $\phi_j \in \Phi_s$. Since the out-degree of small variables and in-degree of small gates are all bounded, each node in G is of degree at most $n^{2\varepsilon+\delta}$, hence there exists an independent set S of size $|V|/n^{2\varepsilon+\delta}$. We define S_ρ to be all the vertices inside the independent set, and put all other variables into F_ρ. Note that condition on $\mathcal{G}$ and $\mathcal{L}$, the size of S_ρ is at least $\beta \triangleq \frac{np/2-4\mu}{n^{2\varepsilon+\delta}}$.

Analysis. Now it is sufficient to show that under the random restriction ρ, with nice probability, $|S_\rho| \geq \beta = \frac{np/2-4\mu}{n^{2\varepsilon+\delta}}$ and for at least a half of the assignments σ to the unfixed variables not in S_ρ, $f\restriction_{\rho\sigma}$ can be approximated by a circuit of size $|S_\rho|^{1+c\varepsilon}$ for an absolute constant c. Let $\rho = (I_\rho, y_\rho)$ and $\sigma = (I_\sigma, y_\sigma)$, the circuit C' that approximates $f\restriction_{\rho\sigma}$ is defined as follows. We start with the circuit C that approximates f.

1. For a small gate $\rho_j \in \Phi_s$, since at most one of its input variables is in S_ρ, it is of in-degree at most 1 after the restrictions ρ and σ. Hence we can make its descendants directly fed by its input nodes and modify the functions computed by its descendants accordingly without deviating the functionality of the circuit.

2. By the phase dealing with large gates, all p^{-c_1}-balanced large gates become in-degree 0. We can replace them by constants and simplify the circuit accordingly. For a p^{-c_1}-imbalanced large gate $\phi_i = \mathsf{LTF}_{w,\theta}$ after the restriction, we simply replace it by the most probable constant following Proposition 8.8.

For $d > 1$, the depth of C' becomes $d-1$ since all the gates in depth 1 are eliminated. When $d = 1$, this circuit becomes transparent, i.e., each output node depends on at most one input bit. By the preceding discussion, we have already known that $|S_\rho| \geq \beta$ with certainty if $\mathcal{G}$ and $\mathcal{L}$ holds simultaneously. Now we set the parameters $\varepsilon_0 \triangleq 0.01 \min\{c_2, 1/(2+4.5/c_2)\}$ and $\delta \triangleq 3\varepsilon/c_2$, such that for large n,

$$\mu = o(np), \beta = \left(\frac{1}{2} - o(1)\right) n^{1-2\varepsilon-3\delta/2} \geq \left(\frac{1}{2} - o(1)\right) n^{1-(2+4.5/c_2)\varepsilon} > n^{0.99}.$$

Since the size of C' is at most the size of C, we can see that for large constant c independent of ε, n and d,

$$\mathrm{size}(C') \leq \mathrm{size}(C) \leq n^{1+\varepsilon} \leq \beta^{\frac{1+\varepsilon}{1-(2+4.5/c_2)\varepsilon}} \leq |S_\rho|^{1+c\varepsilon}$$

for sufficently large n conditioning on $\mathcal{G}$ and $\mathcal{L}$. Let $\mathcal{S}$ be the event that $|S_\rho| \geq \beta$ and $\mathrm{size}(C') \leq |S_\rho|^{1+c\varepsilon}$. So we can see that

$$\Pr_{\rho\leftarrow\mathcal{R}_p, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}}[\mathcal{S} \mid \mathcal{G}] = \Pr_{\rho\leftarrow\mathcal{R}_p, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}}[\mathcal{L}] \cdot \Pr_{\rho\leftarrow\mathcal{R}_p, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}}[\mathcal{S} \mid \mathcal{G}, \mathcal{L}] = \Pr_{\rho\leftarrow\mathcal{R}_p, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}}[\mathcal{L}] \geq \frac{3}{4}. \tag{4}$$

Now we show that C' approximates $f\restriction_{\rho\sigma}$. Fix an arbitrary generic I_ρ. Firstly, we can see that

$$\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[f\restriction_{\rho\sigma}(z) \neq C(y_\rho, y_\sigma, z)] = \Pr_{z\leftarrow\mathbb{F}_2^n}[f(z) \neq C(z)] \leq \alpha_1(n),$$

since f is $\alpha_1(n)$-approximated by C. Notice that replacing imbalanced gates is the only place that introduces additional errors in the procedure defining S_ρ. Let $\phi\restriction_\rho$ denote the gate ϕ under restriction ρ (so that its input is restricted to unfixed variables and its internal function is modified). We can see that

$$\begin{aligned}
&\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[C(y_\rho, y_\sigma, z) \neq C'(z)] \\
\leq &\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[\exists \phi = \mathsf{LTF}_{w,\theta} \in \Phi_l, \phi\restriction_\rho \text{ is } p^{-c_1}\text{-imbalanced} \wedge \phi(y_\rho, y_\sigma, z) = \mathrm{sgn}(\phi\restriction_\rho)] \\
\leq &\sum_{\phi=\mathsf{LTF}_{w,\theta}\in\Phi_l} \left[\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[\phi \text{ is } p^{-c_1}\text{-imbalanced} \wedge \phi = \mathrm{sgn}(\phi\restriction_\rho)]\right] \qquad \text{(Union bound)} \\
\leq &\sum_{\phi=\mathsf{LTF}_{w,\theta}\in\Phi_l} \left[\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[\phi = \mathrm{sgn}(\phi\restriction_\rho) \mid \phi \text{ is } p^{-c_1}\text{-imbalanced}]\right] \\
\leq &2|\Phi_l| \cdot \exp(-2p^{-c_1}) \qquad \text{(Proposition 8.8)} \\
\leq &2n^{1+\varepsilon-\delta} \exp(-2n^{\delta c_1}).
\end{aligned}$$

