Cryptography from Information Loss

Marshall Ball
Columbia University, New York, NY, USA
marshall@cs.columbia.edu

Elette Boyle
IDC Herzliya, Israel
eboyle@alum.mit.edu

Akshay Degwekar
MIT, Cambridge, MA, USA
degwekarakshay@gmail.com

Apoorvaa Deshpande
Brown University, Providence, RI, USA
apoorvaa_deshpande@brown.edu

Alon Rosen
IDC Herzliya, Israel
alon.rosen@idc.ac.il

Vinod Vaikuntanathan
MIT, Cambridge, MA, USA
vinodv@mit.edu

Prashant Nalini Vasudevan
UC Berkeley, CA, USA
prashvas@berkeley.edu

Abstract

Reductions between problems, the mainstay of theoretical computer science, efficiently map an instance of one problem to an instance of another in such a way that solving the latter allows solving the former.[1] The subject of this work is "lossy" reductions, where the reduction loses some information about the input instance. We show that such reductions, when they exist, have interesting and powerful consequences for lifting hardness into "useful" hardness, namely cryptography.

Our first, conceptual, contribution is a definition of lossy reductions in the language of mutual information. Roughly speaking, our definition says that a reduction C is t-lossy if, for any distribution X over its inputs, the mutual information $I(X;\mathsf{C}(X)) \leq t$. Our treatment generalizes a variety of seemingly related but distinct notions such as worst-case to average-case reductions, randomized encodings (Ishai and Kushilevitz, FOCS 2000), homomorphic computations (Gentry, STOC 2009), and instance compression (Harnik and Naor, FOCS 2006).

We then proceed to show several consequences of lossy reductions:

1. We say that a language L has an f-reduction to a language L' for a Boolean function f if there is a (randomized) polynomial-time algorithm C that takes an m-tuple of strings $X = (x_1, \ldots, x_m)$, with each $x_i \in \{0,1\}^n$, and outputs a string z such that with high probability,

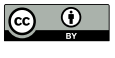
   $$L'(z) = f(L(x_1), L(x_2), \ldots, L(x_m))$$

   Suppose a language L has an f-reduction C to L' that is t-lossy. Our first result is that one-way functions exist if L is worst-case hard and one of the following conditions holds:
   - f is the OR function, $t \leq m/100$, and L' is the same as L
   - f is the Majority function, and $t \leq m/100$
   - f is the OR function, $t \leq O(m \log n)$, and the reduction has no error

   This improves on the implications that follow from combining (Drucker, FOCS 2012) with (Ostrovsky and Wigderson, ISTCS 1993) that result in *auxiliary-input* one-way functions.
2. Our second result is about the stronger notion of t-compressing f-reductions – reductions that only output t bits. We show that if there is an average-case hard language L that has a t-compressing Majority reduction to some language for $t = m/100$, then there exist collision-resistant hash functions.

   This improves on the result of (Harnik and Naor, STOC 2006), whose starting point is a cryptographic primitive (namely, one-way functions) rather than average-case hardness, and whose assumption is a compressing OR-reduction of SAT (which is now known to be false unless the polynomial hierarchy collapses).

[1] Such reductions are called many-one or Karp reductions. To be sure, there are more general types of reductions, such as oracle reductions (or Cook reductions), which we do not deal with in this paper.



11th Innovations in Theoretical Computer Science Conference (ITCS 2020).
Editor: Thomas Vidick; Article No. 81; pp. 81:1–81:27

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Along the way, we define a non-standard one-sided notion of average-case hardness, which is the notion of hardness used in the second result above, that may be of independent interest.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography; Theory of computation → Problems, reductions and completeness; Theory of computation → Cryptographic primitives

Keywords and phrases Compression, Information Loss, One-Way Functions, Reductions, Generic Constructions

Digital Object Identifier 10.4230/LIPIcs.ITCS.2020.81

Funding Marshall Ball is supported by an IBM Research PhD Fellowship. This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA) via Contract No. 2019-1902070006. Elette Boyle is supported in part by ISF grant 1861/16 and AFOSR Award FA9550-17-1-0069. Akshay Degwekar is supported in part by ISF grant 1861/16, AFOSR Award FA9550-17-1-0069, NSF Grants CNS-1413920 and CNS-1350619, and by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contracts W911NF-15-C-0226 and W911NF-15-C-0236. Alon Rosen is supported by ISF grant No. 1399/17 and via Project PROMETHEUS (Grant 780701). Vinod Vaikuntanathan is supported in part by NSF Grants CNS-1350619, CNS-1718161 and CNS-1414119, an MIT-IBM grant, a Microsoft Faculty Fellowship and a DARPA Young Faculty Award. Prashant Vasudevan is supported in part from AFOSR Award FA9550-19-1-0200, AFOSR YIP Award, NSF CNS Award 1936826, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). This work was done when Akshay Degwekar was a student at MIT, and in part while Marshall Ball, Akshay Degwekar, Apoorvaa Deshpande, and Prashant Vasudevan were visiting the FACT Center in IDC Herzliya. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either express or implied, of ODNI, IARPA, the U.S. Government, or other funding agencies. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

1 Introduction

Consider a polynomial-time reduction R from a language L to another L'.[2] That is, R takes as input $x \in \{0,1\}^n$ for any n and produces an output $y \in \{0,1\}^{m(n)}$ for some polynomial m such that $x \in L$ if and only if $y \in L'$. Such a reduction relates the computational complexities of L and L'; its existence says that the complexity of L is at most the complexity of L', upto some additional polynomial running time. In general, the reduction says little more than this. If the reduction has certain additional properties, however, it tells us more about the complexity of L and starts becoming more useful. This work is about an important such class of reductions, namely *lossy reductions* that forget information about the input instance.

Losing Information through Compression. One way to lose information about the input instance is by compressing it. Continuing the discussion above, if $m(n) = O(\log n)$, the reduction enables L to be decided in non-uniform polynomial-time. More generally, L can be decided on n-bit inputs by a circuit of size roughly $2^{m(n)}$. Such *instance compressing*

[2] Much of our discussion also applies to promise problems and search problems.

reductions, those that have $m(n) \ll n$, have been an important part of the study of fixed parameter tractable algorithms, where they are called kernelizations (see [11] and the references therein). A classical example is the vertex cover problem on n-node graphs parameterized by the size of the cover k, where there is a $\mathsf{poly}(n)$-time algorithm that takes the graph as input and outputs a smaller graph of size $\mathsf{poly}(k)$ which has a cover of size k if and only if the original graph does.

Harnik and Naor [18] showed a variety of cryptographic applications of such compressing reductions. For example, assuming that the Boolean Satisfiability problem (SAT) has good enough instance compression, they showed how to construct a collision-resistant hash function starting from any one-way function, a task known to be impossible to do in a black-box manner [28]. Furthermore, if this compression has some additional properties, they also show how to get public-key encryption schemes and oblivious transfer protocols from one-way functions. However, Fortnow and Santhanam [14] and later Drucker [12] showed that, unless the polynomial hierarchy collapses, such instance compression algorithms for SAT cannot exist.

This Work: Losing Information without Compression. Compression is one way to lose information, but not the only one. In this paper, we study cryptographic implications of *lossy reductions* – randomized algorithms that forget information about the input instance but not necessarily by compressing them. Such reductions are abundant in the study of average-case complexity and cryptography.

An example is a reduction $\mathsf{R}(x)$ that outputs a sample from a distribution that is (almost) completely determined by whether x is contained in L (and, as before, $\mathsf{R}(x) \in L'$ if and only if $x \in L$). Such a reduction loses all information about x other than its membership in L. Thus, it relates the complexity of deciding L in the *worst-case* to the complexity of deciding L' *on average* over a specific distribution[3]. In other words, a reduction that loses all information about its input except membership in L is a *worst-case to average-case reduction* from L to L'.

From a different vantage point, such reductions are equivalent to *randomized encodings* [19], a powerful and versatile notion in cryptography (see the survey by Applebaum [1] and the references therein). In particular, randomized encodings have been used to delegate computations [3], for secure computation [31, 7, 20], for parallel cryptography [2], for key-dependent message security and much more.

Given this relevance to cryptography of reductions that lose all information except membership, both as average-case reductions and as randomized encodings, we ask whether anything similar can be said of reductions that are almost this way. That is, *are reductions that lose* almost *all other information except membership useful for cryptography*?

Lossy Reductions and Cryptography. Our first contribution is a definition of such lossy reductions through the lens of mutual information. For a function $t : \mathbb{N} \to \mathbb{R}^+$, we say that a reduction C from a language L to a language L' is *t-lossy* if for any random variable X over bit-strings of length n (capturing an input distribution), the mutual information $I(X; \mathsf{C}(X))$ is at most $t(n)$. Note that this definition does not fix an input distribution, rather quantifies over all of them. In this sense, it is reminiscent of the prior-free information complexity definition of [10]. Roughly speaking, if X has little entropy, we don't care; if X has more than $t(n)$ bits of entropy, we want the reduction to reveal at most $t(n)$ of it.

[3] The distribution is given by picking any $x_N \notin L$ and any $x_Y \in L$, and taking the equal convex combination of the distributions sampled by $\mathsf{R}(x_N)$ and $\mathsf{R}(x_Y)$.

Whereas Harnik and Naor showed that instance compression can be used to construct other cryptographic primitives *from* one-way functions, we investigate the possibility of using compression or information loss to construct one-way functions from simpler hardness properties. Some results along these lines are already known, or are implicit in prior work: (1) Drucker [12] showed that the existence of certain kinds of compression for a language L implies that L has a statistical zero-knowledge proof.[4] Together with a result of Ostrovsky [25], this implies that if such a language is *average-case hard*, then one-way functions exist; (2) Replacing average-case with worst-case hardness in the above gives us *auxiliary input* one-way functions, a weaker object [26]; (3) on the other hand, if the reduction is a randomized encoding, then replacing average-case with worst-case hardness in the above gives us one-way functions [5].

We demonstrate a number of other similar sufficient conditions to elevate worst-case hardness to one-way functions. To give the reader a taste of what is to come, one of our results is a way to achieve the best of (1), (2) and (3) above, showing how to elevate worst-case hardness of L into one-way functions if L has a lossy reduction to L'.

▶ **Informal Theorem 1.1.** *Suppose there is a perfect reduction from L to L' that is $O(\log n)$-lossy on n-bit inputs. If L is worst-case hard, then One-Way Functions exist.*

The above follows as a corollary to Informal Theorem 1.4 described later. As noted earlier, randomized encodings may be seen roughly as a 1-lossy reduction from L to some problem L', and the worst-case hardness of a problem that has randomized encodings implies the existence of a one-way function. The above theorem says that it is in fact sufficient for the reduction to be $O(\log n)$-lossy (rather than 1-lossy) for this conclusion to hold, if it has the additional property that it does not make any errors – that is, a YES instance of L is never mapped to a NO instance of L', and also the other way round. This may also be interpreted as saying that the implication to OWFs for randomized encodings still holds if the privacy guarantee of the encodings is much weaker, as long as the correctness is perfect.

f-Reductions and OWFs. Our main theorems are about the implications of lossy reductions for the composition of a simple Boolean function with membership in a language. Let $f = \{f^n\}$ denote a family of (partial) Boolean functions, where for some polynomial m, the function f^n takes $m(n)$ bits as input. We denote by $f \circ L$ the composition of f with L – on $m(n)$ inputs $x_1, \ldots, x_{m(n)} \in \{0,1\}^n$, this function is computed as $f(L(x_1), \ldots, L(x_{m(n)}))$. We refer to a reduction from $f \circ L$ to some problem as an f-reduction of L.

