Query Complexity Lower Bounds for Local List-Decoding and Hard-Core Predicates (Even for Small Rate and Huge Lists)

Noga Ron-Zewi
University of Haifa, Israel
noga@cs.haifa.ac.il

Ronen Shaltiel
University of Haifa, Israel
ronen@cs.haifa.ac.il

Nithin Varma
University of Haifa, Israel
nvarma@bu.edu

Abstract

A binary code $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ is $(\frac{1}{2}-\epsilon, L)$-*list decodable* if for every $w \in \{0,1\}^n$, there exists a set $\mathrm{List}(w)$ of size at most L, containing all messages $m \in \{0,1\}^k$ such that the relative Hamming distance between $\mathrm{Enc}(m)$ and w is at most $\frac{1}{2}-\epsilon$. A q-query *local list-decoder* for Enc is a randomized procedure Dec that when given oracle access to a string w, makes at most q oracle calls, and for every message $m \in \mathrm{List}(w)$, with high probability, there exists $j \in [L]$ such that for every $i \in [k]$, with high probability, $\mathrm{Dec}^w(i,j) = m_i$.

We prove lower bounds on q, that apply even if L is huge (say $L = 2^{k^{0.9}}$) and the rate of Enc is small (meaning that $n \geq 2^k$):

- For $\epsilon = 1/k^{\nu}$ for some constant $0 < \nu < 1$, we prove a lower bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$, where δ is the error probability of the local list-decoder. This bound is tight as there is a matching upper bound by Goldreich and Levin (STOC 1989) of $q = O(\frac{\log(1/\delta)}{\epsilon^2})$ for the Hadamard code (which has $n = 2^k$). This bound extends an earlier work of Grinberg, Shaltiel and Viola (FOCS 2018) which only works if $n \leq 2^{k^{\nu}}$ and the number of coins tossed by Dec is small (and therefore does not apply to the Hadamard code, or other codes with low rate).
- For smaller ϵ, we prove a lower bound of roughly $q = \Omega(\frac{1}{\sqrt{\epsilon}})$. To the best of our knowledge, this is the first lower bound on the number of queries of local list-decoders that gives $q \geq k$ for small ϵ.

Local list-decoders with small ϵ form the key component in the celebrated theorem of Goldreich and Levin that extracts a *hard-core predicate* from a one-way function. We show that black-box proofs *cannot* improve the Goldreich-Levin theorem and produce a hard-core predicate that is hard to predict with probability $\frac{1}{2} + \frac{1}{\ell^{\omega(1)}}$ when provided with a one-way function $f : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$, where f is such that circuits of size $\mathrm{poly}(\ell)$ cannot invert f with probability $\rho = 1/2^{\sqrt{\ell}}$ (or even $\rho = 1/2^{\Omega(\ell)}$). This limitation applies to any proof by black-box reduction (even if the reduction is allowed to use nonuniformity and has oracle access to f).

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Local list-decoding, Hard-core predicates, Black-box reduction, Hadamard code

Digital Object Identifier 10.4230/LIPIcs.ITCS.2021.33

Related Version A full version of the paper is available at https://eccc.weizmann.ac.il/report/2020/133/.

12th Innovations in Theoretical Computer Science Conference (ITCS 2021).
Editor: James R. Lee; Article No. 33; pp. 33:1–33:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Funding *Noga Ron-Zewi*: Partially supported by ISF grant 735/20.
Ronen Shaltiel: Partially supported by ISF grant 1628/17.
Nithin Varma: Partially supported by ISF grant 497/17, and Israel PBC Fellowship for Outstanding Postdoctoral Researchers from India and China.

Acknowledgements We are grateful to Ilan Newman for participating in early stages of this research and for many helpful discussions.

1 Introduction

We prove limitations on local list-decoding algorithms and on reductions establishing hard-core predicates.

1.1 Locally list-decodable codes

List-decodable codes are a natural extension of (uniquely decodable) error-correcting codes, as it allows (list) decoding for error regimes where unique decoding is impossible. This is an extensively studied area; see [8] for a survey. In this paper, we will be interested in list-decoding of binary codes.

▶ **Definition 1** (List-decodable code). *For a function* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$*, and* $w \in \{0,1\}^n$*, we define*

$$\mathrm{List}_\alpha^{\mathrm{Enc}}(w) = \left\{ m \in \{0,1\}^k : \mathsf{dist}(\mathrm{Enc}(m), w) \le \alpha \right\}.^{1}$$

We say that Enc *is* (α, L)*-list-decodable if for every* $w \in \{0,1\}^n$*,* $|\mathrm{List}_\alpha^{\mathrm{Enc}}(w)| \le L$*.*

The task of *algorithmic* list-decoding is to produce the list $\mathrm{List}_\alpha^{\mathrm{Enc}}(w)$ on input $w \in \{0,1\}^n$.

Local unique decoding algorithms are algorithms that given an index $i \in [k]$, make few oracle queries to w, and reproduce the bit m_i (with high probability over the choice of their random coins), where m denotes the unique codeword close to w. This notion of *local decoding* has many connections and applications in computer science and mathematics [18].

We will be interested in *local* list-decoding algorithms that receive oracle access to a received word $w \in \{0,1\}^n$, as well as inputs $i \in [k]$ and $j \in [L]$. We will require that for every $m \in \mathrm{List}_\alpha^{\mathrm{Enc}}(w)$, with high probability, there exists a $j \in [L]$ such that for every $i \in [k]$, when Dec receives oracle access to w and inputs i, j, it produces m_i with high probability over its choice of random coins. This motivates the next definition.

▶ **Definition 2** (Randomized local computation). *We say that a procedure* $P(i,r)$ *locally computes a string* $m \in \{0,1\}^k$ *with error* δ*, if for every* $i \in [k]$*,* $\Pr[P(i,R) = m_i] \ge 1 - \delta$ *(where the probability is over a uniform choice of the "string of random coins"* R*).*

The definition of local list-decoders considers an algorithmic scenario that works in two steps:

- At the first step (which can be thought of as a preprocessing step) the local list-decoder Dec is given oracle access to w and an index $j \in [L]$. It tosses random coins (which we denote by r^{shared}).
- At the second step, the decoder receives the additional index $i \in [k]$, and tosses additional coins r.

[1] For two strings $x, y \in \{0,1\}^n$ we use $\mathsf{dist}(x,y)$ to denote the relative Hamming distance between x and y, namely, $\mathsf{dist}(x,y) = |\{i \in [n] : x_i \ne y_i\}|/n$.

- It is required that for every $w \in \{0,1\}^n$ and $m \in \mathrm{List}_\alpha^{\mathrm{Enc}}(w)$, with probability $2/3$ over the choice of the shared coins r^{shared}, there exists $j \in [L]$ such that when the local list-decoder receives j, it locally computes m (using its "non-shared" coins r). The definition uses two types of random coins because the coins r^{shared} are "shared" between different choices of $i \in [k]$ and allow different i's to "coordinate". The coins r, are chosen independently for different choices of $i \in [k]$.

This is formally stated in the next definition.