Let $\alpha_2(n) \triangleq 2n^{1+\varepsilon-\delta}\exp(-2n^{\delta c_1})$. Again by union bound,

$$\Pr_{y_\rho\leftarrow\mathbb{F}_2^{|I_\rho|}, y_\sigma\leftarrow\mathbb{F}_2^{|I_\sigma|}, z\leftarrow\mathbb{F}_2^{|S_\rho|}}[f\restriction_{\rho\sigma}(z) \neq C'(z)] \leq \alpha_1(n) + \alpha_2(n).$$

By averaging argument, for at least $1/2$ fraction of assignments y_ρ to I_ρ, there exists $1/2$ fraction of assignments y_σ to I_σ such that

$$\Pr_{z \leftarrow \mathbb{F}_2^{|S_\rho|}}[f\restriction_{\rho\sigma}(z) \neq C'(z)] \leq 4(\alpha_1(n) + \alpha_2(n)).$$

Let $\mathcal{A}$ be the event that for at least $1/2$ fraction of assignments y_σ to the variables in I_σ, $f\restriction_{\rho\sigma}$ is $4(\alpha_1(n) + \alpha_2(n))$-approximated by C'. Then

$$\Pr_{\rho \leftarrow \mathcal{R}_p}[\mathcal{A} \mid \mathcal{G}] \geq \frac{1}{2}. \tag{5}$$

Combining (4) and (5), we can see that for large n,

$$\begin{aligned}
&\Pr_{\rho \leftarrow \mathcal{R}_\rho}[\mathcal{S} \wedge \mathcal{A}] \\
\geq &\Pr_{\rho \leftarrow \mathcal{R}_\rho}[\mathcal{S} \wedge \mathcal{A} \mid \mathcal{G}] \Pr_{\rho \leftarrow \mathcal{R}_p}[\mathcal{G}] \\
\geq &\left(1 - \Pr_{\rho \leftarrow \mathcal{R}_\rho}[\neg\mathcal{S} \mid \mathcal{G}] - \Pr_{\rho \leftarrow \mathcal{R}_\rho}[\neg\mathcal{A} \mid \mathcal{G}]\right) \Pr_{\rho \leftarrow \mathcal{R}_p}[\mathcal{G}] \\
\geq &\left(1 - \frac{1}{4} - \frac{1}{2}\right)(1 - \mathsf{negl}(n)) \\
\geq &\frac{1}{8}.
\end{aligned}$$

This means that under a random restriction $\rho \leftarrow \mathcal{R}_p$, with probability $1/8$, there exists a subset S_ρ of unfixed variables such that $|S_\rho| \geq \beta > n^{0.99}$ and for at least a half of the assignments σ to unfixed variables not in S_ρ, the function $f\restriction_{\rho\sigma}$ can be $4(\alpha_1(n) + \alpha_2(n))$-approximated by a circuit of size $|S_\rho|^{1+c\varepsilon}$, which completes the proof. □

Acknowledgement

We are greatly thankful to Yilei Chen for his support throughout this project. We thank Lijie Chen, Hanlin Ren, and Ryan Williams for the insightful discussion; and Lijie Chen for proofreading an earlier draft of the paper. We are grateful to Yuval Ishai for telling us recent results on low-complexity cryptography, and Roei Tell for helpful discussion on the black-box natural proof barrier. We thank Igor C. Oliveira for organizing a virtual seminar and valuable comments. We are thankful to Tianyi Zhang for addressing a typo in an earlier draft and Yiding Zhang for his help in improving some writing. We also thank anonymous reviewers for comments on the presentation of the paper.

References

[ABGKR14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. "Candidate weak pseudorandom functions in $\mathsf{AC}^0 \circ \mathsf{MOD}_2$". In: *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*. Ed. by Moni Naor. ACM, 2014, pp. 251–260. DOI: 10.1145/2554797.2554821. URL: https://doi.org/10.1145/2554797.2554821 (cit. on p. 10).

[AK10] Eric Allender and Michal Koucký. "Amplifying lower bounds by means of self-reducibility". In: *J. ACM* 57.3 (2010), 14:1–14:36. DOI: 10.1145/1706591.1706594. URL: https://doi.org/10.1145/1706591.1706594 (cit. on pp. 3, 7, 9, 15, 16).

[And87] A. E. Andreev. "On a method for obtaining more than quadratic effective lower bounds for the complexity of π-schemes". In: *Vestnik Moskov. Univ. Ser. 1. Mat. Mekh.* (1 1987), pp. 70–73 (cit. on pp. 8, 10, 20, 21).

[AHIKV17] Benny Applebaum, Naama Haramaty, Yuval Ishai, Eyal Kushilevitz, and Vinod Vaikuntanathan. "Low-Complexity Cryptographic Hash Functions". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*. Ed. by Christos H. Papadimitriou. Vol. 67. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 7:1–7:31. DOI: 10.4230/LIPIcs.ITCS.2017.7. URL: https://doi.org/10.4230/LIPIcs.ITCS.2017.7 (cit. on p. 62).

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. "Pseudorandom Functions and Lattices". In: *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. Lecture Notes in Computer Science. Springer, 2012, pp. 719–737. DOI: 10.1007/978-3-642-29011-4_42. URL: https://doi.org/10.1007/978-3-642-29011-4_42 (cit. on pp. 3, 35).

[Blu84] Norbert Blum. "A Boolean Function Requiring 3n Network Size". In: *Theor. Comput. Sci.* 28 (1984), pp. 337–345. DOI: 10.1016/0304-3975(83)90029-4. URL: https://doi.org/10.1016/0304-3975(83)90029-4 (cit. on p. 20).

[BR17] Andrej Bogdanov and Alon Rosen. "Pseudorandom Functions: Three Decades Later". In: *Tutorials on the Foundations of Cryptography*. Ed. by Yehuda Lindell. Springer International Publishing, 2017, pp. 79–158. DOI: 10.1007/978-3-319-57048-8_3. URL: https://doi.org/10.1007/978-3-319-57048-8_3 (cit. on pp. 3, 5).