Such f-reductions that are also compressing (in terms of bit length) have been the subject of considerable past work in the study of parametrized complexity [18, 9, 14, 13] (see also further references in [13]), especially for simple functions f like AND and OR. Most relevant to our work are the results of Drucker [13], who showed that if there is a sufficiently compressing reduction from $\mathrm{OR}_m \circ L$ to any problem L', then L is contained in SZK (where m is some polynomial and OR_m is the family of OR function on $m(n)$ inputs). As noted earlier, this implied membership in SZK lets us lift the average-case hardness of L to a one-way function, or its worst-case hardness to an auxiliary-input OWF.

Our starting point is the observation that Drucker's proofs still work if the reduction were just lossy and not necessarily compressing. We prove the following theorems that show three different sufficient conditions for lossy reductions to be useful in lifting worst-case

[4] Drucker's results were stated for the stronger notion of compression in terms of bit-length, but his proofs imply the same results for notion of lossy reductions as well.

hardness directly to one-way functions. Each of the three imply OWFs along different paths, as described briefly below. These paths and the pointers to the relevant parts of the paper are presented in Figure 1.

▶ **Informal Theorem 1.2.** *Suppose, for some polynomial m, and promise problem L, there is a reduction from $\mathrm{OR}_m \circ L$ to L that is $(m(n)/100)$-lossy on $(n \cdot m(n))$-bit inputs. If L is worst-case hard, then One-Way Functions exist.*

We prove this by first showing that Drucker's techniques actually imply a certain kind of worst-case to average-case reduction to L'. And if L is worst-case hard, then this reduction lets us conclude that L' is *one-sided average-case hard*, a notion explained later in this section. Thus, if L' is L itself, then it is both contained in SZK and is one-sided average-case hard. Finally, we show that the existence of any such problem implies the existence of a OWF.

The next theorem we prove is that a similarly lossy Majority-reduction for L by itself lets us lift the worst-case hardness of L to a OWF, without worrying about what it reduces to.

▶ **Informal Theorem 1.3.** *Suppose, for some polynomial m, and promise problem L, there is a reduction from $\mathrm{MAJ}_m \circ L$ to some problem that is $(m(n)/100)$-lossy on $(n \cdot m(n))$-bit inputs. If L is also worst-case hard, then One-Way Functions exist.*

In this case, we extend Drucker's techniques to show that such a reduction from $\mathrm{MAJ}_m \circ L$ implies a *two-sided* average-case reduction from L to some problem L' – where the YES and NO parts of the distribution over instances of L' that L is reduced to can be sampled separately. In different terms, this implies that L has statistical randomized encodings, and the conclusion follows from the fact that the worst-case hardness of a problem that has randomized encodings implies one-way functions [5].

Finally, the following theorem says that we can make do with OR_m-reductions that are not as lossy as before – only $O(m(n)\log n)$-lossy instead of $m(n)/100$ – if the reduction is perfect, meaning that it never maps YES instances of $\mathrm{OR}_m \circ L$ to NO instances of L', and also the other way. Note that a reduction that is $\Omega(m(n)\log n)$-lossy could still preserve some information about all of its $m(n)$ inputs – for instance, whether each x_i is a YES or NO instance of L, etc..

▶ **Informal Theorem 1.4.** *Suppose, for some polynomial m, and promise problems L and L', there is a* perfect *reduction from $\mathrm{OR}_m \circ L$ to L' that is $O(m(n)\log n)$-lossy on $(n \cdot m(n))$-bit inputs. If L is worst-case hard, then One-Way Functions exist.*

We prove this by showing that a perfect reduction from $\mathrm{OR}_m \circ L$ implies a one-sided average-case reduction that is perfect in the sense that the NO and YES distributions that are generated by the reduction are disjoint. On the other hand, the hardness that is implied by the reduction is quite weak due to it not being lossy enough – it only says that the NO distribution is no better than $(1 - 1/\mathsf{poly}(n))$-distinguishable from the YES distribution for a given algorithm. However, the perfectness lets us show that the sampler for the NO distribution itself is a weak one-way function (with the input as its random string).

Collision-Resistance from Compression. Finally, we show that compressing reductions – a strong form of lossy reductions where the output length is smaller than the input length – can be used to lift one-sided average-case hardness to collision-resistant hash functions.

▶ **Informal Theorem 1.5.** *Suppose, for some polynomial m, and promise problem L, there is a perfect reduction from $\mathrm{OR}_m \circ L$ to some problem that compresses $(n \cdot m(n))$-bit inputs to $m(n)/100$ bits. If L is also one-sided average-case hard with perfect sampling*[5]*, then Collision-Resistant Hash Functions exist.*

Our construction builds directly on the construction of collision-resistant hash functions from homomorphic commitments by Ishai et al [21]. In our construction, the keys of the hash function for hashing strings of length m correspond to a set of $2m$ instances (grouped into m pairs) sampled from the NO distribution. Each bit of an input x is used to select an instance from the corresponding pair, and the hash function is computed by running the compressing OR-reduction on this set.

The construction in [21] may be seen to use essentially the same approach, where instead of using a compressing OR-reduction for a hard problem, they use the homomorphism of a commitment scheme. The security of the commitment scheme there is the analogue of the one-sided average-case hardness of L, and our observation is that, while homomorphism is one way to compress (XOR-compression in their case), it is not necessary, and any compressing OR-reduction is sufficient to use with the construction.

One-Sided Average-Case Hardness. In the process of studying the implications of OR-compression of a problem to itself, we introduce the notion of one-sided average-case hardness, which we describe next. Recall that the worst-case hardness of a problem L says that for any polynomial-time algorithm A that attempts to decide L, there exists some input x on which it is wrong. Perhaps the simplest notion of average-case hardness flips the quantifiers here and says that there exists a (samplable) distribution over inputs such that any algorithm fails to decide L with large advantage when inputs are sampled from this distribution.

A stronger notion of average-case hardness that we call *two-sided* average-case hardness (and one that directly relates to one-way functions), says that there are two (samplable) distributions, one over the just the YES instances of L and another over just the NO instances, such that no algorithm A can distinguish between them. One-sided average-case hardness is a notion between this and worst-case hardness, and says that there exists one samplable distribution N over (mostly) NO instances such that for any algorithm A, there exists a distribution Y_A over (mostly) YES instances such that A cannot distinguish between N and Y_A.

One-sided average-case hardness implies worst-case hardness, is implied by two-sided average-case hardness, and does not seem to be related in this manner to plain average-case hardness. Ostrovsky [25] proved that plain average-case hardness of a problem in SZK implies OWFs. We prove the following theorem that, to our knowledge, is incomparable to it.

▶ **Informal Theorem 1.6.** *If there is a problem in SZK that is one-sided average-case hard, then One-Way Functions exist.*

The theorem is proven using a reduction to (the complement of) the Statistical Difference problem that is complete for SZK [27]. We also show that the existence of an analogous notion of worst-case to average-case reduction – called one-sided average-case reduction – from a problem L to any other problem implies that L is contained in SZK. Thus, along with the worst-case hardness of L, they imply auxiliary-input OWFs analogous to how two-sided average-case hardness implies OWFs. Such reductions are closely related to the notion of semi-private randomized encodings [4].

[5] The perfect sampling condition implies that the YES and NO distributions that come up in the definition of the hardness are contained completely within the YES and NO parts of the problem.

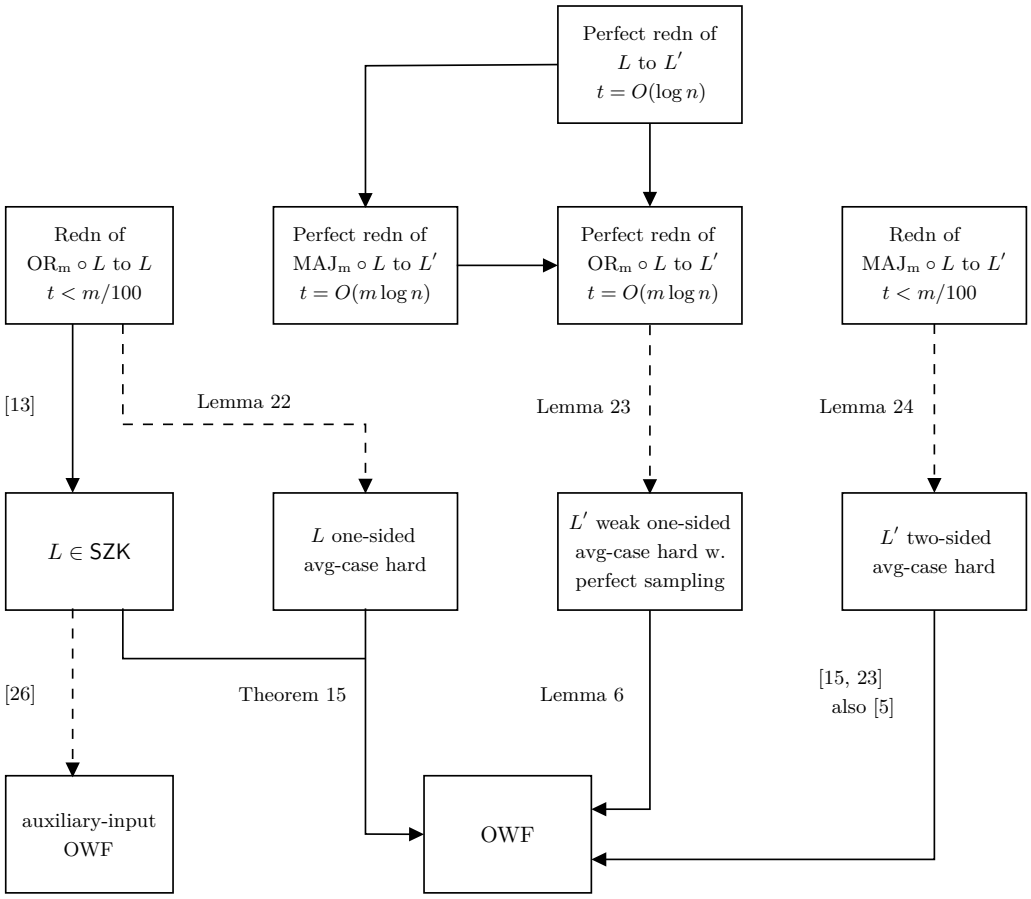


Figure 1 The many paths to One-Way Functions. L and L' are promise problems. The solid arrows denote implications, and the dashed arrows denote implication conditioned on the worst-case hardness of L. In all cases, m and t are polynomials in n, and the reductions lose all but $t(n)$ bits of information on a set of $m(n)$ inputs of size n – that is, t-lossy according to Definition 21 (when reducing L to L', $m(n)$ is set to 1).