▶ **Definition 3** (Local list-decoder). *Let* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *be a function. An* (α, L, q, δ)-*local list-decoder (LLD) for* Enc *is an oracle procedure* $\mathrm{Dec}^{(\cdot)}$ *that receives oracle access to a word* $w \in \{0,1\}^n$*, and makes at most* q *calls to the oracle. The procedure* Dec *also receives inputs:*

- $i \in [k]$ *: The index of the symbol that it needs to decode.*
- $j \in [L]$ *: An index to the list.*
- *Two strings* r^{shared}, r *that are used as random coins.*

It is required that for every $w \in \{0,1\}^n$*, and for every* $m \in \mathrm{List}_\alpha^{\mathrm{Enc}}(w)$*, with probability at least* $2/3$ *over choosing a uniform string* r^{shared}*, there exists* $j \in [L]$ *such that the procedure*

$$P_{w,j,r^{\mathrm{shared}}}(i, r) = \mathrm{Dec}^w(i, j, r^{\mathrm{shared}}, r)$$

locally computes m *with error* δ*. If we omit* δ*, then we mean* $\delta = 1/3$*.*

▶ Remark 4 (On the generality of Definition 3). The goal of this paper is to prove lower bounds on local list-decoders, and so, making local list-decoders as general as possible, makes our results stronger. We now comment on the generality of Definition 3.

- In Definition 3 we do not require that $L = |\mathrm{List}_\alpha^{\mathrm{Enc}}(w)|$, and allow the local list-decoder to use a larger L. This means that on a given w, there may be many choices of $j \in [L]$ such that the procedure $P_{w,j,r^{\mathrm{shared}}}(i, r) = \mathrm{Dec}^w(i, j, r^{\mathrm{shared}}, r)$ locally computes messages $m \notin \mathrm{List}_\alpha^{\mathrm{Enc}}(w)$.
- In Definition 3 we do not place any restriction on the number of random coins used by the local list-decoder, making the task of local list-decoding easier.
- We allow Dec to make *adaptive* queries to its oracle.
- We are only interested in the total number of queries made by the combination of the two steps. It should be noted that w.l.o.g., a local list-decoder can defer all its queries to the second step (namely, after it receives the input i), and so, this definition captures local list-decoding algorithms which make queries to the oracle at both steps.
- To the best of our knowledge, all known local list-decoders in the literature are of the form presented in Definition 3.

1.1.1 Lower bounds on the query complexity of local list-decoders

In this paper we prove lower bounds on the number of queries q of $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoders. Our goal is to show that the number of queries q has to be large, when ϵ is *small.* Our lower bounds apply even if the size of the list L is huge and approaches 2^k (note that a local list-decoder can trivially achieve $L = 2^k$ with a list of all messages). Our lower bounds also apply even if the rate of the code is very small, and $n \geq 2^k$.

We remark that this parameter regime is very different than the one studied in lower bounds on the number of queries of local decoders for *uniquely* decodable codes (that is, for $L = 1$). By the Plotkin bound, uniquely decodable codes cannot have $\epsilon < \frac{1}{4}$, and so, the main focus in uniquely decodable codes is to show that local decoders for codes with "good rate" and "large" $\epsilon = \Omega(1)$, must make many queries. In contrast, we are interested in the case where ϵ is small, and want to prove lower bounds that apply to huge lists and small rate.

Lower bounds for large ϵ

Our first result is a tight lower bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ on the number of queries, assuming ϵ is sufficiently large, namely $\epsilon \geq \frac{1}{k^\nu}$ for some constant $0 < \nu \leq 1$).

▶ **Theorem 5** (Tight lower bounds for large ϵ). *There exists a universal constant $\nu > 0$ such that for any $L \leq 2^{k^{0.9}}$, $\epsilon \in (k^{-\nu}, \frac{1}{4})$, and $\delta \in (k^{-\nu}, \frac{1}{3})$, we have that every $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoder for* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *must have $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$.*

Theorem 5 is tight in the sense that the Hadamard code (which has length $n = 2^k$) has $(\frac{1}{2} - \epsilon, L = O(1/\epsilon^2), q = O(\frac{\log(1/\delta)}{\epsilon^2}), \delta)$ local list-decoders [6]. In fact, the Hadamard code was the motivation for this research, and is a running example in this paper.

Our results show that even if we allow list sizes L which approach 2^k, it is impossible to reduce the number of queries for the Hadamard code. Our results also show that even if we are willing to allow very small rate $n \geq 2^k$, and huge list sizes, it is impossible to have codes whose local list-decoders make fewer queries than the local list-decoders for the Hadamard code.

Comparison to previous work

Theorem 5 improves and extends an earlier work by Grinberg, Shaltiel and Viola [7] that gave the same bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ for a more limited parameter regime: Specifically, in [7], for the lower bound to hold, it is also required that $n \leq 2^{k^\nu}$, for some constant $\nu > 0$, and that the total number of coins tossed by the local list-decoder is less than $k^\nu - \log L$.[2] We stress that because of these two limitations, the lower bounds of [7] do not apply to the Hadamard code and other low rate codes.

Extensions to large alphabet and erasures

The scenario that we consider in Theorem 5 has *binary* alphabet, and decoding from *errors*. We remark that in the case of large alphabets, or decoding from erasures, there are local list-decoders which achieve $q = O(\frac{\log(1/\delta)}{\epsilon})$ (which is smaller than what is possible for binary alphabet and decoding from errors), as was shown for the case of Hadamard codes in [11]. Our results extend to give a matching lower bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon})$ for decoding from erasures (for any alphabet size), and also the same lower bound on decoding from errors for any alphabet size. The exact details are deferred to the final version.

Lower bounds for small ϵ

The best bound on q that Theorem 5 (as well as the aforementioned lower bounds of [7]) can give is $q \geq k^{\Omega(1)}$. The next theorem shows that even for small $\epsilon < 1/k$, we can obtain a lower bound on q which is polynomial in $1/\epsilon$.

▶ **Theorem 6** (Lower bounds for small ϵ). *There exist universal constants $\beta, c_1, c_2 > 0$ such that for every $L \leq \beta \cdot 2^k$, $\delta < \frac{1}{3}$ and $\epsilon \geq \frac{\beta}{\sqrt{n}}$ we have that every $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoder for* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *must have $q \geq \frac{1}{c_1 \log(k) \cdot \sqrt{\epsilon}} - c_2 \log L$.*

[2] The work of [7] is concerned with proving lower bounds on the number of queries of "nonuniform reductions for hardness amplification" [17, 15, 3, 7]. As explained in [17, 15, 3, 7] such lower bounds translate into lower bounds on local list-decoders, by "trading" the random coins of a local list-decoder for "nonuniform advice" for the reduction, and proving a lower bound on the number of queries made by the reduction.

Note that for sufficiently small $\epsilon = 1/(\log k)^{\omega(1)}$, we get $q = \Omega(\frac{1}{\epsilon^{1/2-o(1)}})$. It follows that together, Theorems 5 and Theorem 6 give a lower bound of $q = \Omega(\frac{1}{\epsilon^{1/2-o(1)}})$ that applies to every choice of $\epsilon \geq \Omega(\frac{1}{\sqrt{n}})$. To the best of our knowledge, Theorem 6 is the first lower bound on local list-decoders that is able to prove a lower bound of $q \geq k$ (and note that this is what we should expect when $\epsilon < \frac{1}{k}$). We also remark that the requirement that ϵ is not too small compared to n (as is made in Theorem 6) is required (as we cannot prove lower bounds on the number of queries in case $\epsilon < \frac{1}{n}$).

Goldreich and Levin [6] showed that locally list-decodable codes with small $\epsilon < 1/k$ can be used to give constructions of hard-core predicates. We explain this connection in the next section.