[BIPSW18] Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu. "Exploring Crypto Dark Matter: - New Simple PRF Candidates and Their Applications". In: *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*. Ed. by Amos Beimel and Stefan Dziembowski. Vol. 11240. Lecture Notes in Computer Science. Springer, 2018, pp. 699–729. DOI: 10.1007/978-3-030-03810-6_25. URL: https://doi.org/10.1007/978-3-030-03810-6_25 (cit. on p. 10).

[Boy+21] Elette Boyle et al. "Low-Complexity Weak Pseudorandom Functions in $\mathsf{AC}^0[\mathsf{MOD2}]$". In: *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part IV*. Ed. by Tal Malkin and Chris Peikert. Vol. 12828. Lecture Notes in Computer Science. Springer, 2021, pp. 487–516. DOI: 10.1007/978-3-030-84259-8_17. URL: https://doi.org/10.1007/978-3-030-84259-8_17 (cit. on p. 10).

[Cha03] L. Sunil Chandran. "A High Girth Graph Construction". In: *SIAM J. Discret. Math.* 16.3 (2003), pp. 366–370. DOI: 10.1137/S0895480101387893. URL: https://doi.org/10.1137/S0895480101387893 (cit. on pp. 6, 12, 34).

[Che18] Lijie Chen. "Toward Super-Polynomial Size Lower Bounds for Depth-Two Threshold Circuits". In: *CoRR* abs/1805.10698 (2018). arXiv: 1805.10698. URL: http://arxiv.org/abs/1805.10698 (cit. on p. 10).

[CJW19] Lijie Chen, Ce Jin, and R. Ryan Williams. "Hardness Magnification for all Sparse NP Languages". In: *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*. Ed. by David Zuckerman. IEEE Computer Society, 2019, pp. 1240–1255. DOI: 10.1109/FOCS.2019.00077. URL: https://doi.org/10.1109/FOCS.2019.00077 (cit. on pp. 3, 4, 7, 9, 15, 16, 19, 21, 22).

[CJW20] Lijie Chen, Ce Jin, and R. Ryan Williams. "Sharp threshold results for computational complexity". In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. Ed. by Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy. ACM, 2020, pp. 1335–1348. DOI: 10.1145/3357713.3384283. URL: https://doi.org/10.1145/3357713.3384283 (cit. on pp. 3, 4, 7, 8, 9, 15, 16, 19, 21, 22).

[CT19] Lijie Chen and Roei Tell. "Bootstrapping results for threshold circuits "just beyond" known lower bounds". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*. Ed. by Moses Charikar and Edith Cohen. ACM, 2019, pp. 34–41. DOI: 10.1145/3313276.3316333. URL: https://doi.org/10.1145/3313276.3316333 (cit. on pp. 3, 4, 6, 7, 8, 9, 10, 11, 13, 15, 16, 18, 19, 21, 22, 35, 37, 38).

[CT21] Lijie Chen and Roei Tell. "Simple and fast derandomization from very hard functions: eliminating randomness at almost no cost". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 283–291. DOI: 10.1145/3406325.3451059. URL: https://doi.org/10.1145/3406325.3451059 (cit. on p. 24).

[Che+20] Lijie Chen et al. "Beyond Natural Proofs: Hardness Magnification and Locality". In: *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*. Ed. by Thomas Vidick. Vol. 151. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 70:1–70:48. DOI: 10.4230/LIPIcs.ITCS.2020.70. URL: https://doi.org/10.4230/LIPIcs.ITCS.2020.70 (cit. on pp. 3, 4, 7, 15, 17).

[CSS18] Ruiwen Chen, Rahul Santhanam, and Srikanth Srinivasan. "Average-Case Lower Bounds and Satisfiability Algorithms for Small Threshold Circuits". In: *Theory Comput.* 14.1 (2018), pp. 1–55. DOI: 10.4086/toc.2018.v014a009. URL: https://doi.org/10.4086/toc.2018.v014a009 (cit. on pp. 6, 8, 14, 16, 20, 23, 44, 48).

[CKLM20] Mahdi Cheraghchi, Valentine Kabanets, Zhenjian Lu, and Dimitrios Myrisiotis. "Circuit Lower Bounds for MCSP from Local Pseudorandom Generators". In: *ACM Trans. Comput. Theory* 12.3 (2020), 21:1–21:27. DOI: 10.1145/3404860. URL: https://doi.org/10.1145/3404860 (cit. on p. 21).

[DK11] Evgeny Demenkov and Alexander S. Kulikov. "An Elementary Proof of a 3n - o(n) Lower Bound on the Circuit Complexity of Affine Dispersers". In: *Mathematical Foundations of Computer Science 2011 - 36th International Symposium, MFCS 2011, Warsaw, Poland, August 22-26, 2011. Proceedings*. Ed. by Filip Murlak and Piotr Sankowski. Vol. 6907. Lecture Notes in Computer Science. Springer, 2011, pp. 256–265. DOI:

10.1007/978-3-642-22993-0_25. URL: https://doi.org/10.1007/978-3-642-22993-0_25 (cit. on pp. 8, 20).

[FGHK16] Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. "A Better-Than-3n Lower Bound for the Circuit Complexity of an Explicit Function". In: *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*. Ed. by Irit Dinur. IEEE Computer Society, 2016, pp. 89–98. DOI: 10.1109/FOCS.2016.19. URL: https://doi.org/10.1109/FOCS.2016.19 (cit. on pp. 8, 20).