1.1 Notation

We will always deal with promise problems and partial functions rather than languages and total functions, as these come up naturally in our discussions, and are also more general. Given a promise problem L, we denote by L_Y and L_N the corresponding sets of YES and NO inputs. Abusing notation, we also denote by L the partial function indicating membership in this promise problem. In the other direction, note that any partial Boolean function $f : \{0,1,\perp\}^n \to \{0,1,\perp\}$ has a corresponding promise problem, which we also sometimes denote by f. The symbol $\perp$ is set to be the output of a partial function on an input on which it is otherwise undefined. In general, we take U_S to be the random variable distributed uniformly over the set S, where S is clear from context we simply write U.

Random variables are capitalized (X), and algorithms are sans serif (A). $\mathsf{A}(X)$ denotes the random variable resulting from sampling x according to X and running A on it. $\mathrm{H}(X)$ is the Shannon entropy of X, and $\mathrm{I}(X;Y)$ is the mutual information between X and Y. We take $\Delta(X;Y)$ to denote the total variation distance between X and Y.

Unless otherwise specified, all algorithms in our work are non-uniform, and assumptions and claims of hardness are also against non-uniform algorithms. Further, we always use notions of hardness that are strong in the following sense – when we say, for instance, that a problem L is hard against polynomial-time algorithms, we mean that for any polynomial-time algorithm A, there is an $n_{\mathsf{A}} \in \mathbb{Z}$ such that for any $n \geq n_{\mathsf{A}}$, the algorithm A fails to decide L correctly on some instance of size n. This is to be contrasted against the more standard notion of hardness, infinitely-often hardness, which would say that there is an infinite sequence of n's such that A fails to decide L on some instance of size n. All of our lemmas and theorems, however, may also be stated in terms of infinitely-often hardness and follow from the same proofs, resulting in infinitely-often cryptographic objects as well.

1.2 Outline of the Paper

In Section 2, we define the various kinds of computational hardness that we will be using, the corresponding reductions, and some lemmas relating the reductions, the hardness, and one-way functions. In Section 3, we prove define the class SZK and prove that one-sided average-case hardness in SZK implies one-way functions, and some associated lemmas. In Section 4, we define our notion of lossy reductions, and prove that certain kinds of lossy reductions along with worst-case hardness imply one-way functions. In Section 5, we prove that one-sided average-case hardness and OR-compressing reductions imply collision-resistance hash functions.

2 Computational Hardness

In this section, we define the various kinds of computational hardness of promise problems that we will be employing in later discussions, and also different kinds of reductions between problems that have implications for such hardness. The proofs of the lemmas in this section are in Appendix A. We start by defining the weakest and simplest form of hardness.

▶ **Definition 1** (Worst-Case Hardness). *A problem L is* worst-case hard *if, for any polynomial-time algorithm A and all large enough n, there is an input $x \in (L_Y \cup L_N) \cap \{0,1\}^n$ such that $\Pr[\mathsf{A}(x) = L(x)] < 2/3$.*

Worst-case hardness says that for any algorithm A (and large enough instance size n), there is an input x on which it is wrong. We next define perhaps the simplest form of average-case hardness, which mostly just swaps these quantifiers. It says that there is a distribution over inputs x such that any algorithm A is wrong on average over this distribution.

▶ **Definition 2** (Average-Case Hardness). *A problem L is* average-case hard *if there is a polynomial-time sampling algorithm D such that:*

- *D mostly samples instances from L_Y and L_N. That is, for all large enough n,*

$$\Pr_{x \leftarrow \mathsf{D}(1^n)}[x \notin L_Y \cup L_N] \leq 0.1$$

- *The membership in L of most instances drawn from D is hard to decide. That is, for any polynomial-time algorithm A, for all large enough n,*

$$\Pr_{x \leftarrow \mathsf{D}(1^n)}[\mathsf{A}(x) = L(x) \vee L(x) = \perp] \leq \frac{1}{2} + 0.15$$

Note that the constant 2/3 in Definition 1 is somewhat arbitrary – if there is an algorithm A that does in fact decide L in the worst-case with success probability bounded away from 1/2 for all n, this can be amplified by repetition to get an algorithm A' with success probability close to 1.

In Definition 2, however, the constants are not arbitrary, at least in this straighforward sense. There are known hardness amplification theorems (see [6, Chapter 19]) that can take a problem L that is somewhat average-case hard as above, and obtain another problem L' and a distribution under which L' is much harder, but these do not say that L itself is any harder than initially supposed. Our choice of the constants we use in our definitions are such that the hardness is mild enough to be implied by the hypotheses that we later start with, and are yet strong enough to have interesting implications for cryptography, as shown by our theorems.

We next define a stronger and natural notion of average-case hardness that is known to be closely related to One-Way Functions. It separates the hard distribution into YES and NO parts, and asks that each of these parts be efficiently samplable.

▶ **Definition 3** (Two-Sided Average-Case Hardness). *A problem L is* two-sided average-case hard *if there are two polynomial-time sampling algorithms Y and N such that:*

- *Y and N mostly sample instances from L_Y and L_N, respectively. That is, for all large enough n,*

$$\Pr_{x \leftarrow \mathsf{Y}(1^n)}[x \notin L_Y] \leq 0.1$$
$$\Pr_{x \leftarrow \mathsf{N}(1^n)}[x \notin L_N] \leq 0.1$$

- *The outputs of Y and N are computationally indistinguishable. That is, for any polynomial-time algorithm A, for all large enough n,*

$$\left| \Pr_{x \leftarrow \mathsf{Y}(1^n)}[\mathsf{A}(x) = 1] - \Pr_{x \leftarrow \mathsf{N}(1^n)}[\mathsf{A}(x) = 1] \right| \leq 0.3$$

It is known that any two-sided average-case hard problem implies the existence of a One-Way Function. This follows from the fact that such hardness implies the existence of a family of statistically far distributions that are computationally indistinguishable (N and Y). Goldreich [15] showed that this is equivalent to the existence of OWFs if the indistinguishability was with negligible advantage, and this was later extended by Naor and Rothblum [23] to hold for weaker indistinguishability that covers the constants used above (see [8] for an alternative proof), leading to the following lemma. A theorem that is roughly equivalent to the combination of Lemmas 4 and 9 was also proven by Applebaum and Raykov [5].

▶ **Lemma 4.** *There is a problem that is two-sided average-case hard if and only if One-Way Functions exist.*

Further, note that the existence of a OWF also implies (as can be seen by the PRG that can be constructed from it) that there is some language that is two-sided average-case hard, but with much stronger guarantees – with negligible functions (in n) in place of the constants 0.1 and 0.3 in Definition 3. In this sense, for the purposes of its relevance to cryptography, the constants in the definition above are also somewhat arbitrary.

Finally, we introduce a notion of average-case hardness that is intermediate between worst-case and two-sided average-case hardness and, to our knowledge, is incomparable to plain average-case hardness. Recall that average-case hardness was obtained by swapping

the quantifiers in worst-case hardness, and two-sided average-case hardness then came out of separating and fixing the YES and NO parts. The following definition fixes just the NO distribution, and requires that for any algorithm A, there exists some YES distribution that it cannot distinguish from this fixed NO distribution.

▶ **Definition 5** (One-Sided Average-Case Hardness). *A problem L is* one-sided average-case hard *if there is a polynomial-time sampling algorithm N such that:*

- N *mostly samples instances from L_N. That is, for all large enough n,*

$$\Pr_{x \leftarrow \mathsf{N}(1^n)} [x \notin L_N] \leq 0.1$$

- *For any polynomial-time algorithm, there is some distribution that is mostly over L_Y that it cannot distinguish from the output of N. That is, for every polynomial-time algorithm A, there is a (possibly inefficient) sampler Y_A such that:*
  - Y_A *mostly samples instances from L_Y. That is, for all large enough n,*

$$\Pr_{x \leftarrow \mathsf{Y}_\mathsf{A}(1^n)} [x \notin L_Y] \leq 0.1$$

  - *The outputs of N and Y_A are indistinguishable to A. That is, for all large enough n,*

$$\left| \Pr_{x \leftarrow \mathsf{N}(1^n)} [\mathsf{A}(x) = 1] - \Pr_{x \leftarrow \mathsf{Y}_\mathsf{A}(1^n)} [\mathsf{A}(x) = 1] \right| \leq 0.3$$

We say that the hardness is strong *if the distinguishing advantage is negligible in n, and* weak *if it is only required to be less than $(1 - 1/n^c)$ for some constant c. We say that L is one-sided average-case hard with* perfect sampling *if N and Y_A only output samples in L_N and L_Y, respectively.*

This kind of hardness turns out to be somewhat related to a weaker kind of OWF called auxiliary-input OWF – see Corollary 17. We do not know how to amplify this kind of hardness, so the choice of constants in its definition is not arbitrary.

We consider separately the weak one-sided average-case hardness with perfect sampling, which requires that the YES and NO distribution be contained completely within the respective parts of the problem, but places a much weaker requirement on their indistinguishability. This variant implies the existence of OWFs, though following an approach different from that taken in Lemma 4.

▶ **Lemma 6.** *If there is a problem that is weak one-sided average-case hard with perfect sampling, then One-Way Functions exist.*

2.1 Reductions

Reductions relate the complexities of different problems, and while there are more involved forms of reductions that still do so, in our work we will be using the simple notion of Karp reductions. Roughly, a Karp reduction from L to L' takes an instance x and produces an instance x' whose membership in L' is completely determined by that of x in L. Later definitions below also ask for some additional properties.

▶ **Definition 7** (Karp Reduction). *A polynomial-time algorithm R is a* Karp reduction *from a problem L to a problem L' if for all large enough n:*

$$x \in L_Y \cap \{0,1\}^n \implies \Pr[\mathsf{R}(x) \in L'_Y] \geq 0.9$$
$$x \in L_N \cap \{0,1\}^n \implies \Pr[\mathsf{R}(x) \in L'_N] \geq 0.9$$

In this case, L is said to reduce *to L'. If the above probabilities are both 1, then A is said to be a* perfect *Karp reduction.*

A worst-case reduction like the one above from L to L' allows us to conclude that L' is worst-case hard if L is worst-case hard. Throughout the rest of this work, we will be drawing conclusions about the two-sided and one-sided average-case hardness of L' that follow from analogous notions of worst-to-average-case reductions defined below.

▶ **Definition 8** (Two-Sided Average-Case Karp Reduction). *A polynomial-time algorithm* R *is a* two-sided average-case Karp reduction *from a problem* L *to a problem* L' *if there are two polynomial-time samplers* Y *and* N *such that the following hold for all large enough* n*:*

- Y *and* N *mostly sample instances from* L'_Y *and* L'_N*, respectively. That is,*

$$\Pr_{x \leftarrow \mathsf{Y}(1^n)}[x \notin L'_Y] \leq 0.1$$

$$\Pr_{x \leftarrow \mathsf{N}(1^n)}[x \notin L'_N] \leq 0.1$$

- *For any* $x \in L_N \cap \{0,1\}^n$*, the output of* $\mathsf{R}(x)$ *is close to that of* $\mathsf{N}(1^n)$*. That is, for such* x*,*

$$\Delta\left(\mathsf{R}(x); \mathsf{N}(1^n)\right) \leq 0.1$$

- *For any* $x \in L_Y \cap \{0,1\}^n$*, the output of* $\mathsf{R}(x)$ *is close to that of* $\mathsf{Y}(1^n)$*. That is, for such* x*,*

$$\Delta\left(\mathsf{R}(x); \mathsf{Y}(1^n)\right) \leq 0.1$$

A two-sided average-case reduction from L to L' allows us to conclude that L' is two-sided average-case hard if L is worst-case hard.