1.2 Hard-core predicates

The celebrated Goldreich-Levin theorem [6] considers the following scenario: There is a computational task where the required output is *non-Boolean* and is hard to compute on average. We would like to obtain a *hard-core predicate*, which is a *Boolean* value that is hard to compute on average.

The Goldreich-Levin theorem gives a solution to this problem, and in retrospect, the theorem can also be viewed as a $(\frac{1}{2} - \epsilon, L^{\text{Had}} = O(\frac{1}{\epsilon^2}), q^{\text{Had}} = O(\frac{k}{\epsilon^2}), \delta = \frac{1}{2k})$-local list-decoder for the Hadamard code, defined by: $\text{Enc}^{\text{Had}} : \{0,1\}^k \to \{0,1\}^{n=2^k}$, where for every $r \in \{0,1\}^k$,

$$\text{Enc}^{\text{Had}}(x)_r = \left(\sum_{i \in [k]} x_i \cdot r_i\right) \mod 2.$$

In retrospect, the Goldreich-Levin theorem can also be seen as showing that *any* locally list-decodable code with suitable parameters can be used to produce hard-core predicates.

We consider two scenarios for the Goldreich-Levin theorem depending on whether we want to extract a hard-core bit from a function $g : \{0,1\}^\ell \to \{0,1\}^\ell$ that is *hard to compute* on a random input, or to extract a hard-core bit from a one-way function $f : \{0,1\}^\ell \to \{0,1\}^\ell$ that is *hard to invert* on a random output.

1.2.1 Functions that are hard to compute

Here the goal is to transform a *non-Boolean function* g that is hard to compute on a random input, into a *predicate* g^{pred} that is hard to compute on a random input. More precisely:

- *Assumption:* There is a non-Boolean function that is hard to compute with probability ρ. Namely, a function $g : \{0,1\}^\ell \to \{0,1\}^\ell$ such that for every circuit C of size s,

  $$\Pr_{x \leftarrow U_\ell}[C(x) = g(x)] \leq \rho.$$[3]

- *Conclusion:* There is a predicate $g^{\text{pred}} : \{0,1\}^{\ell'} \to \{0,1\}$ that is hard to compute with probability $\frac{1}{2} + \epsilon$.

  Namely, for every circuit C' of size s',

  $$\Pr_{x \leftarrow U_{\ell'}}[C'(x) = g^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

[3] We use U_ℓ to denote the uniform distribution on ℓ bits.

- *Requirements:* The goal is to show that for every g, there exists a function g^{pred} with as small an ϵ as possible, without significant losses in the other parameters (meaning that s' is not much smaller than s, and ℓ' is not much larger than ℓ).

The Goldreich-Levin theorem for this setting can be expressed as follows.

▶ **Theorem 7** (Goldreich-Levin for functions that are hard to compute [6])**.** *For every function $g : \{0,1\}^\ell \to \{0,1\}^\ell$, define $g^{\text{pred}} : \{0,1\}^{\ell'=2\ell} \to \{0,1\}$ by $g^{\text{pred}}(x,r) = \text{Enc}^{\text{Had}}(g(x))_r$, and $\rho = \frac{\epsilon}{2 \cdot L^{\text{Had}}} = \text{poly}(\epsilon)$. If for every circuit C of size s,*

$$\Pr_{x \leftarrow U_\ell}[C(x) = g(x)] \leq \rho,$$

then for every circuit C' of size $s' = \frac{s}{q^{\text{Had}} \cdot \text{poly}(\ell)} = s \cdot \text{poly}(\frac{\epsilon}{\ell})$,

$$\Pr_{x \leftarrow U_{2\ell}}[C'(x) = g^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

The Hadamard code can be replaced by any locally list-decodable code with list size L for decoding from radius $\frac{1}{2} - \epsilon$, with q queries for $\delta = 1/(2k)$. For such a code (assuming also that the local list-decoder can be computed efficiently) one gets the same behavior. Specifically, if the initial function is sufficiently hard and $\rho = \frac{\epsilon}{2L}$, then the Boolean target function is hard to compute, up to $\frac{1}{2} + \epsilon$ for circuits of size roughly $s' = s/q$.

Is it possible to improve the Goldreich-Levin theorem for $\rho \ll 1/s$?

Suppose that we are given a function $g : \{0,1\}^\ell \to \{0,1\}^\ell$ that is hard to compute for circuits of size $s = \text{poly}(\ell)$, with success say $\rho = 1/2^{\sqrt{\ell}}$. When applying Theorem 7, we gain nothing compared to the case that $\rho = 1/\text{poly}(\ell)$. In both cases, we can obtain $\epsilon = 1/\text{poly}(\ell)$, but not smaller! (Since otherwise $s' = s \cdot \text{poly}(\epsilon/\ell)$ is smaller than 1 and the result is meaningless).

This is disappointing, as we may have expected to obtain $\epsilon \approx \rho = 1/2^{\sqrt{\ell}}$, or at least, to gain over the much weaker assumption that $\rho = 1/\text{poly}(\ell)$. This leads to the following open problem:

▶ **Open problem 8** (Improve Goldreich-Levin for functions that are hard to compute)**.** *Suppose we are given a function $g : \{0,1\}^\ell \to \{0,1\}^\ell$ such that circuits C of size $s = \text{poly}(\ell)$ cannot compute g with success $\rho = 1/2^{\sqrt{\ell}}$. Is it possible to convert g into a predicate with hardness $\frac{1}{2} + \epsilon$ for $\epsilon = 1/\ell^{\omega(1)}$?*

This is not possible to achieve using the Hadamard code, because the number of queries is $q \geq 1/\epsilon$, and Theorem 7 requires $s \geq s' \cdot q \geq q \geq 1/\epsilon$, which dictates that $\epsilon \geq 1/s$.

Note that when ρ is small, we can afford list-decodable codes with huge list sizes of $L \approx 1/\rho$. Motivated by this observation, we can ask the following question:

> Is it possible to solve this open problem by substituting the Hadamard code with a better code? Specifically, is it possible for local list-decoders to have $q = \frac{1}{\epsilon^{o(1)}}$ if allowed to use *huge* lists of size say $2^{\sqrt{k}}$, approaching the trivial bound 2^k? (Note that in the Hadamard code, the list size used is $\text{poly}(1/\epsilon) = \text{poly}(k)$ which is exponentially smaller).

We show, in Theorem 6, that it is impossible to solve the open problem by replacing the Hadamard code with a different locally list-decodable code.

The natural next question is whether we can use other techniques (not necessarily local list-decoding) to achieve the goal stated above. In this paper, we show that this cannot be done by *black-box techniques*:

▶ **Informal Theorem 9** (Black-box impossibility result for functions that are hard to compute). *If $\rho \geq \frac{1}{2^{\ell/3}}$, $s = 2^{o(\ell)}$ is larger than some fixed polynomial in ℓ, and $\epsilon = \frac{1}{s^{\omega(1)}}$, then there does not exist a map that converts a function g into a function g^{pred} together with a black-box reduction showing that g^{pred} is a hard-core predicate for g.*

The parameters achieved in Theorem 9 rule out black-box proofs in which $\epsilon = \frac{1}{s^{\omega(1)}}$, not only for $s = \text{poly}(\ell)$ and $\rho = 2^{-\sqrt{\ell}}$ (as in Open problem 8) but also for $\rho = 2^{-\Omega(\ell)}$, and allowing much larger s as long as $s = 2^{o(\ell)}$.

The precise statement of Theorem 9 is stated in Theorem 19, and the precise model is explained in Section 3.1.