[GHKPV13] Anna Gál, Kristoffer Arnsfelt Hansen, Michal Koucký, Pavel Pudlák, and Emanuele Viola. "Tight Bounds on Computing Error-Correcting Codes by Bounded-Depth Circuits With Arbitrary Gates". In: *IEEE Trans. Inf. Theory* 59.10 (2013), pp. 6611–6627. DOI: 10.1109/TIT.2013.2270275. URL: https://doi.org/10.1109/TIT.2013.2270275 (cit. on p. 32).

[GDP73] S. I. Gelfand, R. L. Dobrushin, and M. S. Pinsker. "On the Complexity of Coding". In: *Second International Symposium on Information Theory*. 1973, pp. 177–184 (cit. on pp. 11, 12, 32).

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. "How to Construct Random Functions (Extended Abstract)". In: *25th Annual Symposium on Foundations of Computer Science, West Palm Beach, Florida, USA, 24-26 October 1984*. IEEE Computer Society, 1984, pp. 464–479. DOI: 10.1109/SFCS.1984.715949. URL: https://doi.org/10.1109/SFCS.1984.715949 (cit. on pp. 3, 18, 20).

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions". In: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*. Ed. by David S. Johnson. ACM, 1989, pp. 25–32. DOI: 10.1145/73007.73010. URL: https://doi.org/10.1145/73007.73010 (cit. on p. 3).

[GW14] Oded Goldreich and Avi Wigderson. "On derandomizing algorithms that err extremely rarely". In: *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*. Ed. by David B. Shmoys. ACM, 2014, pp. 109–118. DOI: 10.1145/2591796.2591808. URL: https://doi.org/10.1145/2591796.2591808 (cit. on pp. 8, 9, 15, 16, 21, 22).

[Gor93] Daniel M. Gordon. "Discrete Logarithms in *GF(P)* Using the Number Field Sieve". In: *SIAM J. Discret. Math.* 6.1 (1993), pp. 124–138. DOI: 10.1137/0406010. URL: https://doi.org/10.1137/0406010 (cit. on p. 36).

[Hås86] Johan Håstad. "Almost Optimal Lower Bounds for Small Depth Circuits". In: *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*. Ed. by Juris Hartmanis. ACM, 1986, pp. 6–20. DOI: 10.1145/12130.12132. URL: https://doi.org/10.1145/12130.12132 (cit. on pp. 8, 16, 18).

[Hås98] Johan Håstad. "The Shrinkage Exponent of de Morgan Formulas is 2". In: *SIAM J. Comput.* 27.1 (1998), pp. 48–64. DOI: 10.1137/S0097539794261556. URL: https://doi.org/10.1137/S0097539794261556 (cit. on pp. 8, 10, 18, 20, 21, 22, 24, 25).

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "A Pseudorandom Generator from any One-way Function". In: *SIAM J. Comput.* 28.4 (1999), pp. 1364–1396. DOI: 10.1137/S0097539793244708. URL: https://doi.org/10.1137/S0097539793244708 (cit. on pp. 3, 20).

[HHTT21] Pooya Hatami, William Hoza, Avishay Tal, and Roei Tell. "Fooling Constant-Depth Threshold Circuits". In: *Electron. Colloquium Comput. Complex.* 28 (2021), p. 2. URL: https://eccc.weizmann.ac.il/report/2021/002 (cit. on pp. 6, 8, 14, 20, 44, 48).

[HS17] Shuichi Hirahara and Rahul Santhanam. "On the Average-Case Complexity of MCSP and Its Variants". In: *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*. Ed. by Ryan O'Donnell. Vol. 79. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 7:1–7:20. DOI: 10.4230/LIPIcs.CCC.2017.7. URL: https://doi.org/10.4230/LIPIcs.CCC.2017.7 (cit. on p. 22).

[IMZ19] Russell Impagliazzo, Raghu Meka, and David Zuckerman. "Pseudorandomness from Shrinkage". In: *J. ACM* 66.2 (2019), 11:1–11:16. DOI: 10.1145/3230630. URL: https://doi.org/10.1145/3230630 (cit. on pp. 8, 22, 24).

[IN93] Russell Impagliazzo and Noam Nisan. "The Effect of Random Restrictions on Formula Size". In: *Random Struct. Algorithms* 4.2 (1993), pp. 121–134. DOI: 10.1002/rsa.3240040202. URL: https://doi.org/10.1002/rsa.3240040202 (cit. on pp. 10, 18, 20).

[IPS93] Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. "Size-depth tradeoffs for threshold circuits". In: *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*. Ed. by S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal. ACM, 1993, pp. 541–550. DOI: 10.1145/167088.167233. URL: https://doi.org/10.1145/167088.167233 (cit. on pp. 10, 22, 26).

[IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. "Cryptography with constant computational overhead". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, 2008, pp. 433–442. DOI: 10.1145/1374376.1374438. URL: https://doi.org/10.1145/1374376.1374438 (cit. on pp. 3, 7, 9, 10, 13, 30).

[IM02] Kazuo Iwama and Hiroki Morizumi. "An Explicit Lower Bound of 5n - o(n) for Boolean Circuits". In: *Mathematical Foundations of Computer Science 2002, 27th International Symposium, MFCS 2002, Warsaw, Poland, August 26-30, 2002, Proceedings*. Ed. by Krzysztof Diks and Wojciech Rytter. Vol. 2420. Lecture Notes in Computer Science. Springer, 2002, pp. 353–364. DOI: 10.1007/3-540-45687-2_29. URL: https://doi.org/10.1007/3-540-45687-2_29 (cit. on p. 10).

[KRT17] Ilan Komargodski, Ran Raz, and Avishay Tal. "Improved Average-Case Lower Bounds for De Morgan Formula Size: Matching Worst-Case Lower Bound". In: *SIAM J. Comput.* 46.1 (2017), pp. 37–57. DOI: 10.1137/15M1048045. URL: https://doi.org/10.1137/15M1048045 (cit. on p. 21).