▶ **Lemma 9.** *Suppose there is a two-sided average-case Karp reduction from a language* L *to a language* L'*, and* L *is worst-case hard. Then,* L' *is two-sided average-case hard.*

Finally, we define one-sided average-case reductions, which similarly relate one-sided average-case hardness to worst-case hardness.

▶ **Definition 10** (One-Sided Average-Case Karp Reduction). *A polynomial-time algorithm* R *is a* one-sided average-case Karp reduction *from a problem* L *to a problem* L' *if there is a polynomial-time sampler* N *such that the following hold for all large enough* n*:*

- N *mostly samples instances from* L'_N*. That is,*

$$\Pr_{x \leftarrow \mathsf{N}(1^n)}[x \notin L'_N] \leq 0.1$$

- *For any* $x \in L_N \cap \{0,1\}^n$*, the output of* $\mathsf{R}(x)$ *is close to that of* $\mathsf{N}(1^n)$*. That is, for such* x*,*

$$\Delta\left(\mathsf{R}(x); \mathsf{N}(1^n)\right) \leq 0.1$$

- *For any* $x \in L_Y \cap \{0,1\}^n$*, the output of* $\mathsf{R}(x)$ *is mostly contained in* L'_Y*. That is, for such* x*,*

$$\Pr_{x' \leftarrow \mathsf{R}(x)}[x' \notin L'_Y] \leq 0.1$$

The reduction is said to be strong *if for any* $x \in L_N \cap \{0,1\}^n$*, the distance* $\Delta\left(\mathsf{R}(x); \mathsf{N}(1^n)\right)$ *is negligible in* n*, and is* weak *if this is at most* $1 - 1/n^c$ *for some constant* c*. The reduction is said to have* perfect sampling *if the probability that the outputs of* N *or* $\mathsf{R}(x)$ *when* $x \in L_Y$ *are not in* L'_N *and* L'_Y*, respectively, are* 0*.*

▶ **Lemma 11.** *Suppose there is a one-sided average-case Karp reduction from a language L to a language L', and L is worst-case hard. Then, L' is one-sided average-case hard. Further, if the reduction is weak and has perfect sampling, then the hardness is also weak and has perfect sampling.*

3 Statistical Zero Knowledge

One of the paths we take to showing the existence of OWFs is through a one-sided average-case hard problem that also has a statistical zero-knowledge proof. The class SZK of problems that have statistical zero-knowledge proofs has been widely studied in the past, partly owing to its connections to cryptography (see [29] and discussions and references therein). Due to a completeness theorem of Sahai and Vadhan [27], we may equivalently define this class in the following manner that is more convenient for us.

▶ **Definition 12** (Statistical Zero Knowledge [27])**.** Statistical Zero Knowledge *(*SZK*) is the class of promise problems which have a perfect Karp reduction to the* Statistical Difference problem *(*SD*), which is defined over pairs of circuits (C_0, C_1), as follows:*

$$\mathrm{SD}_Y = \{(C_0, C_1) \mid \Delta(C_0(U)); C_1(U))) > 2/3\}$$
$$\mathrm{SD}_N = \{(C_0, C_1) \mid \Delta(C_0(U)); C_1(U))) < 1/3\}$$

where n is the output length of both C_0 and C_1, and the U's above represent uniform distributions over the appropriate input domains.

We will also use the following two results regarding SZK.

▶ **Lemma 13** (Polarization [27])**.** *There exists an efficient procedure* Polarize *that when given two circuits C_0, C_1 and a parameter 1^λ as input outputs C_0', C_1' such that*

- *If $\Delta(C_0(U); C_1(U)) > 2/3$, then $\Delta(C_0'(U); C_1'(U)) > 1 - 2^{-\lambda}$.*
- *If $\Delta(C_0(U); C_1(U)) < 1/3$, then $\Delta(C_0(U); C_1(U)) < 2^{-\lambda}$.*

▶ **Theorem 14** (SZK closed under complement [24])**.** *If $\Pi = (\Pi_Y, \Pi_N) \in \mathsf{SZK}$, then $\overline{\Pi} = (\overline{\Pi}_Y, \overline{\Pi}_N) \in \mathsf{SZK}$ where $\overline{\Pi}_Y = \Pi_N$ and $\overline{\Pi}_N = \Pi_Y$.*

As noted in Lemma 4, two-sided average-case hardness of any problem implies the existence of OWFs. Ostrovsky [25] showed that *plain* average-case hardness of any problem in SZK also implies the existence of OWFs. Furthermore, Ostrovsky and Wigderson [26] observe that the *worst-case* hardness of any problem in SZK implies the existence of *auxiliary-input* one-way functions.[6] We show the following incomparable theorem.

▶ **Theorem 15.** *If there is a problem in* SZK *that is one-sided average-case hard, then One-Way Functions exist.*

[6] Roughly, auxiliary-input one-way functions (against non-uniform adversaries) exist if there is polynomial p such that for every family of poly size circuits, $\{\mathsf{A}_n\}_{n\in\mathbb{N}}$ there is family of circuits of size $p(n)$, $\{\mathsf{F}_n\}_{n\in\mathbb{N}}$, such that A_n fails to invert F_n when given a description of F_n as an auxiliary input. Contrast this with the notion of non-uniform one-way functions we use throughout: a (fixed) family of poly-sized circuits, $\{\mathsf{F}_n\}_{n\in\mathbb{N}}$ is a non-uniform one-way function if for *any* poly-sized circuit family $\{\mathsf{F}_n\}$ is hard to invert. So far as we know, the latter is strictly stronger. Interestingly however, if one is concerned with security against *uniform* adversaries, the quantifiers can be switched via diagonalization arguments and the two notions are equivalent.

Proof. We show that such hardness implies the existence of two samplable distributions that are α-statistically far but β-computationally indistinguishable, where there is a noticeable gap between α and β. (Here, we will take $\alpha = 4/5 - \text{negl}(n)$ and $\beta = 3/4$.) The rest follows from the results of Goldreich and Naor-Rothblum [15, 23].

The completeness of the complement of Statistical Difference(by Theorem 14) implies that for any problem L in SZK, there is a reduction R that takes input x and outputs two circuits that sample distributions that are far if $x \in L_N$, and negligibly close if $x \in L_Y$. Let L be the one-sided average-case hard problem in SZK and let N be the sampler for its fixed NO distribution. The two distributions we want are sampled by the samplers D_0 and D_1 below given security parameter n:

- $\mathsf{D}_0(1^n)$: Sample $x \leftarrow \mathsf{N}(1^n)$. Compute $(C_0, C_1) \leftarrow \mathsf{R}(x)$. Compute circuits $(C'_0, C'_1) \leftarrow \text{Polarize}(1^n, C_0, C_1)$. Pick random r of appropriate length and output $((C'_0, C'_1), C_0(r))$.
- $\mathsf{D}_1(1^n)$: Same as above, but output $((C'_0, C'_1), C'_1(r))$ at the end.

By definition, the event that $\mathsf{N}(1^n)$ outputs something not in L_N happens with probability at most $1/10$. By the definition of $\overline{\text{SD}}$, conditioned on this event not happening, for any fixed output C_0, C_1 we have that $\Delta(C_0(U); C_1(U)) > 2/3$. If this is the case, it follows from Lemma 13 that $\Delta(C'_0(U); C'_1(U)) > 1 - 2^{-n}$ It follows that, *conditioned on this event not happening*, we have $\Delta(\mathsf{D}_0(1^n); \mathsf{D}_1(1^n)) > 1 - \text{negl}(n)$. By standard manipulations[7], we get that $\Delta(\mathsf{D}_0(1^n); \mathsf{D}_1(1^n)) > .8 - \text{negl}(n)$.

To see that these distributions are 3/4-computationally indistinguishable, consider any distinguisher A for them. Suppose A has more than 3/4 advantage in distinguishing $\mathsf{D}_0(1^n)$ from $\mathsf{D}_1(1^n)$. But then consider A' that attempts to solve L by on input x running the reduction,R, and polarization procedure, $\text{Polarize}(1^n, \cdot, \cdot)$ to get C'_0, C'_1, flipping a coin $b \leftarrow U_{\{0,1\}}$, and outputting 1 if $\mathsf{A}(C'_0, C'_1, C'_b(U)) = b$. If A' is given inputs from $N(1^n)$, then what it feeds A is identically distributed to either $D_0(1^n)$ (if $b = 0$) or $D_1(1^n)$ (if $b = 1$). It follows from A's advantage that $Pr[\mathsf{A}'(N(1^n))] > \frac{1+3/4}{2} = .875$.

On the other hand, by virtue of the fact that L is one-sided average case hard there is a distribution $\mathsf{Y}_{\mathsf{A}'}$ which is .3-indistinguishable from N to A'. Consider $\mathsf{A}'(\mathsf{Y}_{\mathsf{A}'})$. First of all $\Pr[\mathsf{Y}_{\mathsf{A}'} \in L_Y] \geq .9$. Moreover, for any such $x \in L_Y$, $\mathsf{R}(x) \in \overline{\text{SD}}_N$. In this case, $\text{Polarize}(1^n)$ will output (C'_0, C'_1) such that $\Delta(C'_0(U); C'_1(U)) < 2^{-n}$. Therefore, with probability at least .9, the two circuits A' gives to A correspond to distributions with negligible distance from one another. It follows from standard manipulations that A, regardless of efficiency, can distinguish with probability at most $.1 + \text{negl}(n)$. Thus, $\Pr[\mathsf{A}'(\mathsf{Y}_{\mathsf{A}'}) = 1] \leq \frac{1+.15}{2} = .575$.

But then $\Pr[\mathsf{A}'(\mathsf{N}_{\mathsf{A}'}) = 1] - \Pr[\mathsf{A}'(\mathsf{Y}_{\mathsf{A}'}) = 1] > .3$, which violates the assumption on $\mathsf{Y}_{\mathsf{A}'}$ following from the average-case hardness of L. ◀

The above theorem talks about the implication of a problem that is both one-sided hard and is in SZK. The following lemma, which was implicitly used by Drucker [13], says that membership of a problem in SZK is implied by the existence of any one-sided average-case reduction *from* it. A stronger version of this lemma (in different terminology) was also proven by Applebaum and Raykov [5].

▶ **Lemma 16.** *If there is a one-sided average-case Karp reduction from a problem L to any other problem, then L is in* SZK.

[7] For any random variables X, Y, Z and any event E, $\Delta(X;Y) \in [\Pr[Z \in E]\Delta(X|Z \in E; Y|Z \in E) \pm \Pr[Z \notin E]]$.

Proof of Lemma 16. We show this by reduction to the Statistical Difference problem, and appealing to its completeness of SZK and the closure of the class under complement. Suppose R is the one-sided average-case Karp reduction and N is the canonical NO distribution from Definition 10. The reduction is, given input x for L, to output the pair of circuits $(\mathsf{N}(1^n;\cdot), \mathsf{R}(x;\cdot))$ – each of these takes the randomness for the respective algorithm as input and produces the corresponding output. The properties of the one-sided reduction guarantee that if $x \in L_N$, then $\Delta(\mathsf{N}(1^n); \mathsf{R}(x))$ is at most 0.1, while if $x \in L_Y$, then this is at least 0.9. ◀

Because worst-case hard languages in SZK imply auxiliary-input one-way functions [26], the following is an immediate consequence of Lemma 16. We refer the reader to [26] for the definition of auxiliary-input one-way functions.