To the best of our knowledge, this is the first result of this kind, that shows black-box impossibility results for Open problem 8. Moreover, we believe that the model that we introduce in Section 3.1 is very general and captures all known black-box techniques. In particular, our model (which we view as a conceptual contribution) allows the reduction to introduce nonuniformity when converting an adversary C' that breaks g^{pred} into an adversary C that breaks g. See discussion in Remark 14.

1.2.2 Functions that are hard to invert

Here the goal is to transform a one-way function f into a new one-way function f^{newOWF} and a *predicate* f^{pred} such that it is hard to compute $f^{\text{pred}}(x)$ given $f^{\text{newOWF}}(x)$. More precisely:

- *Assumption:* There is a one-way function that is hard to invert with probability ρ.

  Namely, a function $f : \{0,1\}^\ell \to \{0,1\}^\ell$ such that for every circuit C of size s,

  $$\Pr_{x \leftarrow U_\ell}[C(f(x)) \in f^{-1}(f(x))] \leq \rho.$$

- *Conclusion:* There is a one-way function $f^{\text{newOWF}} : \{0,1\}^{\ell'} \to \{0,1\}^{\ell'}$, and a predicate $f^{\text{pred}} : \{0,1\}^{\ell'} \to \{0,1\}$, such that it is hard to predict $f^{\text{pred}}(x)$ with advantage $\frac{1}{2} + \epsilon$, when given access to $f^{\text{newOWF}}(x)$.

  Namely, for every circuit C' of size s',

  $$\Pr_{x \leftarrow U_{\ell'}}[C'(f^{\text{newOWF}}(x)) = f^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

- The goal is to show that for every f, there exist functions $f^{\text{newOWF}}, f^{\text{pred}}$ with as small ϵ as possible, without significant losses in the other parameters (meaning that: s' is not much smaller than s, and ℓ' is not much larger than ℓ).

The Goldreich-Levin theorem for this setting can be expressed as follows.

▶ **Theorem 10** (Goldreich-Levin for functions that are hard to invert). *For a function $f : \{0,1\}^\ell \to \{0,1\}^\ell$, define $f^{\text{newOWF}} : \{0,1\}^{2\ell} \to \{0,1\}^{2\ell}$ by $f^{\text{newOWF}}(x,r) = (f(x), r)$, $f^{\text{pred}} : \{0,1\}^{2\ell} \to \{0,1\}$ by $f^{\text{pred}}(x,r) = \text{Enc}^{\text{Had}}(x)_r$, and $\rho = \frac{\epsilon}{2 \cdot L^{\text{Had}}} = \text{poly}(\epsilon)$. If for every circuit C of size s,*

$$\Pr_{x \leftarrow U_\ell}[C(f(x)) \in f^{-1}(f(x))] \leq \rho,$$

then for every circuit C' of size $s' = \frac{s}{q^{\text{Had}} \cdot \text{poly}(\ell)} = s \cdot \text{poly}(\frac{\epsilon}{\ell})$,

$$\Pr_{x \leftarrow U_{2\ell}}[C'((f^{\text{newOWF}}(x))) = f^{\text{pred}}(x)] \leq \frac{1}{2} + \epsilon.$$

▶ Remark 11. The problem of obtaining a hard-core predicate for one-way functions, is interesting only if an unbounded adversary $\phi : \{0,1\}^{\ell'} \to \{0,1\}$ can predict $f^{\mathrm{pred}}(x)$ when given $f^{\mathrm{newOWF}}(x)$ as input. If this is not required, then one can take $\ell' = \ell + 1$, $f^{\mathrm{pred}}(x) = x_1$, and $f^{\mathrm{newOWF}}(x_1, \dots, x_{n+1}) = f(x_2, \dots, x_{n+1})$. However, this is trivial, and is not useful in applications. Therefore, when considering this problem, we will assume that there exists such a $\phi : \{0,1\}^{\ell'} \to \{0,1\}$.

A natural example is the case where the original one-way function f and the constructed function f^{newOWF} are one-way permutations. In fact, in the case that $f, f^{\mathrm{newOWF}}(x)$ are permutations, the setup of "functions that are hard to invert" can be seen as a special case of the setup of functions that are "hard to compute" by taking $g = f^{-1}$, and $g^{\mathrm{pred}}(y) = f^{\mathrm{pred}}((f^{\mathrm{newOWF}})^{-1}(y))$.

We point out that, in this setting, the circuit C' that is trying to invert f (that is, to compute g) has an advantage over its counterpart in the setup of "functions that are hard to compute": It can use the efficient algorithm that computes the "forward direction" of f, when trying to invert f. In terms of g, this means that the circuit C' can compute g^{-1} for free.

Is it possible to improve the Goldreich-Levin theorem for $\rho \ll 1/s$?

The same problem that we saw with functions that are hard to compute, also shows up in the setup of functions that are hard to invert. Suppose that we are given a function $f : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$ that is hard to invert for circuits of size $s = \mathrm{poly}(\ell)$ with success, say, $\rho = 1/2^{\sqrt{\ell}}$. When applying Theorem 10, we gain nothing compared to the case that $\rho = 1/\mathrm{poly}(\ell)$. In both cases, we can obtain $\epsilon = 1/\mathrm{poly}(\ell)$, but not smaller! This is expressed in the next open problem:

▶ **Open problem 12** (Improve Goldreich-Levin for functions that are hard to invert). *If we are given a one-way function $f : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$ such that circuits C of size $s = \mathrm{poly}(\ell)$ cannot invert f with success $\rho = 1/2^{\sqrt{\ell}}$. Is it possible to obtain a hard-core predicate f^{pred} with hardness $\frac{1}{2} + \epsilon$ for $\epsilon = 1/\ell^{\omega(1)}$ for some choice of one-way function f^{newOWF}?*

In this paper, we show that this cannot be done by *black-box techniques.* The formulation of Theorem 13 below, is very similar to that of Theorem 9 with some small modification in the parameters.

▶ **Informal Theorem 13** (Black-box impossibility result for functions that are hard to invert). *If $\rho \geq 2^{-\ell/5}$, $s = 2^{o(\ell)}$ is larger than some fixed polynomial in ℓ, and $\epsilon = \frac{1}{s^{\omega(1)}}$ then there does not exist a map that converts a function f into functions $f^{\mathrm{newOWF}}, f^{\mathrm{pred}}$ together with a black-box reduction showing that f^{pred} is a hard-core predicate for f^{newOWF}.*

To the best of our knowledge, this is the first result of this kind, that shows black-box impossibility results for open problem 12. Moreover, we believe that the model that we introduce is very general, and captures all known black-box techniques. In particular, our model (which we view as a conceptual contribution) allows the reduction to compute the easy direction of the function f, and to introduce nonuniformity when converting an adversary C' that breaks f^{pred} into an adversary C that breaks f.

▶ Remark 14 (The model we use for black-box proofs). Many different models of "black-box techniques" for cryptographic primitives were studied in the literature and the reader is referred to [12] for a discussion and a taxonomy. The model for "black-box technique" that we use is described in detail in Section 3. The notion that we use is incomparable to the ones discussed in [12], specifically:

- We require that there is a “transformation map” which given any function f produces functions f^{newOWF} and f^{pred}, however, unlike [12], we do not make the rquirement that this transformation map can be efficiently computed.
- We require that there is a reduction Red such that for any f, and for every adversary C' (not necessarily efficient) breaking the security of f^{pred}, $\text{Red}^{f,C'}$ can be used to invert f.
- However, we give reductions more power: We also allow Red to introduce nonuniformity (that could depend on C' and f). Formally, for every adversaty C' that breaks the security of f^{pred}, we require that there exists a short nonuniform advice string α such that $\text{Red}^{C',f}(\cdot,\alpha)$ inverts f.