[KL01] Matthias Krause and Stefan Lucks. "On the Minimal Hardware Complexity of Pseudorandom Function Generators". In: *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science, Dresden, Germany, February 15-17, 2001, Proceedings*. Ed. by Afonso Ferreira and Horst Reichel. Vol. 2010. Lecture Notes in Com-

puter Science. Springer, 2001, pp. 419–430. DOI: 10.1007/3-540-44693-1_37. URL: https://doi.org/10.1007/3-540-44693-1_37 (cit. on p. 10).

[LY21] Jiatu Li and Tianqi Yang. "3.1n - o(n) Circuit Lower Bounds for Explicit Functions". In: *Electron. Colloquium Comput. Complex.* 28 (2021), p. 23. URL: https://eccc.weizmann.ac.il/report/2021/023 (cit. on pp. 8, 10, 20).

[MMW19] Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. "Weak lower bounds on resource-bounded compression imply strong separations of complexity classes". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*. Ed. by Moses Charikar and Edith Cohen. ACM, 2019, pp. 1215–1225. DOI: 10.1145/3313276.3316396. URL: https://doi.org/10.1145/3313276.3316396 (cit. on pp. 3, 7, 9, 15, 16).

[MV15] Eric Miles and Emanuele Viola. "Substitution-Permutation Networks, Pseudorandom Functions, and Natural Proofs". In: *J. ACM* 62.6 (2015), 46:1–46:29. DOI: 10.1145/2792978. URL: https://doi.org/10.1145/2792978 (cit. on pp. 3, 10, 18).

[MW20] Cody D. Murray and R. Ryan Williams. "Circuit Lower Bounds for Nondeterministic Quasi-polytime from a New Easy Witness Lemma". In: *SIAM J. Comput.* 49.5 (2020). DOI: 10.1137/18M1195887. URL: https://doi.org/10.1137/18M1195887 (cit. on p. 10).

[NR04] Moni Naor and Omer Reingold. "Number-theoretic constructions of efficient pseudo-random functions". In: *J. ACM* 51.2 (2004), pp. 231–262. DOI: 10.1145/972639.972643. URL: https://doi.org/10.1145/972639.972643 (cit. on pp. 3, 10, 35, 36).

[NRR00] Moni Naor, Omer Reingold, and Alon Rosen. "Pseudo-random functions and factoring (extended abstract)". In: *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*. Ed. by F. Frances Yao and Eugene M. Luks. ACM, 2000, pp. 11–20. DOI: 10.1145/335305.335307. URL: https://doi.org/10.1145/335305.335307 (cit. on pp. 3, 10).

[Nec66] E.I. Nechiporuk. "On a Boolean function". In: *Soviet Math. Dokl.* (4 1966), pp. 999–1000 (cit. on pp. 10, 20).

[OPS19] Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. "Hardness Magnification near State-Of-The-Art Lower Bounds". In: *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*. Ed. by Amir Shpilka. Vol. 137. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 27:1–27:29. DOI: 10.4230/LIPIcs.CCC.2019.27. URL: https://doi.org/10.4230/LIPIcs.CCC.2019.27 (cit. on pp. 3, 4, 7, 9, 15, 16, 19, 21, 22, 23).

[OS18] Igor Carboni Oliveira and Rahul Santhanam. "Hardness Magnification for Natural Problems". In: *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*. Ed. by Mikkel Thorup. IEEE Computer Society, 2018, pp. 65–76. DOI: 10.1109/FOCS.2018.00016. URL: https://doi.org/10.1109/FOCS.2018.00016 (cit. on pp. 3, 7, 9, 15, 16, 21).

[OSS19] Igor Carboni Oliveira, Rahul Santhanam, and Srikanth Srinivasan. "Parity Helps to Compute Majority". In: *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*. Ed. by Amir Shpilka. Vol. 137. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 23:1–23:17. DOI: 10.4230/LIPIcs.CCC.2019.23. URL: https://doi.org/10.4230/LIPIcs.CCC.2019.23 (cit. on p. 36).

[PZ93] Mike Paterson and Uri Zwick. "Shrinkage of de Morgan Formulae under Restriction". In: *Random Struct. Algorithms* 4.2 (1993), pp. 135–150. DOI: 10.1002/rsa.3240040203. URL: https://doi.org/10.1002/rsa.3240040203 (cit. on pp. 10, 18, 20).

[PS90] Ramamohan Paturi and Michael E. Saks. "On Threshold Circuits for Parity". In: *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*. IEEE Computer Society, 1990, pp. 397–404. DOI: 10.1109/FSCS.1990.89559. URL: https://doi.org/10.1109/FSCS.1990.89559 (cit. on p. 47).

[PS94] Ramamohan Paturi and Michael E. Saks. "Approximating Threshold Circuits by Rational Functions". In: *Inf. Comput.* 112.2 (1994), pp. 257–272. DOI: 10.1006/inco.1994.1059. URL: https://doi.org/10.1006/inco.1994.1059 (cit. on p. 26).

[Pau77] Wolfgang J. Paul. "A 2.5 n-Lower Bound on the Combinational Complexity of Boolean Functions". In: *SIAM J. Comput.* 6.3 (1977), pp. 427–443. DOI: 10.1137/0206030. URL: https://doi.org/10.1137/0206030 (cit. on p. 20).

[RRV02] Ran Raz, Omer Reingold, and Salil P. Vadhan. "Extracting all the Randomness and Reducing the Error in Trevisan's Extractors". In: *J. Comput. Syst. Sci.* 65.1 (2002), pp. 97–128. DOI: 10.1006/jcss.2002.1824. URL: https://doi.org/10.1006/jcss.2002.1824 (cit. on pp. 16, 19, 22).

[Raz87] Alexander A. Razborov. "Lower bounds on the size of constant-depth networks over a complete basis with logical addition". In: *Mathematicheskie Zametki* 41.4 (1987), pp. 598–607 (cit. on pp. 8, 21).