▶ **Corollary 17.** *If L is worst-case hard and there is a one-sided average-case Karp reduction from L to any other problem, then auxiliary-input one-way functions exist.*

Starting with a stronger hypothesis – a *two-sided* average-case reduction from L – we show membership in SRE, which is the class of problems that have statistical randomized encodings (and is a subset of SZK). We refer the reader to [5] for the definitions of randomized encodings and this class.

▶ **Lemma 18.** *If there is a two-sided average-case Karp reduction from a problem L to any other problem, then L is in* SRE*.*

This lemma is proven in the same way as Lemma 16, with the two fixed distributions of the randomized encodings taken to be Y and N from the two-sided average-case reduction (see Definition 8).

4 Lossy Reductions

In this section, we define our notions of lossy reductions, and show their implications for OWFs. We start with a definition of what it means for a generic algorithm to lose information about its input. All algorithms in this section are randomized and non-uniform unless specified otherwise.

▶ **Definition 19** (Lossy Algorithm)**.** *An algorithm* C *is said to* t-lossy on n-bit inputs *for some $n \in \mathbb{N}$ and $t \in \mathbb{R}^+$ if, for any random variable X over $\{0,1\}^n$,*

$$\mathrm{I}(X; \mathsf{C}(X)) \leq t$$

Note that being t-lossy means, in a sense, that the algorithm loses $(n - t(n))$ bits of information. For convenience, we overload the above terminology for the following special cases of reductions.

▶ **Definition 20** (Lossy Reduction)**.** *For a function $t : \mathbb{N} \to \mathbb{R}^+$, a Karp reduction from a problem L to a problem L' is said to be* t-lossy *if, for every $n \in \mathbb{N}$, it is $t(n)$-lossy on n-bit inputs.*

For a polynomial m, consider a family $f = \left\{f_n : \{0,1,\perp\}^{m(n)} \to \{0,1,\perp\}\right\}$ of partial functions. Define the problem $f \circ L$ on inputs from $\{0,1\}^{n \times m(n)}$ as the composition of f with $m(n)$ copies of L; that is, for $x_1, \ldots, x_{m(n)} \in \{0,1\}^n$, define $(f \circ L)(x_1, \ldots, x_{m(n)}) = f(L(x_1), \ldots, L(x_{m(n)}))$. We also overload the terminology for reductions from such compositions as follows.

▶ **Definition 21** (Lossy f-Reduction). *Let $m : \mathbb{N} \to \mathbb{N}$ and $t : \mathbb{N} \to \mathbb{R}^+$ be polynomials, and $f = \left\{f_n : \{0,1,\bot\}^{m(n)} \to \{0,1,\bot\}\right\}$ be a family of partial functions. For a problem L, a Karp reduction from $f \circ L$ to a problem L' is said to be* t-lossy *if, for every $n \in \mathbb{N}$, it is $t(n)$-lossy on $(n \cdot m(n))$-bit inputs. We say in this case that L has a t-lossy f-reduction to L'.*

In the last two definitions, the problem L is said to have a t-lossy *self-reduction* (or self-f-reduction) if L' is the same as L. During many of our discussions, the size parameter n will be fixed, and when it is, we will use just t and m to denote the numbers $t(n)$ and $m(n)$.

We first consider the family of OR functions – for some polynomial m, the family $\mathrm{OR}_m = \left\{\mathrm{OR}^n : \{0,1,\bot\}^{m(n)} \to \{0,1,\bot\}\right\}_{n\in\mathbb{N}}$, where OR^n is a function from $\{0,1,\bot\}^{m(n)}$ to $\{0,1,\bot\}$ that is the Boolean OR of its inputs if they are all from $\{0,1\}$, and is $\bot$ if any of its inputs is $\bot$. Drucker [13] showed that any problem that is OR_m-compressible to $O(m(n)\log n)$ bits is contained in SZK, where his notion of compression to t bits was that the output length of the reduction is at most t bits (similar to Definition 31). We observe that his proof works almost as is for reductions that are lossy in the sense of Definition 21 (but that may not be compressing). We isolate the following lemma that is implicit in [13], which will be useful for us, and state it in terms of average-case Karp reductions.

▶ **Lemma 22.** *Suppose, for some polynomial m and problems L, L', there is a reduction from $\mathrm{OR}_m \circ L$ to L'. If this reduction is $(m/100)$-lossy and $m(n) > 100$ for all large enough n, then there is a one-sided average-case Karp reduction from L to L'.*

As noted above, Drucker shows membership in SZK as long as the compression (or loss) is to $O(m \log n)$ bits, but the implication of the one-sided average-case reduction as in Lemma 22 seems to follow only if the loss is to somewhat less than m bits. In the case where the reduction from $\mathrm{OR}_m \circ L$ does not make any errors, however, we can recover a weak one-sided average-case reduction with perfect sampling even if the loss is only to $O(m \log n)$ bits.

▶ **Lemma 23.** *Suppose, for some polynomial m and problems L, L', there is a* perfect *reduction from the problem $\mathrm{OR}_m \circ L$ to L'. If this reduction is $O(m \log n)$-lossy, then there is a weak one-sided average-case Karp reduction with perfect sampling from L to L'.*

Define MAJ_m in the same way as OR_m, but with the Majority function. We extend Drucker's results and show the following stronger conclusion from lossy Majority-reduction – that it implies a two-sided average-case reduction (or a randomized encoding, following Lemma 18).

▶ **Lemma 24.** *Suppose, for some polynomial m and problems L, L', there is a reduction from the problem $\mathrm{MAJ}_m \circ L$ to L'. If this reduction is $m/100$-lossy and $m(n) > 100$ for all large enough n, then there is a two-sided average-case Karp reduction from L to L'.*

The proofs of these lemmas are in Appendix B.

4.1 Lossy Reductions and One-Way Functions

We now state and prove our main theorems, showing that certain lossy reductions for a problem L can be used to lift its worst-case hardness to a One-Way Function. Each of our three theorems is proven by following a different path, as explained in the respective proofs. The first says that good enough lossy self-OR-reduction implies a OWF.

▶ **Theorem 25.** *Suppose, for some polynomial m and a problem L, there is a reduction from $\mathrm{OR}_m \circ L$ to L that is $(m/100)$-lossy. If L is worst-case hard and $m(n) > 100$ for all large enough n, then One-Way Functions exist.*

Proof of Theorem 25. This theorem is proven by showing that the hypothesis implies that L is both one-sided average-case hard and contained in SZK which, as seen in Section 3, implies a OWF. The proof is as follows:

- Lemma 22 implies that there is a one-sided average-case reduction from L to itself.
- Lemma 16 and the one-sided reduction imply that $L \in \mathsf{SZK}$.
- Together with the worst-case hardness of L and the one-sided reduction, Lemma 11 implies that L is one-sided average-case hard.
- Theorem 15, along with the above two conclusions, implies that OWFs exist. ◀

Our second theorem uses an incomparable hypothesis – that there is a lossy MAJ-reduction, but not necessarily to L itself.

▶ **Theorem 26.** *Suppose, for some polynomial m and a problem L, there is a reduction from $\mathrm{MAJ}_m \circ L$ to some problem that is $(m/100)$-lossy. If L is worst-case hard and $m(n) > 100$ for all large enough n, then One-Way Functions exist.*

Proof of Theorem 26. This theorem is proven by showing that such a reduction implies the existence of a two-sided average-case reduction from L, and using this to go from worst-case hardness of L to OWFs. Suppose the reduction is to a problem L'. The proof is as follows:

- Lemma 24 implies that there is a two-sided average-case reduction from L to L'.
- Together with the worst-case hardness of L and the two-sided reduction, Lemma 9 implies that L' is two-sided average-case hard.
- Lemma 4, along with the above two-sided hardness, implies that OWFs exist. ◀

The third theorem also uses a hypothesis incomparable to the other two – it assumes that the lossy reduction is perfect, but works even if the loss is to more bits than the number of instances it takes as input.

▶ **Theorem 27.** *Suppose, for a polynomial m and a problem L, there is a* perfect *reduction from $\mathrm{OR}_m \circ L$ to some problem that is $O(m \log n)$-lossy. If L is also worst-case hard, then One-Way Functions exist.*

We draw the following interesting corollaries of this theorem, both of which follow from a direct implication of lossy perfect OR-reduction by their respective hypotheses. The first is a statement along the lines of Lemma 4, but incomparable to it. Lemma 4 may be rephrased as saying that a 1-lossy reduction from a worst-case hard language implies OWFs, and Corollary 28 relaxes the requirement to $O(\log n)$-lossiness, but requires the reduction to be perfect.

▶ **Corollary 28.** *Suppose there is a* perfect *reduction from a problem L to another problem that is $O(\log n)$-lossy. If L is also worst-case hard, then One-Way Functions exist.*

The second corollary is the analogue of Theorem 27 for Majority reductions.

▶ **Corollary 29.** *Suppose, for a polynomial m and a problem L, there is a* perfect *reduction from $\mathrm{MAJ}_m \circ L$ to some problem that is $O(m \log n)$-lossy. If L is also worst-case hard, then One-Way Functions exist.*

Proof of Theorem 27. This theorem is proven by showing that a perfect lossy OR-reduction leads to a weak one-sided average-case reduction with perfect sampling, which along with worst-case hardness leads to OWFs. Suppose the reduction is to a problem L'. The proof is as follows:

- Lemma 23 implies that there is a weak one-sided average-case reduction from L to L' with perfect sampling.
- Together with the worst-case hardness of L and the above reduction, Lemma 11 implies that L' is weak one-sided average-case hard with perfect sampling.
- Lemma 6, along with the above hardness, implies that OWFs exist. ◀

5 Collision Resistance

In this section, we show that that one-sided average-case hardness of a language L combined with compressiblility of $\mathrm{OR}_m \circ L$ (in the standard sense of compressed output length), implies the existence of *collision-resistant hash functions.*

The reduction builds directly on the construction of collision-resistant hash functions from homomorphic encryption due to Ishai, Kushilevitz, and Ostrovsky [21] (more generally, from any one-round private information retrieval (PIR) protocol, or homomorphic one-way commitments). Indeed, these cryptographic notions can be viewed precisely as providing compressing reductions for related languages whose average-case hardness is implied by security: e.g., additively homomorphic encryption corresponds directly to self-compression from $(\mathrm{XOR}_m \circ L)$ to L, for the (average-case hard) language L where L_Y is the set of ciphertexts encrypting the bit 1, and L_N is the set of ciphertexts of 0.

We begin by formally defining collision-resistant hash function families and the required form of output-length compression.