1.3 More related work

Lower bounds on the number of queries of local decoders for *uniquely* decodable codes

In this paper, we prove lower bounds on the number of queries of local list-decoders. There is a long line of work that is concerned with proving lower bounds on the number of queries of uniquely decodable codes. As we have explained in Section 1.1.1, the parameter regime considered in the setting of uniquely decodable codes is very different than the parameter regime we consider here [18].

Lower bounds on nonuniform black-box reductions for hardness amplification

A problem that is closely related to proving lower bounds on the number of queries of local list-decoders is the problem of proving lower bounds on the number of queries of nonuniform black-box reductions for hardness amplification. We have already discussed this line of work [17, 15, 3, 7, 14] in Section 1.1.1.

Lower bounds on such reductions can be translated to lower bounds on local list-decoders (as long as the number of coins tossed by the local list-decoders is small). We remark that for the purpose of hardness amplification, it does not make sense to take codes with small rate (namely, codes with $n = 2^{k^{\Omega(1)}}$). The focus of Theorem 5 is to handle such codes.

Additionally, when using codes for hardness amplification, it does not make sense to take $\epsilon < 1/k$ (or even $\epsilon < 1/\sqrt{k}$). In contrast, the parameter regime considered in Theorem 6 focuses on small ϵ.

Motivated by hardness amplification, there is also a related line of work studying limitations on the complexity of local list decoders (and specifically, whether these decoders need to compute the majority function) [17, 15, 9, 3, 7, 14].

Another approach to prove limitations on hardness amplification is to show that assuming certain cryptographic assumptions, hardness amplification that is significantly better that what is currently known is impossible, see e.g., [5] for a discussion.

Other improvements of the Goldreich-Levin theorem

In this paper, we are interested in whether the Goldreich-Levin theorem can be improved. Specifically, we are interested in improvements where, when the original function has hardness $\rho = 2^{-\Omega(\ell)}$ for polynomial size circuits, then the hard-core predicate has hardness $\frac{1}{2}+\epsilon$ for $\epsilon = \ell^{-\omega(1)}$. We remark that there are other aspects of the Goldreich-Levin theorem that one may want to improve.

- When given an initial non-Boolean function on ℓ bits, the Goldreich-Levin theorem produces a hard-core predicate on $\ell' = 2\ell$ bits. It is possible to make ℓ' smaller (specifically, $\ell' = \ell + O(\log(1/\epsilon))$ by using other locally list-decodable codes instead of Hadamard. Our limitations apply to *any* construction (even one that is not based on codes) and in particular also for such improvements.
- It is sometimes desirable to produce many hard-core bits (instead of the single hard-core bit) that is obtained by a hard-core predicate. This can be achieved by using "extractor codes" with a suitable local list-decoding algorithm. The reader is referred to [16] for more details. Once again, our limitations obviously apply also for the case of producing many hard-core bits.

Organization of the paper

We give a high level overview of our technique in Section 2. Some of our results on hard-core predicates appear in Section 3 (which includes a precise description of the model and formal restatements of Theorem 9) We refer the interested reader to the full version [13] for a precise description of the model and formal restatement of Theorem 13. Section 4 contains some concluding remarks and open problems. All proofs can be found in the full version.

2 Technique

In this section we give a high level overview of our technique. Our approach builds on earlier work for proving lower bounds on the number of queries of reductions for hardness amplification [17, 15, 7]. In this section, we give a high level overview of the arguments used to prove our main theorems.

2.1 Local list-decoders on random noisy codewords

Following [17, 15, 7], we will consider a scenario which we refer to as "random noisy codewords" in which a uniformly chosen message m is encoded, and the encoding is corrupted by a binary symmetric channel.

▶ **Definition 15** (Binary symmetric channels). *Let* BSC_p^n *be the experiment in which a string* $Z \in \{0,1\}^n$ *is sampled, where* $Z = Z_1, \ldots, Z_n$ *is composed of i.i.d. bits, such that for every* $i \in [n]$, $\Pr[Z_i = 1] = p$.

▶ **Definition 16** (Random noisy codewords). *Given a function* $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$ *and* $p > 0$ *we consider the following experiment (which we denote by* $\mathrm{RNSY}_p^{\mathrm{Enc}}$*):*

- *A message* $m \leftarrow \{0,1\}^k$ *is chosen uniformly.*
- *A noise string* $z \leftarrow \mathrm{BSC}_p^n$ *is chosen from a binary symmetric channel.*
- *We define* $w = \mathrm{Enc}(m) \oplus z$.

We use $(m, z, w) \leftarrow \mathrm{RNSY}_p^{\mathrm{Enc}}$ *to denote* m, z, w *which are sampled by this experiment. We omit* Enc *if it is clear from the context.*

Our goal is to prove lower bounds on the number of queries q of a $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoder Dec for a code $\mathrm{Enc} : \{0,1\}^k \to \{0,1\}^n$. For this purpose, we will consider the experiment RNSY_p for the values $p = \frac{1}{2} - 2\epsilon$ and $p = \frac{1}{2}$.

For $p = \frac{1}{2} - 2\epsilon$, and $(m, z, w) \leftarrow \mathrm{RNSY}_{\frac{1}{2}-\epsilon}$, by a Chernoff bound, the Hamming weight of z is, with very high probability, less than $\frac{1}{2} - \epsilon$. This implies that $\mathsf{dist}(w, \mathrm{Enc}(m)) \le \frac{1}{2} - \epsilon$, meaning that $m \in \mathrm{List}_{\frac{1}{2}-\epsilon}^{\mathrm{Enc}}(w)$. It follows that there must exist $j \in [L]$ such that when given input j, and oracle access to w, the decoder Dec recovers the message m.

For $p = \frac{1}{2}$, and $(m, z, w) \leftarrow \mathrm{RNSY}_{\frac{1}{2}}$, the string z is uniformly distributed and independent of m. This means that $w = \mathrm{Enc}(m) \oplus z$ is uniformly distributed and independent of m. Consequently, when Dec is given oracle access to w, there is no information in w about the message m, and so, for every $j \in [L]$, the probability that Dec recovers m when given input j and oracle access to w is exponentially small.

Loosely speaking, this means that Dec can be used to distinguish $\mathrm{BSC}^n_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^n_{\frac{1}{2}}$. It is known that distinguishing these two distributions requires many queries. We state this informally below.

▶ **Informal Theorem 17.** *Any function $T : \{0,1\}^q \to \{0,1\}$ that distinguishes $\mathrm{BSC}^q_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^q_{\frac{1}{2}}$ with advantage δ, must have $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$.*

Thus, in order to prove a tight lower bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$, it is sufficient to show how to convert a $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoder Dec, into a function T that distinguishes $\mathrm{BSC}^q_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^q_{\frac{1}{2}}$ with advantage δ. Note that we can allow T to be a "randomized procedure" that tosses coins, as by an averaging argument, such a randomized procedure can be turned into a deterministic procedure.

2.2 Warmup: the case of unique decoding

Let us consider the case that $L = 1$ (that is unique decoding). We stress that this case is uninteresting, as by the Plotkin bound, it is impossible for nontrivial codes to be uniquely decodable for $\epsilon < \frac{1}{4}$, and so, there are no local decoders for $L = 1$ and $\epsilon < \frac{1}{4}$, regardless of the number of queries. Nevertheless, this case will serve as a warmup for the approach we use later.