[RR97] Alexander A. Razborov and Steven Rudich. "Natural Proofs". In: *J. Comput. Syst. Sci.* 55.1 (1997), pp. 24–35. DOI: 10.1006/jcss.1997.1494. URL: https://doi.org/10.1006/jcss.1997.1494 (cit. on pp. 3, 7, 8, 9, 10, 15, 16, 17, 18, 20).

[RT92] John H. Reif and Stephen R. Tate. "On Threshold Circuits and Polynomial Computation". In: *SIAM J. Comput.* 21.5 (1992), pp. 896–908. DOI: 10.1137/0221053. URL: https://doi.org/10.1137/0221053 (cit. on p. 36).

[Smo87] Roman Smolensky. "Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity". In: *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*. Ed. by Alfred V. Aho. ACM, 1987, pp. 77–82. DOI: 10.1145/28395.28404. URL: https://doi.org/10.1145/28395.28404 (cit. on pp. 8, 21).

[Spi96] Daniel A. Spielman. "Linear-time encodable and decodable error-correcting codes". In: *IEEE Trans. Inf. Theory* 42.6 (1996), pp. 1723–1731. DOI: 10.1109/18.556668. URL: https://doi.org/10.1109/18.556668 (cit. on pp. 11, 30, 31, 32).

[Sto77] Larry J. Stockmeyer. "On the Combinational Complexity of Certain Symmetric Boolean Functions". In: *Math. Syst. Theory* 10 (1977), pp. 323–336. DOI: 10.1007/BF01683282. URL: https://doi.org/10.1007/BF01683282 (cit. on p. 20).

[Tal14] Avishay Tal. "Shrinkage of De Morgan Formulae by Spectral Techniques". In: *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*. IEEE Computer Society, 2014, pp. 551–560. DOI: 10.1109/FOCS.2014.65. URL: https://doi.org/10.1109/FOCS.2014.65 (cit. on pp. 8, 10, 18, 20, 21, 22, 24, 25).

[Tel17] Roei Tell. "A Note on the Limitations of Two Black-Box Techniques in Quantified Derandomization". In: *Electron. Colloquium Comput. Complex.* 24 (2017), p. 187. URL: https://eccc.weizmann.ac.il/report/2017/187 (cit. on pp. 7, 17).

[Tel18] Roei Tell. "Quantified derandomization of linear threshold circuits". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*. Ed. by Ilias Diakonikolas, David Kempe, and Monika Henzinger. ACM, 2018, pp. 855–865. DOI: 10.1145/3188745.3188822. URL: https://doi.org/10.1145/3188745.3188822 (cit. on pp. 3, 4, 6, 7, 14, 15, 16, 18, 20, 23, 44, 48).

[Tel21] Roei Tell. "How to Find Water in the Ocean: A Survey on Quantified Derandomization". In: *Electron. Colloquium Comput. Complex.* 28 (2021), p. 120. URL: https://eccc.weizmann.ac.il/report/2021/120 (cit. on pp. 7, 16, 17, 18).

[Tre01] Luca Trevisan. "Extractors and pseudorandom generators". In: *J. ACM* 48.4 (2001), pp. 860–879. DOI: 10.1145/502090.502099. URL: https://doi.org/10.1145/502090.502099 (cit. on pp. 16, 19, 22).

[Vio15] Emanuele Viola. "The communication complexity of addition". In: *Comb.* 35.6 (2015), pp. 703–747. DOI: 10.1007/s00493-014-3078-3. URL: https://doi.org/10.1007/s00493-014-3078-3 (cit. on pp. 3, 10, 36).

A The leftover lemma for Levin's trick

Lemma A.1. Let $m = m(n) = \Theta(n^\varepsilon)$ for $0 < \varepsilon < 1$. Assume that $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ is an almost universal hash function, then $\mathcal{B} = \{B_n\}_{n\geq 1}$ and $\mathcal{B}' = \{B'_n = \{f \circ h \mid f \in B_m, h \in H_n\}\}_{n\geq 1}$ are indistinguishable. ♢

Proof. Towards a contradiction we assume that $\mathcal{B}$ and $\mathcal{B}'$ are distinguishable, then there exists a p.p.t. adversary $\mathcal{A}$ distinguishing B and B'. Without loss of generality, we assume that $\mathcal{A}$ queries the oracle on exactly $t = n^d$ distinct points for some constant d.

Let $X_1, X_2, \ldots, X_t$ be random variables such that X_i denotes the i^{th} query point, h be the random variable denoting the hash function involved in $\mathcal{B}'$, and $\mathcal{E}_i$ be the event that $h(X_1), h(X_2), \ldots, h(X_i)$ are pairwisely distinct during the execution of $\mathcal{A}$. Clearly, we know that

$$\Pr_{f\leftarrow B_n,\mathcal{A}}[\mathcal{A}^f(1^n) = 1] = \Pr_{f'\leftarrow B'_n,\mathcal{A}}[\mathcal{A}^{f'}(1^n) = 1 \mid \mathcal{E}_t],$$

since under $\mathcal{E}_t$, f and f' are identically distributed. To obtain a contradiction, it is sufficient to show that

$$\Pr_{f'\leftarrow B',\mathcal{A}}[\neg\mathcal{E}_t] < \mathsf{negl}(n)$$

by elementary probability calculation. By union bound, we can see that

$$\Pr_{f'\leftarrow B',\mathcal{A}}[\neg\mathcal{E}_t] = \Pr_{f'\leftarrow B',\mathcal{A}}[\exists i, \neg\mathcal{E}_i \wedge \mathcal{E}_{i-1}] \leq \sum_{1\leq i\leq t} \Pr_{f'\leftarrow B',\mathcal{A}}[\neg\mathcal{E}_i \mid \mathcal{E}_{i-1}],$$

hence it is sufficient to show that $\Pr_{f'\leftarrow B',\mathcal{A}}[\neg\mathcal{E}_i \mid \mathcal{E}_{i-1}] \leq \mathsf{negl}(n)$.