▶ **Definition 30.** Collision-resistant hash functions *exist if there exist* $\ell, \ell' : \mathbb{N} \to \mathbb{N}$ *with* $\ell(n) > \ell'(n)$*, an index set* $I \subseteq \{0,1\}^*$*, and probabilistic polynomial-time algorithms of the form:*

- $\mathit{Gen}(1^n)$*, which outputs an index* $s \in I$*,*
- $\mathit{Eval}(s, y)$*, which given index* $s \in I$ *and input* $y \in \{0,1\}^{\ell(n)}$*, outputs* $h_s(y) \in \{0,1\}^{\ell'(n)}$*,*

for which finding collisions is computationally hard. That is, for every polynomial-time (non-uniform) A*, there exists a negligible function* ν *for which*

$$\Pr_{\substack{s \leftarrow \mathit{Gen}(1^n) \\ (y,y') \leftarrow \mathsf{A}(s)}} [(y \neq y') \wedge (\mathit{Eval}(s,y) = \mathit{Eval}(s,y'))] \leq \nu(n).$$

▶ **Definition 31** (Compressing Reduction)**.** *An algorithm* C *is said to* compress n-bit inputs to t bits *for some* $n, t \in \mathbb{N}$ *if for any* $x \in \{0,1\}^n$ *the output bit length is* $|\mathsf{C}(x)| \leq t$*.*

For a function $t : \mathbb{N} \to \mathbb{N}$*, a Karp reduction from a problem* L *to a problem* L' *is said to* t-compressing *if, for every* $n \in \mathbb{N}$*, it is* $t(n)$*-compressing on* n*-bit inputs.*

We now present the main theorem of the section. Note for simplicity, we present the reduction with respect to a *perfect* compressing reduction of $\mathrm{OR}_m \circ L$ (see Definition 7), as well as assuming *perfect* sampling for the one-sided average-case hard language L (that is, for any polynomial-time A, the probability that the outputs of N or Y_A are not in L_N and L_Y, respectively, are set to 0).

▶ **Theorem 32.** *Suppose, for some polynomial* m *and a problem* L*, there is a perfect reduction from* $\mathrm{OR}_m \circ L$ *to* L' *that compresses to* $m/100$ *bits. If* L *is strongly one-sided average-case hard with perfect sampling and* $m(n) > 100$ *for all large enough* n*, then Collision-Resistant Hash Functions exist.*

Recall that existence of a compressing reduction from $(\mathrm{MAJ}_{2m+1} \circ L)$ to L', or from $(\mathrm{AND}_m \circ L)$ to L', each directly imply equivalent compressing reduction from $(\mathrm{OR}_m \circ L)$ to L', up to constant factors of compression. This yields the following corollary.

▶ **Corollary 33.** *The following variations of Theorem 32 additionally hold. Let m be polynomial and L a problem. Then collision-resistant hash functions exist assuming existence of a compressing reduction of the following corresponding kinds (to $m/100$ bits), in addition to the specified hardness requirements on L:*

1. *If there is a compressing reduction from $\mathrm{MAJ}_m \circ L$ to L' and L is strongly one-sided average-case hard with perfect sampling.*
2. *If there is a compressing reduction from $\mathrm{AND}_m \circ L$ to L' and the complement language $\mathsf{co}L$ is strongly one-sided average-case hard with perfect sampling.*

At a high level, the construction mirrors the approach of [21], as follows. Each hash function in the family will be keyed by a collection s of $2m$ randomly sampled no-instances of L (via N). The corresponding hash function $\mathsf{Eval}(s, \cdot)$ takes as input a bit-string $y \in \{0,1\}^m$, and outputs the $m/100$-bit instance of L' generated by applying the compressing reduction on the $(OR_m \circ L)$ instance defined by the m no-instances selected by the bits of y. A successful collision-finder can be used to gain contradictorily high advantage in deciding L (more specifically, in distinguishing a sample of N from the (possibly inefficient) Y_A), by embedding the challenge instance x into a random location $i^* \in [m]$ of the key, and seeing whether the colliding inputs $y \neq y'$ differ in this index i^*. If x was sampled from N, then the concocted key is properly distributed, independent of i^*, and thus y, y' must differ in position i^* with noticeable probability. On the other hand, if x was sampled from Y_A, then any $(\mathrm{OR}_m \circ L)$ instance containing x is a *yes* instance, whereas any instance not containing x is (by construction) a *no* instance; by perfect correctness of the compressing reduction, it thus cannot be the case that any two such inputs y, y' that differ in position i^* could collide to an identical output in L'.

Proof of Theorem 32. Given such a pair of languages L, L', we construct the desired collision-resistant hash function family.

Denote by R the weakly compressing reduction from $(\mathrm{OR}_m \circ L)$ to L'. By one-sided average-case hardness of L (as per Definition 5), there exists a polynomial-time sampling algorithm N which samples instances from L_N. Let $\ell(n) = m$ and $\ell'(n) = m/100$ be the input and (compressed) output length of the hash function. Consider the following algorithms.

- $\mathsf{Gen}(1^n)$: Independently sample $\ell(n) = m$ instances from N. That is, for $(i, b) \in [m] \times \{0,1\}$, let $x_{i,b} \leftarrow \mathsf{N}(1^n)$. Output $s = (x_{i,b})_{i \in [m], b \in \{0,1\}}$.
- $\mathsf{Eval}(s, y)$: Parse $s = (x_{i,b})_{i \in [m], b \in \{0,1\}}$ and $y = (y_1, \ldots, y_m) \in \{0,1\}^m$. Output the R-compression of the $\mathrm{OR}_m \circ L$ instance selected by y. That is, output $\mathsf{R}\left((x_{i,y_i})_{i \in [\ell]}\right) \in \{0,1\}^{m/100}$.

We prove that the above constitutes a collision-resistant hash function family, by showing that any successful collision finder would violate one-sided average-case hardness of L.

Suppose there exists a (non-uniform) polynomial-time algorithm A for which

$$\Pr_{\substack{s \leftarrow \mathsf{Gen}(1^n) \\ (y,y') \leftarrow \mathsf{A}(s)}} [(y \neq y') \wedge (\mathsf{Eval}(s, y) = \mathsf{Eval}(s, y'))] = \epsilon.$$

Consider the following associated algorithm A', which receives as input a bit string $x \in \{0,1\}^n$ and outputs a bit (corresponding to a prediction for $x \in L_Y$ or L_N). Intuitively, A' embeds the input x into a random index of the hash function description, runs the collision-finder A, and outputs 1 if A successfully finds a collision which differs in the embedded index.

Algorithm 1 Algorithm $\mathsf{A}'(x)$.

1. Select a random index $(i^*, b^*) \leftarrow [m] \times \{0,1\}$; let $x_{i^*,b^*} := x$.
   For every $(i,b) \neq (i^*,b^*)$, sample $x_{i,b} \leftarrow \mathsf{N}$. Set $s = (x_{i,b})_{i\in[m],b\in\{0,1\}}$.
2. Execute $(y, y') \leftarrow \mathsf{A}(s)$.
3. If it holds that: (1) $y \neq y'$, (2) $\mathsf{Eval}(s,y) = \mathsf{Eval}(s,y')$, and (3) $y_{i^*} \neq y'_{i^*}$, then output 1.
   Else, output a randomly selected bit $c \leftarrow \{0,1\}$.

By the strong one-sided average-case hardness of L, there exists a negligible function ν' and (possibly inefficient) sampler algorithm $\mathsf{Y}_{A'}$ corresponding to A', for which

$$\left| \Pr_{x\leftarrow\mathsf{N}(1^n)}[\mathsf{A}'(x)=1] - \Pr_{x\leftarrow\mathsf{Y}_{\mathsf{A}'}(1^n)}[\mathsf{A}'(x)=1] \right| \leq \nu'(n). \tag{1}$$

Claim 1: $\Pr_{x\leftarrow\mathsf{N}(1^n)}[\mathsf{A}'(x)=1] \geq \epsilon/m$. Given $x \leftarrow \mathsf{N}(1^n)$, the value of s as generated by A' is identically distributed to that of $\mathsf{Gen}(1^n)$, independent of the selected choice of $i^* \in [m]$. This implies that

$$\begin{aligned}\Pr_{x\leftarrow\mathsf{N}(1^n)}[\mathsf{A}'(x)=1] &= \Pr_{\substack{s\leftarrow\mathsf{Gen}(1^n)\\(y,y')\leftarrow\mathsf{A}(s)}}[(y\neq y') \wedge (\mathsf{Eval}(s,y)=\mathsf{Eval}(s,y')) \wedge (y_{i^*}\neq y'_{i^*})] \\ &\geq \Pr_{\substack{s\leftarrow\mathsf{Gen}(1^n)\\(y,y')\leftarrow\mathsf{A}(s)}}[(y\neq y') \wedge (\mathsf{Eval}(s,y)=\mathsf{Eval}(s,y'))] \cdot \frac{1}{m} = \frac{\epsilon}{m}.\end{aligned}$$

Claim 2: $\Pr_{x\leftarrow\mathsf{Y}_{\mathsf{A}'}(1^n)}[\mathsf{A}'(x)=1] = 0$. Assuming perfect correctness $\Pr_{x\leftarrow\mathsf{Y}_{\mathsf{A}'}}[x \in L_Y] = 1$ and $\Pr_{x\leftarrow\mathsf{N}}[x \in L_N] = 1$, then the embedded instance satisfies $x_{i^*,b^*} \in L_Y$, whereas the ambient instances are in $x_{i,b} \in L_N$ for $(i,b) \neq (i^*,b^*)$. This means for any $y, y' \in \{0,1\}^m$ in which $y_{i^*} \neq y'_{i^*}$, the corresponding selected $\mathrm{OR}_m \circ L$ instances *necessarily* disagree: $(x_{i,y_i})_{i\in[m]} \in (\mathrm{OR}_m \circ L)_Y$ and $(x_{i,y'_i})_{i\in[m]} \in (\mathrm{OR}_m \circ L)_N$, or vice versa. However, by perfect correctness of the compression algorithm R, this implies $\mathsf{R}((x_{i,y_i})_{i\in[m]}) \neq \mathsf{R}((x_{i,y'_i})_{i\in[m]})$. Thus, for any $y, y' \in \{0,1\}^m$ with $y_{i^*} \neq y'_{i^*}$, it must be the case that $\mathsf{Eval}(s,y) \neq \mathsf{Eval}(s,y')$.

Combining the two above claims together with Equation (1) implies that the collision-finding success probability ϵ of A must be bounded above by a negligible value. The theorem follows. ◀

References

1. Benny Applebaum. Cryptographic Hardness of Random Local Functions - Survey. *Computational Complexity*, 25(3):667–722, 2016. doi:10.1007/s00037-015-0121-8.
2. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 166–175. IEEE Computer Society, 2004. doi:10.1109/FOCS.2004.20.
3. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. From Secrecy to Soundness: Efficient Verification via Secure Computation. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, volume 6198 of *Lecture Notes in Computer Science*, pages 152–163. Springer, 2010. doi:10.1007/978-3-642-14165-2_14.
4. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Minimizing Locality of One-Way Functions via Semi-private Randomized Encodings. *J. Cryptology*, 31(1):1–22, 2018. doi:10.1007/s00145-016-9244-6.

5 Benny Applebaum and Pavel Raykov. On the Relationship Between Statistical Zero-Knowledge and Statistical Randomized Encodings. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 449–477. Springer, 2016. doi:10.1007/978-3-662-53015-3_16.

6 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264.

7 Donald Beaver, Silvio Micali, and Phillip Rogaway. The Round Complexity of Secure Protocols (Extended Abstract). In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 503–513. ACM, 1990. doi:10.1145/100216.100287.