Our goal is to convert Dec into a randomized procedure $T : \{0,1\}^q \to \{0,1\}$ that distinguishes $\mathrm{BSC}^q_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^q_{\frac{1}{2}}$. The procedure T will work as follows: On input $x \leftarrow \{0,1\}^q$, we choose $m \leftarrow \{0,1\}^k$, and $i \leftarrow [k]$. We then run Dec on input i, and when Dec makes its t'th query $\ell_t \in [n]$ to the oracle, we answer it by $\mathrm{Enc}(m)_{\ell_t} \oplus x_t$. That is, we answer as if Dec is run with input i and oracle access to $w = \mathrm{Enc}(m) \oplus z$, for z chosen from a binary symmetric channel. The final output of T is whether Dec reproduced m_i. This procedure T simulates $\mathrm{Dec}^w(i)$, and therefore distinguishes $\mathrm{BSC}^q_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^q_{\frac{1}{2}}$, implying the desired lower bound.

Both Theorem 5 and Theorem 6 will follow by modifying the basic approach to handle $L > 1$. In the remainder of this section, we give a high level overview of the methods that we use. The formal section of this paper does not build on this high level overview, and readers can skip this high level overview and go directly to the formal section if they wish.

2.3 Reducing to the coin problem for AC^0

We start with explaining the approach of proving Theorem 6. Consider a randomized procedure C that on input $z \in \{0,1\}^n$, chooses $m \leftarrow \{0,1\}^k$ and prepares $w = \mathrm{Enc}(m) \oplus z$. The procedure then computes $\mathrm{Dec}^w(i, j)$ for all choices of $i \in [k]$ and $j \in [L]$ and accepts if there exists a $j \in [L]$ such that $\mathrm{Dec}^w(\cdot, j)$ recovers m. By the same rationale as in Section 2.2, C distinguishes $\mathrm{BSC}^n_{\frac{1}{2}-2\epsilon}$ from $\mathrm{BSC}^n_{\frac{1}{2}}$. This does not seem helpful, because C receives n input bits, and we cannot use Theorem 17 to get a lower bound on q.

Inspired by a lower bound on the size of nondeterministic reductions for hardness amplification due to Applebaum et al. [2], we make the following observation: The procedure C can be seen as $k \cdot L$ computations (one for each choice of $i \in [k]$ and $j \in [L]$) such that:

- These $k \cdot L$ computations can be run *in parallel*.
- Once these computations are made, the final answer $C(z)$ is computed by a constant-depth circuit.
- Each of the $k \cdot L$ computations makes q queries into z, and therefore can be simulated by a size $O(q \cdot 2^q)$ circuit of depth 2.

Overall, this means that we can implement C by a circuit of size $s = \text{poly}(k, L, 2^q)$ and constant depth. (In fact, a careful implementation gives depth 3).

This is useful because there are lower bounds showing that small constant-depth circuits cannot solve the "coin problem". Specifically, by the results of Cohen, Ganor and Raz [4] circuits of size s and depth d cannot distinguish $\text{BSC}^n_{\frac{1}{2}-2\epsilon}$ from $\text{BSC}^n_{\frac{1}{2}}$ with constant advantage, unless $s \geq \exp(\Omega(\frac{1}{\epsilon^{d-1}}))$.[4] This gives the bound stated in Theorem 6.

We find it surprising that an *information theoretic* lower bound on the number of queries of local list-decoders is proven by considering concepts like constant-depth circuits from *circuit complexity*.

Extending the argument to lower bounds on hard-core predicates

It turns out that this argument is quite versatile, and this is the approach that we use to prove Theorems 9 and 13. Loosely speaking, in these theorems, we want to prove a lower bound on the number of queries made by a reduction that, when receiving oracle access to an adversary that breaks the hard-core predicate, is able to compute (or invert) the original function too well. Such lower bounds imply that such reductions do not produce small circuits when used in black-box proofs for hard-core predicates.

We will prove such lower bounds by showing that a reduction that makes q queries can be used to construct a circuit of size $s \approx 2^q$ and constant depth that solves the coin problem. Interestingly, this argument crucially relies on the fact that constant-depth circuits *can* distinguish BSC^n_{ϵ} from $\text{BSC}^n_{2\epsilon}$ with size $\text{poly}(n/\epsilon)$ which follows from the classical results of Ajtai on constant depth circuits for approximate majority [1].[5]

2.4 Conditioning on a good j

A disadvantage of the approach based on the coin problem is that at best, it can give lower bounds of $q = \Omega(1/\sqrt{\epsilon})$, and cannot give tight lower bounds of the form $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$. In order to achieve such a bound (as is the case in Theorem 5) we will try to reduce to Theorem 17 which *does* give a tight bound in case ϵ is not too small.

Our approach builds on the earlier work of Grinberg, Shaltiel and Viola [7] that we surveyed in Section 1.1.1. When given a $(\frac{1}{2} - \epsilon, L, q, \delta)$-local list-decoder Dec, we say that an index $j \in [L]$ is *decoding*, if in the experiment $(m, z, w) \in \text{RNSY}_{\frac{1}{2}-2\epsilon}$, when Dec is given oracle access to w and input j, then with probability $1 - 10\delta$ over $i \in [k]$, we have that $\text{Dec}^w(i, j)$ recovers m_i.

[4] These results of [4] improve upon earlier work of Shaltiel and Viola [15] that gave slightly worse bound. These results are tight as shown by Limaye et al. [10] (that also extended the lower bound to hold for circuits that are also allowed to use parity gates).

[5] The proof of Theorem 13 uses an additional versatility of the argument (which we express in the terminology of codes): The argument works even if the individual procedures that are run in parallel are allowed to have some limited access to the message m, as long as this does not enable them to recover m. This property is used to handle reductions in a cryptographic setup, where reductions have access to the easy direction of a one-way function.

We use a careful averaging argument to show that there exists an index $j' \in [L]$, and a fixed choice of the random coins of Dec, such that j' is decoding with probability at least $\Omega(1/L)$. We then consider the experiment $\mathrm{RNSY}'_{\frac{1}{2}-2\epsilon}$ in which we choose $(m, z, w) \leftarrow \mathrm{RNSY}_{\frac{1}{2}-2\epsilon}$ *conditioned* on the event $\{j'$ is decoding$\}$.

We have made progress, because in the experiment $\mathrm{RNSY}'_{\frac{1}{2}-2\epsilon}$ there exists a unique j' that is decoding, and so, when we implement the strategy explained in Section 2.2 we only need to consider this *single* j', which intuitively means that our scenario is similar to the warmup scenario of unique decoding described in Section 2.2.

The catch is that when choosing $(m, z, w) \leftarrow \mathrm{RNSY}'_{\frac{1}{2}-2\epsilon}$, we no longer have that z is distributed like $\mathrm{BSC}^n_{\frac{1}{2}-2\epsilon}$ (as the distribution of z may be skewed by conditioning on the event that j' is decoding).

Shaltiel and Viola [15] (and later work [7, 14]) developed tools to handle this scenario. Loosely speaking, using these tools, it is possible to show that a large number of messages m are "useful" in the sense that there exists an event A_m such that if we consider (m, z, w) that are chosen from $\mathrm{RNSY}'_{\frac{1}{2}-2\epsilon}$ *conditioned* on A_m, then there exists a subset $B(m) \subseteq [n]$ of small size b, such that $z_{B(m)}$ is fixed, and $z_{[n]\setminus B(m)}$ is distributed like $\mathrm{BSC}^{n-b}_{\frac{1}{2}-2\epsilon}$.