Conditioning on the fact that $\mathcal{E}_{i-1}$ happens (the hash values of the first $i-1$ queries to the oracle are pairwisely distinct), the oracle returns independent random values. This means that the

adaptive adversary does not gain any advantage from the oracle queries, hence the probability that $\neg\mathcal{E}_i$ happens is exactly the collision probability of the hash function, i.e.

$$\begin{aligned}\Pr[\neg\mathcal{E}_i \mid \mathcal{E}_{i-1}] &= \Pr_{h\leftarrow H_n}[\exists j < i, h(x_i) = h(x_j) \mid \mathcal{E}_{i-1}] \\ &\leq \sum_{1\leq j<i} \Pr[h(x_i) = h(x_j) \mid \mathcal{E}_{i-1}] \\ &= \sum_{1\leq j<i} \Pr[h(x_i) = h(x_j) \mid x_i \neq x_j] \\ &\leq \sum_{1\leq j<i} \mathsf{negl}(n) \\ &\leq \mathsf{negl}(n),\end{aligned}$$

which completes the proof. □

Remark. From the proof one can see that $\mathcal{B}$ and $\mathcal{B}'$ are indistinguishable even if the adversary is not computational bounded, as long as it can perform oracle query for only polynomially many times.

B Proof of Lemma 5.6

Reminder of Lemma 5.6. Let $t = \omega(\log n)$ and $d \geq 3$. There exists a constant $\varepsilon \in (0,1)$, such that for $r = \Theta(n^{\varepsilon})$ and $m = \Theta(n^{1-\varepsilon/2})$, with probability at least $1 - n^{-0.1k}$, Algorithm 1 generates a good graph (and therefore a 1-detector) for sufficiently large n. ♢

Reminder of Algorithm 1: Generating good graphs

```
for i = 1, 2, ..., t do
    Let G ← (V_1 ∪ V_2, ∅) be an empty graph;
    for v ∈ V_1, j = 1, 2, ..., d do
        Link a random edge e_{v,j} = (v, v') for v' ← V_2;
    end
    if ∀S ⊆ V_1 of size ≤ k, there exists v' ∈ V_2 connects to odd number of vertices in S then
        return G;
    end
end
return ⊥;
```

To analyze this algorithm, we will separately bound the probability that it returns $\perp$ and it returns a graph that is not good. In both of the case, our analysis is similar to the standard probabilistic argument while proving the existence of good graphs.

Proposition B.1. Let $d \geq 3$ and $k \geq 1$ be constants, $t = \omega(\log n)$. For any $0 < \varepsilon < 2 - 4/d$ and $m = \Theta(n^{1-\varepsilon/2})$, the algorithm returns $\perp$ with negligible probability. ♢

Proof. We bound the probability of refusing to return the graph G in each iteration of the main loop. This will happen when there exists a subset $S \subseteq V_1$ of size at most k, such that each $v' \in V_2$ connects to even number of vertices in S. Since there are at most $d|S|$ wires connecting to vertices in S, at most $d|S|/2$ vertices in V_2 connect to more than 2 vertices in S. This means that if some

subset $S \subseteq V_1$ of size at most k spans more than $d|S|/2$ vertices in V_2, then there must exist some $v' \in V_2$ with exactly 1 incidence in S. Hence we can calculate the probability at follows.

$$\begin{aligned}
&\Pr[\exists |S| \le k, \forall v' \in V_2, v' \text{ connects to even \# of vertices in } S] \\
\le &\sum_{S \subseteq V_1, |S| \le k} \left[\sum_{T \subseteq V_2, |T| = \lfloor d|S|/2 \rfloor} \Pr[S \text{ only connects to } T] \right] \\
\le &\sum_{i=1}^{k} \binom{n}{i} \binom{m}{\lfloor di/2 \rfloor} \left(\frac{\lfloor di/2 \rfloor}{m} \right)^{di} \\
\le &\sum_{i=1}^{k} \left(\frac{ne}{i} \right)^{i} \left(\frac{2me}{di} \right)^{di/2} \left(\frac{di}{2m} \right)^{di} \\
= &\sum_{i=1}^{k} \left(i^{d/2-1} t \right)^{i}. \qquad \left(t \triangleq (ne) \left(\tfrac{2me}{d} \right)^{d/2} \left(\tfrac{d}{2m} \right)^{d} \right)
\end{aligned}$$

Since both d and k are constants, clearly $t = \Theta(nm^{-d/2})$. If $\varepsilon < \min\{2 - 4/d, 1\}$, then $t = o(1)$. In such case, the probability that the algorithm refuses to return G in each iteration is at most $1/2$ for sufficiently large n, which is reduced to negligible for $t = \omega(\log n)$ repetitions. □

Proposition B.2. Let $d \ge 3$ be a constant, $0 < \varepsilon < \frac{2d-5}{3d-4}$, $r = \Theta(n^{\varepsilon})$ and $m = \Theta(n^{1-\varepsilon/2})$. Conditioned on the fact that the algorithm does not return $\perp$, the probability that it outputs a good graph is at least $1 - n^{-0.2k}$ for sufficiently large n. ♢

Proof. Note that a graph $G = (V_1 \cup V_2, E)$ is not good if and only if there exists a set $S \subseteq V_1$ of size $\le r$ such that every vertex in V_2 connects to even number of vertices in S. Let $\mathcal{E}$ be the event that the algorithm outputs $\perp$. Condition on $\neg\mathcal{E}$, a graph is not good if there exists such a set S with $k < |S| \le r$ (see Line 6 to 8). Similar to Proposition B.1, we can calculate the probability as follows.