8 Itay Berman, Akshay Degwekar, Ron D. Rothblum, and Prashant Nalini Vasudevan. Statistical Difference Beyond the Polarizing Regime. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:38, 2019. URL: https://eccc.weizmann.ac.il/report/2019/038.

9 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009. doi:10.1016/j.jcss.2009.04.001.

10 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 151–160. ACM, 2013. doi:10.1145/2488608.2488628.

11 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

12 Andrew Drucker. New Limits to Classical and Quantum Instance Compression. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 609–618, 2012.

13 Andrew Drucker. New Limits to Classical and Quantum Instance Compression. *SIAM J. Comput.*, 44(5):1443–1479, 2015. doi:10.1137/130927115.

14 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011. doi:10.1016/j.jcss.2010.06.007.

15 Oded Goldreich. A Note on Computational Indistinguishability. *Inf. Process. Lett.*, 34(6):277–281, 1990. doi:10.1016/0020-0190(90)90010-U.

16 Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001.

17 Oded Goldreich, Russell Impagliazzo, Leonid A. Levin, Ramarathnam Venkatesan, and David Zuckerman. Security Preserving Amplification of Hardness. In *FOCS*, pages 318–326. IEEE Computer Society, 1990.

18 Danny Harnik and Moni Naor. On the Compressibility of *NP* Instances and Cryptographic Applications. *SIAM J. Comput.*, 39(5):1667–1713, 2010. doi:10.1137/060668092.

19 Yuval Ishai and Eyal Kushilevitz. Randomizing Polynomials: A New Representation with Applications to Round-Efficient Secure Computation. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 294–304, 2000.

20 Yuval Ishai and Eyal Kushilevitz. Perfect Constant-Round Secure Computation via Perfect Randomizing Polynomials. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, pages 244–256. Springer, 2002. doi:10.1007/3-540-45465-9_22.

21 Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Sufficient Conditions for Collision-Resistant Hashing. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 445–456. Springer, 2005. doi:10.1007/978-3-540-30576-7_24.

22 Richard J. Lipton and Neal E. Young. Simple strategies for large zero-sum games with applications to complexity theory. In Frank Thomson Leighton and Michael T. Goodrich, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 734–740. ACM, 1994. doi:10.1145/195058.195447.

23 Moni Naor and Guy N. Rothblum. Learning to impersonate. In William W. Cohen and Andrew Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 649–656. ACM, 2006. doi:10.1145/1143844.1143926.

24 Tatsuaki Okamoto. On Relationships between Statistical Zero-Knowledge Proofs. *J. Comput. Syst. Sci.*, 60(1):47–108, 2000. doi:10.1006/jcss.1999.1664.

25 Rafail Ostrovsky. One-Way Functions, Hard on Average Problems, and Statistical Zero-Knowledge Proofs. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference, Chicago, Illinois, USA, June 30 - July 3, 1991*, pages 133–138, 1991.

26 Rafail Ostrovsky and Avi Wigderson. One-Way Fuctions are Essential for Non-Trivial Zero-Knowledge. In *Second Israel Symposium on Theory of Computing Systems, ISTCS 1993, Natanya, Israel, June 7-9, 1993, Proceedings*, pages 3–17. IEEE Computer Society, 1993. doi:10.1109/ISTCS.1993.253489.

27 Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003. doi:10.1145/636865.636868.

28 Daniel R. Simon. Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions? In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998. doi:10.1007/BFb0054137.

29 Salil Pravin Vadhan. *A study of statistical zero-knowledge proofs*. PhD thesis, Massachusetts Institute of Technology, 1999.

30 Andrew Chi-Chih Yao. Theory and Applications of Trapdoor Functions (Extended Abstract). In *FOCS*, pages 80–91. IEEE Computer Society, 1982.

31 Andrew Chi-Chih Yao. How to Generate and Exchange Secrets (Extended Abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986. doi:10.1109/SFCS.1986.25.

A OWFs and One/Two-Sided Average-Case Hardness

In this section, we restate and prove the lemmas stated in Section 2 about average-case hardness and reductions, and their connections to one-way functions.

A.1 Proof of Lemma 6

▶ **Lemma 6.** *If there is a problem that is weak one-sided average-case hard with perfect sampling, then One-Way Functions exist.*

Proof. Let L be weak one-sided average-case hard with perfect sampling, and N the poly-time sampling algorithm, such that there is a constant c such that for any poly-time distinguisher D there is Y_D such that for almost all n, $|\Pr_r[\mathsf{D}(\mathsf{N}(1^n;r))=1] - \Pr[\mathsf{Y}_D(1^n;r)=1]| < 1 - 1/n^c$. Moreover, $\mathrm{Supp}(\mathsf{N}) \subseteq L_N$ and $\mathrm{Supp}(\mathsf{Y}) \subseteq L_Y$.

Let F be defined for output length n as $\mathsf{F}_n(x) := \mathsf{N}(1^n; x)$. We will show that F is a weak-one-way function. The lemma then follows from classical results. [30, 17, 16]

Suppose, for the sake of contradiction, there exists an efficient A that can invert F_n with probability $> 1 - 1/n^c$, for infinitely many n. Then, consider the (efficient) distinguisher D that simply checks if A was succesful. In particular, D on input x simply:

1. $y \leftarrow \mathsf{A}(x)$
2. If $\mathsf{F}_n(y) = x$, output 1. Otherwise, output 0.

Because A has advantage $1 - 1/n^c$ on F_n, $\Pr_r[\mathsf{A}(\mathsf{N}(1^n; r)) = 1] > 1 - 1/n^c$, for infinitely many n. Now by our assumption on L, there exists Y_D such that $|\Pr_r[\mathsf{D}(\mathsf{N}(1^n; r)) = 1] - \Pr_r[\mathsf{D}(\mathsf{Y}_D(1^n; r)) = 1]| < 1 - 1/n^c$ and $\mathrm{Supp}(\mathsf{Y}_D) \cap \mathrm{Supp}(\mathsf{N}) = \emptyset$, for almost all n. It follows from the latter condition that A will never find preimages (under F) for outputs from Y_D, because they don't exist. It follows that $\Pr_r[\mathsf{D}(Y_D(1^n; r))] = 0$.

But then, $\Pr[\mathsf{D}(\mathsf{N}) = 1] - \Pr[\mathsf{D}(\mathsf{Y}_D) = 1] > 1 - 1/n^c$ for infinitely many n, which contradicts the weak one-sided average-case hardness of L. ◀

A.2 Proof of Lemma 9

▶ **Lemma 9.** *Suppose there is a two-sided average-case Karp reduction from a language L to a language L', and L is worst-case hard. Then, L' is two-sided average-case hard.*

Let Y, N be the samplers guaranteed by the two-sided-ness of the reduction from L to L', R. Let Y, N denote the random variable distributed according to Y and N evaluated on random inputs, respectively.

It suffices to show that for every efficient A, $|\Pr[\mathsf{A}(Y) = 1] - \Pr[\mathsf{A}(Y) = 1]| \leq 3/10$ (for almost all n).

Suppose not, and let A be a counter-example to the above in that $\Delta(\mathsf{A}(Y); \mathsf{A}(N)) > 3/10$ for infinitely many n. Moreover, suppose without loss of generality that $\Pr[\mathsf{A}(Y) = 1] > \Pr[\mathsf{A}(N) = 1]$. Then define A' to be the algorithm for L that on input x: (i) runs the reduction to get $z \leftarrow \mathsf{R}(x)$, then (ii) runs $b \leftarrow \mathsf{A}(z)$, and finally (iii) outputs b. Let R_x be the random variable distributed according to $\mathsf{R}(x)$. By definition, it is the case that $\Delta(R_x; Y) \leq 1/10$ for all $x \in L_Y$ and similarly $\Delta(R_x; N) \leq 1/10$ for all $x \in L_N$. It follows that $\Delta(\mathsf{A}(R_x); \mathsf{A}(N)) \leq 1/10$ for all $x \in L_Y$ and similarly $\Delta(\mathsf{A}(R_x); \mathsf{A}(Y)) \leq 1/10$ for all $x \in L_N$.

Consequently,

$$\begin{aligned} x \in L_Y &\implies \Pr[\mathsf{A}'(x) = 1] = \Pr[\mathsf{A}(R_x) = 1] \geq \Pr[\mathsf{A}(Y) = 1] - 1/10 \\ x \in L_N &\implies \Pr[\mathsf{A}'(x) = 1] = \Pr[\mathsf{A}(R_x) = 1] \leq \Pr[\mathsf{A}(N) = 1] + 1/10 \\ &\qquad\qquad \leq \Pr[\mathsf{A}(Y) = 1] - 2/10. \end{aligned}$$

It follows from standard amplification that L is not worst-case hard, contradicting our assumption on L.

A.3 Proof of Lemma 11

▶ **Lemma 11.** *Suppose there is a one-sided average-case Karp reduction from a language L to a language L', and L is worst-case hard. Then, L' is one-sided average-case hard. Further, if the reduction is weak and has perfect sampling, then the hardness is also weak and has perfect sampling.*

Let R be the one-sided average-case Karp reduction from L to L' with a corresponding NO-instance sampler N. In order to show that L' is one-sided average-case hard, we need to show a NO-instance sampler for L' and, for any polynomial-time A, a YES-distribution Y_A that it cannot distinguish from it. Our NO-instance sampler will be the N from the reduction itself, which satisfies the property that its samples are in L'_N except with probability 0.1.

Given a polynomial-time algorithm A, we claim that, for all large enough n, there exists an $x_y \in L_Y \cap \{0,1\}^n$ such that A cannot distinguish between $\mathsf{R}(x_y)$ and $\mathsf{N}(1^n)$ with advantage more than 0.3. If this were not the case, that is, if for every $x \in L_Y \cap \{0,1\}^n$, the algorithm A could distinguish between $\mathsf{R}(x)$ and $\mathsf{N}(1^n)$ with advantage more than 0.3, then the algorithm A' that, on input x, estimates the probability that $\mathsf{A}(\mathsf{R}(x)) = 1$ up to error say 0.01 could be used to decide L on all instances $x \in (L_Y \cup L_N) \cap \{0,1\}^n$, as for all $x \in L_N$, the reduction promises that $\mathsf{R}(x)$ is at most 0.1-far from $\mathsf{N}(1^n)$. The distribution $\mathsf{Y}_\mathsf{A}(1^n)$ is simply $\mathsf{R}(x_y)$ for such an x_y, and the reduction guarantees that $\mathsf{R}(x_y)$ is contained in L_Y except with probability 0.1, as necessary.

Note that if R is a weak one-sided average-case reduction with perfect sampling, then for any $x \in L_Y, \Pr[\mathsf{R}(x) \notin L'_Y] = 0$, $\Pr[\mathsf{N}(1^n) \notin L'_N] = 0$, and there is a constant c such that for any $x \in L_N$, $\Delta(\mathsf{N}(1^n); \mathsf{R}(x)) \leq 1 - n^{-c}$. Then, again, for any efficient A it follows that $\Delta(\mathsf{A}(\mathsf{N}(1^n)); \mathsf{A}(\mathsf{R}(x))) \leq 1 - n^{-c}$, for any $x \in L_Y$. Now, we can modify the A' above to approximate $\Pr[\mathsf{A}(\mathsf{R}(x)) = 1]$ to within a $\frac{1}{2n^c}$ factor. We claim, as above, that there is some $x_y \in L_Y$ $\Delta(\mathsf{A}(\mathsf{R}(x)); \mathsf{A}(\mathsf{N}(1^n))) \leq 1 - \frac{1}{4n^c}$. If not then for every $x \in L_Y \cup L_N$, $\Delta(\mathsf{A}(\mathsf{R}(x)); \mathsf{A}(\mathsf{N}(1^n))) > 1 - \frac{1}{4n^c}$ and A' will output such that $\mathsf{A}'(x) = L(x)$ with overwhelming probability. It follows that such an x_y exists and again we can take $\mathsf{Y}_A(1^n) = \mathsf{R}(x_y)$. Because $x \in L_Y, \Pr[\mathsf{R}(x) \notin L'_Y] = 0$, it follows that $\Pr[\mathsf{Y}_A(1^n) \notin L'_Y] = 0$.