If the number of possible choices for sets $B(m)$ is small, then by the pigeon-hole principle, there exists a fixed choice B that is good for a large number of useful messages m. This can be used to imitate the argument we used in the warmup, and prove a lower bound.[6]

Extending the argument to the case of small rate

A difficulty, that prevented [7] from allowing length as large as $n = 2^k$, is that $B(m)$ is a subset of $[n]$, and so, even if $b = |B| = 1$, the number of possible choices for such sets is at least n. For the pigeon-hole principle argument above, we need that the number of messages (that is 2^k) is much larger than the number of possible choices for $B(m)$ (which is at least n). This means that one can only handle n which is sufficiently smaller than 2^k, and this approach cannot apply to codes with small rate (such as the Hadamard code).

We show how to solve this problem, and prove lower bounds for small rate codes. From a high level, our approach can be explained as follows: We consider the distribution of $B(m) = \{Y_1(m) < \ldots < Y_b(m)\}$ for a uniformly chosen useful m. We first show that if all the Y_j's have large min-entropy, then it is possible to prove a lower bound on q by reducing to Theorem 17 (the details of this are explained in the actual proof).

If on the other hand, one of the Y_j's has low min-entropy, then we will restrict our attention to a subset of useful messages on which Y_j is fixed. Loosely speaking, this reduces b by one, while not reducing the number of useful messages by too much (because the low min-entropy condition says that the amount of information that Y_j carries on m is small). In this trench warfare, in every iteration, we lose a fraction of useful messages, for the sake of decreasing b by one. Thus, eventually, we either reach the situation that all the Y_j's have large min-entropy, in which case we are done, or we reach the situation where B is fixed for all messages which we can also handle by the above.

We can withstand the losses and eventually win if ϵ is sufficiently larger than $1/\sqrt{k}$.

[6] Loosely speaking, this is because for good messages, in the conditioned experiment, z is distributed like $\mathrm{BSC}_{\frac{1}{2}-2\epsilon}$ (except that some bits of z are fixed as a function of m). Furthermore, as there are many good messages, the local list-decoder does not have enough information to correctly recover the message when given oracle access to $\mathrm{Enc}(m) \oplus \mathrm{BSC}^n_{\frac{1}{2}} = \mathrm{BSC}^n_{\frac{1}{2}}$.

3 Limitations on black-box proofs for hard-core predicates

In this section, we present some of our results regarding the limitations on black-box proofs for hard-core predicate theorems. Specifically, we state our results for functions that are hard to compute, give a formal restatement of Theorem 9. Due to space limitation, the formal model and precise statement of results for the case of functions that are hard to invert appear in the full version [13].

3.1 The case of functions that are hard to compute

3.1.1 The model for black-box proofs

In this section, we state and explain our model for black-box proofs for hard core predicates, in the setting of functions that are hard to compute. The formal definition is given in Definition 18. Below, we provide a detailed explanation for the considerations made while coming up with the formal definition. The reader can skip directly to the formal definition if he wishes.

Explanation of the model

Recall that (as explained in Section 1.2.1) the Goldreich-Levin theorem (stated precisely in Theorem 7) has the following form:

- We are given an arbitrary hard function $g : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$. (Intuitively, it is assumed that it is hard to compute g with success probability ρ).
- There is a specified construction that transforms g into a predicate $g^{\text{pred}} : \{0,1\}^{\ell'} \to \{0,1\}$ for some ℓ' related to ℓ. (Intuitively, we will want to argue that g^{pred} is a hard-core predicate that is hard to compute with success $\frac{1}{2} + \epsilon$).
  We will model this construction as a map Con, which, given g produces g^{pred}. We place *no limitations* on the map Con (and, in particular, do not require that g^{pred} can be efficiently computed if g is). This only makes our results stronger.
  In the case of Theorem 7, we have that: $\text{Con}(g) = g^{\text{pred}}$ where $\ell' = 2\ell$ and we think of the ℓ'-bit long input of g^{pred} as two strings $x, r \in \{0,1\}^{\ell}$, setting:

$$g^{\text{pred}}(x,r) = \text{Enc}^{\text{Had}}(g(x))_r = (\sum_{i \in [\ell]} g(x)_i \cdot r_i) \mod 2.$$

- We model the proof showing that g^{pred} is a hard-core predicate in the following way: The proof is a pair (Con, Red) where $\text{Red}^{(\cdot)}$ is an oracle procedure, such that when $\text{Red}^{(\cdot)}$ receives oracle access to an "adversary" $h : \{0,1\}^{\ell'} \to \{0,1\}$ that breaks the security of g^{pred}, we have that Red^h breaks the security of g. More precisely, we require that: for every $g : \{0,1\}^{\ell} \to \{0,1\}^{\ell}$ and for every $h : \{0,1\}^{\ell'} \to \{0,1\}$ such that:

$$\Pr_{x \leftarrow U_{\ell'}}[h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

  it holds that:

$$\Pr_{x \leftarrow U_{\ell}}[\text{Red}^h(x) = g(x)] \geq \rho.$$

- In the actual definition, we will allow the reduction to have more power (which only makes our results stronger). As we are aiming to prove a result on circuits (which are allowed to use nonuniform advice) we will allow the reduction to receive an advice string α of length t, where, this advice string can depend on g and h. This leads to the following strengthening of the requirement above. Namely, we will require that: for every $g : \{0,1\}^\ell \to \{0,1\}^\ell$ and for every $h : \{0,1\}^{\ell'} \to \{0,1\}$, that:

$$\Pr_{x \leftarrow U_{\ell'}}[h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

there exists $\alpha \in \{0,1\}^t$ such that:

$$\Pr_{x \leftarrow U_\ell}[\text{Red}^h(x,\alpha) = g(x)] \geq \rho.$$

We remark that in many related settings (for example, "hardness amplification"; see [15, 7], for a discussion) known proofs by reduction *critically make use* of the ability to introduce nonuniformity, and so, we feel that when ruling out black-box proofs in scenarios involving circuits, it is necessary to consider nonuniform black-box reductions.

- We make no restrictions on the complexity of the procedure $\text{Red}^{(\cdot)}$, except for requiring that it makes at most q queries to its oracle (for some parameter q). Our black-box impossibility results will follow from proving lower bounds on q.

Formal definition

We now give a formal definition of our model for black-box proofs for hard-core predicates.

▶ **Definition 18** (Nonuniform black-box proofs for hard-core predicates for hard-to-compute functions). *A pair* (Con, Red) *is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters $\ell, \ell', \rho, \epsilon$, that uses q queries, and t bits of advice if:*

- Con *is a* construction map *which given a function $g : \{0,1\}^\ell \to \{0,1\}^\ell$, produces a function* $\text{Con}(g) = g^{\text{pred}}$, *where $g^{\text{pred}} : \{0,1\}^{\ell'} \to \{0,1\}$.*
- $\text{Red}^{(\cdot)}$ *is a* reduction, *that is an oracle procedure that, given oracle access to a function $h : \{0,1\}^{\ell'} \to \{0,1\}$, makes at most q queries to its oracle.*

Furthermore, for every functions $g : \{0,1\}^\ell \to \{0,1\}^\ell$ and $h : \{0,1\}^{\ell'} \to \{0,1\}$ such that:

$$\Pr_{x \leftarrow U_{\ell'}}[h(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

there exists $\alpha \in \{0,1\}^t$, such that:

$$\Pr_{x \leftarrow U_\ell}[\text{Red}^h(x,\alpha) = g(x)] \geq \rho.$$

The role of the number of queries, and black-box impossibility results

We now explain the role of the parameter q (that measures the number of queries made by Red) and why lower bounds on q translate into black-box impossibility results.