$$\begin{aligned}
&\Pr[G \text{ is not good} \mid \mathcal{E}] \\
\le &\sum_{i=k+1}^{r} \left(\frac{ne}{i} \right)^{i} \left(\frac{2me}{di} \right)^{di/2} \left(\frac{di}{2m} \right)^{di} \\
= &\sum_{i=k+1}^{r} \left(i^{d/2-1} t \right)^{i}. \qquad \left(t \triangleq (ne) \left(\tfrac{2me}{d} \right)^{d/2} \left(\tfrac{d}{2m} \right)^{d} \right)
\end{aligned}$$

As before, we have $t = \Theta(nm^{-d/2})$. Note that $i \le r = \Theta(n^{\varepsilon})$, if

$$1 - \frac{d(1-\varepsilon/2)}{2} + \varepsilon \left(\frac{d}{2} - 1 \right) < -\frac{1}{4},$$

that is $\varepsilon < \frac{2d-5}{3d-4}$, then $i^{d/2-1} t = o(n^{-0.2})$. In such case, for sufficiently large n, the probability that G is not good condition on $\mathcal{E}$ is at most

$$\sum_{i=k+1}^{\infty} \frac{1}{n^{0.2i}} \le n^{-0.2k}.$$

□

Then our proof for Lemma 5.6 follows directly.

Proof of Lemma 5.6. Let $\varepsilon \triangleq \frac{1}{2}\min\{2-4/d, (2d-5)/(3d-4)\}$. Assume that $\mathcal{G}$ is the event that the algorithm outputs a good graph and $\mathcal{E}$ is the event that the algorithm outputs $\perp$, then

$$\Pr[\neg\mathcal{G}] \leq \Pr[\mathcal{E}] + \Pr[\neg\mathcal{G} \mid \neg\mathcal{E}] \leq \mathsf{negl}(n) + n^{-0.2k} \leq n^{-0.1k}$$

according to Proposition B.1 and B.2. □

Remark. From the proof we can clearly see that the *if* clause in Line 6 to 8 is the key to amplify the success probability. A natural question is whether there exists a p.p.t. algorithm checking if a graph is good or not, since such algorithm would further reduce the error probability to negligible instead of polynomial. Unfortunately, such algorithm may not exist, if solving binary analogy of shortest vector problem (binarySVP) is hard for random sparse matrix. Such assumption is used to construct low complexity collision resistant hash function by Applebaum, Haramaty, Ishai, Kushilevitz, and Vaikuntanathan [AHIKV17]. Intuitively, Line 6 to 8 of our algorithm solves binarySVP with width $\leq k = O(1)$, which is sufficient since the error probability is mainly contributed by small width terms.

Reminder of Corollary 5.7. For some constant $\varepsilon \in (0,1)$, let $m = m(n) = \Theta(n^{1-\varepsilon/2})$, there exists an almost universal hash function $\mathcal{H} = \{H_n \subseteq B_{n,m}\}_{n\geq 1}$ with weakly uniform complexity $3n$. ♢

Proof. Let $d = 3$, $\varepsilon \in (0,1)$ be a constant given by Lemma 5.6 and $r = \Theta(n^{\varepsilon})$. We firstly define a p.p.t. sampling algorithm $\mathcal{G}$ for the hash function and then write down the explicit form $\{H_n \subseteq B_{n,m}\}_{n\geq 1}$ (and the distribution $\mathcal{D}_n$ over H_n). Given parameters n and c (where the desired success probability is n^{-c}), our algorithm firstly runs Algorithm 1 with $k = 10c$ and obtain a depth-1 XOR circuit C with unbounded fan-in (if Algorithm 1 returns $\perp$, we immediately output $\perp$). We expand C to a B_2 circuit of size $3n$ by realizing each XOR gate with a tree of fan-in 2 XOR gates. Then, similar to Lemma 5.5, we randomly choose a subset $S \subseteq [n]$ of size $s = \Theta(n^{1-\varepsilon/2})$, say $S = \{i_1, i_2, \ldots, i_s\}$, and label $x_{i_1}, x_{i_2}, \ldots, x_{i_s}$ to be output bits. The resulting circuit is the output of our algorithm.

Now we write down the explicit form. Let $\mathcal{E}$ be the event that Algorithm 1 generates a good graph. Let C_n be the set of circuits that are generated by $\mathcal{G}(1^n, k=0)$ with non-zero probability condition on $\mathcal{E}$, and $H_n \triangleq \{h \mid \exists C \in C_n, C \text{ computes } h\}$. The distribution $\mathcal{D}_n$ over H_n is defined as

$$\mathcal{D}_n(h) \triangleq \Pr_{\mathcal{G}}[\mathcal{G}(1^n, c=0) \text{ outputs a circuit computing } h \mid \mathcal{E}]$$

for all $h \in H_n$. By Lemma 5.6, we know that $\Pr[\mathcal{E}] \geq 1 - n^{-0.1k} = 1 - n^{-c}$. We can easily see that the parameter k does not influence the output distribution of Algorithm 1 condition on $\mathcal{E}$, hence for all positive integer c,

$$\mathcal{D}_n(h) = \Pr_{\mathcal{G}}[\mathcal{G}(1^n, c) \text{ outputs a circuit computing } h \mid \mathcal{E}].$$

This means that the family $\mathcal{H} = \{H_n\}_{n\geq 1}$ is weakly uniform of complexity $3n$.

Finally it is sufficient to show that $\mathcal{H}$ is actually a hash function. According to the definition, one can see that a random function $h \leftarrow H_n$ can be represented by a pair (C, S) of random variables, where C is a (n, r, m) 1-detector and S is a subset of $[n]$ of size s. Similar to Lemma 5.5, we can see that for each fixed C and any $x \neq y$, $\Pr[h(x) = h(y)] < \mathsf{negl}(n)$. Hence it directly implies that $\Pr_{h=(C,S)\leftarrow H_n}[h(x) = h(y)] < \mathsf{negl}(n)$, which completes the proof. □

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