For this purpose, it is illustrative to examine the argument showing that nonuniform black-box proofs yield hard-core predicates: When given a pair (Con, Red) that is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters $\ell, \ell', \rho, \epsilon$, that uses q queries, and t bits of advice, we obtain that for any function $g : \{0,1\}^\ell \to \{0,1\}^\ell$, if there exists a circuit $C' : \{0,1\}^{\ell'} \to \{0,1\}$ of size s' such that:

$$\Pr_{x \leftarrow U_{\ell'}}[C'(x) = g^{\text{pred}}(x)] \geq \frac{1}{2} + \epsilon,$$

then there exists $\alpha \in \{0,1\}^t$, such that:

$$\Pr_{x \leftarrow U_\ell}[\mathrm{Red}^{C'}(x,\alpha) = g(x)] \geq \rho.$$

Note that if the reduction Red can be implemented by a circuit of size r, then the circuit $C(x) = \mathrm{Red}^{C'}(x,\alpha)$ is a circuit of size:

$$s = r + t + q \cdot s'$$

that computes g with success probability ρ.

It follows that in a black-box proof, with q queries, and t bits of advice, we get a hard-core theorem that needs to assume that the original function g has hardness against circuits of size s, for:

$$s \geq q + t.$$

3.1.2 Precise statements of limitations

Our main result on black-box proofs for hard-core predicates in the setting of functions that are hard to compute is the following theorem.

▶ **Theorem 19.** *There exists a universal constant $\beta > 0$ such that for every sufficiently large ℓ and ℓ' we have that if* $(\mathrm{Con}, \mathrm{Red})$ *is a nonuniform black-box proof for hard-core predicates for hard-to-compute functions with parameters $\ell, \ell', \rho, \epsilon$, that uses q queries, and t bits of advice, and furthermore $\epsilon \geq \frac{1}{2^{\ell'/3}}$, $t \leq 2^{\ell/3}$ and $\rho \geq \frac{1}{2^{\ell/3}}$, then*

$$q \geq \Omega(\frac{1}{\epsilon^\beta}) - O(t + \ell).$$

We now explain why Theorem 19 implies the informal statement made in Theorem 9. Recall that in Section 3.1.1 we explained that when using a nonuniform black-box proof to obtain a hard-core predicate, we get a hard-core predicate theorem in which $s \geq q + t$.

Theorem 19 implies that for $s > \ell^{2/\beta}$ it is impossible for such a proof to establish $\epsilon = 1/s^{\frac{2}{\beta}}$ (even if ρ is very small). This follows as otherwise, using the fact that $s \geq q + t \geq t$, we get that:

$$q \geq \Omega(\frac{1}{\epsilon^\beta}) - O(t + \ell) \geq \Omega(s^2) - O(t) > s,$$

which is a contradiction to $s \geq q + t \geq q$. In particular, the parameter setting considered in Theorem 9, in which $s = 2^{o(\ell)}$ and $\epsilon = \frac{1}{s^{\omega(1)}}$, is impossible to achieve.

4 Conclusion and open problems

Unlike Theorem 5 (that handles large ϵ), Theorem 6 (that handles small ϵ) does not achieve a bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$, and only achieves a bound of $\Omega(\frac{1}{\sqrt{\epsilon}})$. A natural open problem is to improve the bound on q for small ϵ to match the bound for large ϵ.

In the case of large ϵ, Theorem 5 can be extended to handle local list-decoding from *erasures*, and gives a lower bound of $q = \Omega(\frac{\log(1/\delta)}{\epsilon})$ on the number of queries of local list-decoders that decode from a $1 - \epsilon$ fraction of erasures. We do not see how to extend the proof of Theorem 6 to erasures.

The model of black-box proofs that we introduce in Section 3 is quite general, and to the best of our knowledge, covers all known proofs in the literature on hard-core predicates for general one-way functions. Is it possible to circumvent the black-box limitations and answer open problems 8 and 12 for *specific candidates* for one-way functions?

More generally, is it possible to come up with non-black-box techniques that circumvent the limitations?

References

1 Miklós Ajtai. Σ_1^1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.

2 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *Comput. Complex.*, 25(2):349–418, 2016. doi:10.1007/s00037-016-0128-9.

3 Sergei Artemenko and Ronen Shaltiel. Lower bounds on the query complexity of non-uniform and adaptive reductions showing hardness amplification. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, volume 6845 of *Lecture Notes in Computer Science*, pages 377–388. Springer, 2011. doi:10.1007/978-3-642-22935-0_32.

4 Gil Cohen, Anat Ganor, and Ran Raz. Two sides of the coin problem. In Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014, Barcelona, Spain*, volume 28 of *LIPIcs*, pages 618–629. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.618.

5 Yevgeniy Dodis, Abhishek Jain, Tal Moran, and Daniel Wichs. Counterexamples to hardness amplification beyond negligible. In Ronald Cramer, editor, *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 476–493. Springer, 2012. doi:10.1007/978-3-642-28914-9_27.

6 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989. doi:10.1145/73007.73010.

7 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 956–966. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00094.

8 Venkatesan Guruswami. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science*, 2(2), 2006. doi:10.1561/0400000007.

9 Dan Gutfreund and Guy N. Rothblum. The complexity of local list decoding. In Ashish Goel, Klaus Jansen, José D. P. Rolim, and Ronitt Rubinfeld, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, volume 5171 of *Lecture Notes in Computer Science*, pages 455–468. Springer, 2008. doi:10.1007/978-3-540-85363-3_36.

10 Nutan Limaye, Karteek Sreenivasaiah, Srikanth Srinivasan, Utkarsh Tripathi, and S. Venkitesh. A fixed-depth size-hierarchy theorem for $\mathrm{AC}^0[\oplus]$ via the coin problem. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 442–453. ACM, 2019. doi:10.1145/3313276.3316339.

11 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin Varma. Erasures versus errors in local decoding and property testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:195, 2018. doi:10.4230/LIPIcs.ITCS.2019.63.

12 Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Notions of reducibility between cryptographic primitives. In Moni Naor, editor, *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004. doi:10.1007/978-3-540-24638-1_1.

13 Noga Ron-Zewi, Ronen Shaltiel, and Nithin Varma. Query complexity lower bounds for local list-decoding and hard-core predicates (even for small rate and huge lists). *Electron. Colloquium Comput. Complex.*, 27:133, 2020. URL: https://eccc.weizmann.ac.il/report/2020/133.

14 Ronen Shaltiel. Is it possible to improve Yao's XOR lemma using reductions that exploit the efficiency of their oracle? In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 10:1–10:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.10.

15 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010. doi:10.1137/080735096.

16 Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Trans. Inf. Theory*, 50(12):3015–3025, 2004. doi:10.1109/TIT.2004.838377.

17 Emanuele Viola. The complexity of hardness amplification and derandomization, 2006.

18 Sergey Yekhanin. Locally decodable codes. *Found. Trends Theor. Comput. Sci.*, 6(3):139–255, 2012. doi:10.1561/0400000030.