B Proofs for Section 4

In this section, we restate and prove the lemmas used in Section 4. The proofs of Lemmas 22 and 23 are based on the following lemma that is implicit in [13], in particular following from the proofs of Theorem 7.1 and Claim 7.2 there. While the statement there is in terms of the compression (where the output length of the reduction itself is restricted to t bits), it may be verified that the proof only uses lossiness as in Definitions 19 and 21.

▶ **Lemma 34** ([13]). *Suppose, for some polynomials $m : \mathbb{N} \to \mathbb{N}$ and $t : \mathbb{N} \to \mathbb{R}$, a problem L has a t-lossy OR_m-reduction to a problem L', and the reduction has error $\epsilon(n) < 0.5$. Let*

$$\delta(n) = \min\left\{\sqrt{\frac{\ln 2}{2} \cdot \frac{t(n)+1}{m(n)}}, 1 - 2^{-\frac{t(n)}{m(n)}-3}\right\}$$

Then, for any constant c, there are polynomial-time algorithms A and B such that:

- $\Pr[\mathsf{A}(1^n) \notin L'_N] \leq \epsilon$
- *For any $x \in L_N \cap \{0,1\}^n$, $\Delta(\mathsf{A}(1^n); \mathsf{B}(x)) \leq \delta(n) + n^{-c}$*
- *For any $x \in L_Y \cap \{0,1\}^n$, $\Pr[\mathsf{B}(x) \notin L'_Y] \leq \epsilon$*

We now use this to prove the following lemmas.

▶ **Lemma 22.** *Suppose, for some polynomial m and problems L, L', there is a reduction from $\mathrm{OR}_m \circ L$ to L'. If this reduction is $(m/100)$-lossy and $m(n) > 100$ for all large enough n, then there is a one-sided average-case Karp reduction from L to L'.*

Proof of Lemma 22. The reduction from the hypothesis has error at most 0.1, and is t-lossy for $t = m/100$. Applying Lemma 34 with $c = 1$ gives us algorithms A and B as in its statement, with $\delta(n) < \sqrt{\ln 2/100}$ for large enough n. We get a one-sided Karp reduction from L to L' with the algorithm B as the reduction and A the corresponding NO-instance sampler. ◀

▶ **Lemma 23.** *Suppose, for some polynomial m and problems L, L', there is a* perfect *reduction from the problem $\mathrm{OR}_m \circ L$ to L'. If this reduction is $O(m \log n)$-lossy, then there is a weak one-sided average-case Karp reduction with perfect sampling from L to L'.*

Proof of Lemma 23. Here, the reduction has error 0, and is t-lossy for $t = c'm \log n$ for some constant c'. Applying Lemma 34 with $c = c' + 2$, we get $\delta(n) = 1 - 2^{-(c' \log n + 3)}$, which is less than $1 - 1/n^{c-1}$ for large enough n. We get a weak-perfect one-sided reduction from L to L' with the algorithm B as the reduction and A as the corresponding NO-instance sampler. ◀

Closely along the lines of Drucker [13], we show the following analogue of Lemma 34 for Majority reductions, and use it to prove Lemma 24. We defer the proof of this lemma to Appendix B.1.

▶ **Lemma 35.** *Suppose, for some polynomials $m : \mathbb{N} \to \mathbb{N}$ and $t : \mathbb{N} \to \mathbb{R}$, a problem L has a t-lossy MAJ_m-reduction to a problem L', and the reduction has error $\epsilon(n) < 0.5$. Let*

$$\delta(n) = \min\left\{\sqrt{\frac{(t+1)\ln 2}{m+1}}, 1 - 2^{-\frac{2t}{m+2}-3}\right\}$$

Then, for any constant c, there are polynomial-time algorithms Y, N and R such that:

- $\Pr[\mathsf{N}(1^n) \notin L'_N] \leq \epsilon$
- $\Pr[\mathsf{Y}(1^n) \notin L'_Y] \leq \epsilon$
- *For any $x \in L_N \cap \{0,1\}^n$, $\Delta(\mathsf{N}(1^n); \mathsf{R}(x)) \leq \delta(n) + n^{-c}$*
- *For any $x \in L_Y \cap \{0,1\}^n$, $\Delta(\mathsf{Y}(1^n); \mathsf{R}(x)) \leq \delta(n) + n^{-c}$*

▶ **Lemma 24.** *Suppose, for some polynomial m and problems L, L', there is a reduction from the problem $\mathrm{MAJ}_m \circ L$ to L'. If this reduction is $m/100$-lossy and $m(n) > 100$ for all large enough n, then there is a two-sided average-case Karp reduction from L to L'.*

Proof of Lemma 24. The reduction we start with has error 0.1 and is t-lossy for $t = m/100$ bits. Applying Lemma 35 with $c = 1$, we get algorithms R, Y and N with $\delta < \sqrt{\ln 2/100}$ for all large enough n. We get a two-sided Karp reduction from L to L' with the algorithm Y as the YES-instance sampler, N as the NO-instance sampler, and R as the reduction. ◀

B.1 Proof of Lemma 35

Throughout this subsection, we will be concerned with a (possibly randomized) mapping $\mathsf{F} : (\{0,1\}^n)^m \to \{0,1\}^*$ for some odd m and $n \in \mathbb{N}$. The following notation will be useful in our discussions. Suppose D_0 and D_1 are distributions over $\{0,1\}^n$. By $\mathsf{F}(D_0^k, D_1^\ell)$, we denote the execution of F on m inputs, k of which are sampled from D_0, and ℓ from D_1, and the whole set of samples is randomly permuted before being input to F. For an $x \in \{0,1\}^n$, $\mathsf{F}(D_0^k, D_1^\ell, x)$ is similar, except along with the $(k + \ell)$ samples, x is also inserted. We will use the following concept of distributional stability that is similar to the one used by Drucker [13], but specialised for pairs of distirbutions.

▶ **Definition 36.** *Let $m, n, k \in \mathbb{N}$, with $m = 2k + 1$, and $\delta \in [0, 1]$. A (possibly randomized) mapping $\mathsf{F} : (\{0,1\}^n)^m \to \{0,1\}^*$ is said to be* δ-distributionally stable *(denoted δ-DS) over a pair of distributions (D_0, D_1) over $\{0,1\}^n$ if the following holds:*

$$\mathop{\mathbb{E}}_{x \leftarrow D_0} \left[\Delta\left(\mathsf{F}(D_0^k, D_1^k, x); \mathsf{F}(D_0^{k+1}, D_1^k)\right)\right] \leq \delta$$

The proof of Lemma 35 follows from the following propositions.

▶ **Proposition 37.** *Let $m, n, k \in \mathbb{N}$, and $t \in \mathbb{R}^+$, with $m = 2k + 1$. Any (possibly randomized) mapping $\mathsf{F} : (\{0,1\}^n)^m \to \{0,1\}^*$ that is t-lossy on nm-bit inputs is δ-distributionally stable over any pair of distributions (D_0, D_1) over $\{0,1\}^n$, where $\delta = \min\left\{\sqrt{\frac{(t+1)\ln 2}{m+1}}, 1 - 2^{-\frac{2t}{m+2}-3}\right\}$.*

▶ **Proposition 38.** *Let $m, n, k \in \mathbb{N}$, with $m = 2k + 1$. Suppose $\mathsf{F} : (\{0,1\}^n)^m \to \{0,1\}^*$ is δ-DS over all pairs of distributions over $\{0,1\}^n$. For any pair of disjoint sets $S_0, S_1 \subseteq \{0,1\}^n$, any $\ell > 0$, and any $\nu \in (0, 1)$, there exists a distibution K over $S_0^\ell \times S_1^\ell$ such that:*

- *K is samplable in time $\mathsf{poly}(n, \ell, 1/\nu^2)$*
- *For any $x \in S_0$,*

$$\mathop{\mathbb{E}}_{(T_0,T_1)\leftarrow K} \left[\Delta\left(\mathsf{F}(U_{T_0}^k, U_{T_1}^k, x); \mathsf{F}(U_{T_0}^{k+1}, U_{T_1}^k)\right)\right] \leq \delta + 2m/\ell + \nu$$

- *For any $x \in S_1$,*

$$\mathop{\mathbb{E}}_{(T_0,T_1)\leftarrow K} \left[\Delta\left(\mathsf{F}(U_{T_1}^k, U_{T_0}^k, x); \mathsf{F}(U_{T_1}^{k+1}, U_{T_0}^k)\right)\right] \leq \delta + 2m/\ell + \nu$$

where U_T is the uniform distribution over the multiset T.

Proof of Lemma 35. Given a reduction A from $\mathrm{MAJ}_m \circ L$ to L' that is t-lossy and the constant c, we construct the algorithms Y, N, and R as required by the lemma. Fix a value of n and, for convenience, denote $m(n)$ and $t(n)$ by just m $(= 2k + 1)$ and t.

First, we apply Proposition 37 to A, which tells us that it is δ-DS over any pair of distributions over $\{0,1\}^n$, with δ as in the statement. This lets us apply Proposition 38 taking the sets S_0 and S_1 to be $L_N \cap \{0,1\}^n$ and $L_Y \cap \{0,1\}^n$, respectively, and with $\ell = 4mn^c$ and $\nu = 1/2n^c$ there. Proposition 38 then gives us a samplable distribution K over $L_N^\ell \times L_Y^\ell$.

The algorithm N, on input 1^n, samples (T_0, T_1) from the K obtained as above, and outputs a sample from $\mathsf{A}(U_{T_0}^{k+1}, U_{T_1}^k)$. The algorithm R is the same, except it outputs $\mathsf{A}(U_{T_1}^{k+1}, U_{T_0}^k)$. The algorithm R, on input x, similarly samples (T_0, T_1) and outputs $\mathsf{A}(U_{T_0}^k, U_{T_1}^k, x)$.

That the outputs of $\mathsf{N}(1^n)$ and $\mathsf{Y}(1^n)$ are contained in L'_N except with error ϵ follows from the fact that $T_0 \subseteq L_N$, $T_1 \subseteq L_Y$, and A has error at most ϵ.

Next, for any $x \in L_N \cap \{0,1\}^n$, we are interested in the following distance:

$$\begin{aligned}
\Delta\left(\mathsf{N}(1^n); \mathsf{R}(x)\right) &= \Delta\left(\mathsf{A}(U_{T_0}^{k+1}, U_{T_1}^k); \mathsf{A}(U_{T_0}^k, U_{T_1}^k, x)\right) \\
&\leq \Delta\left((T_0, T_1, \mathsf{A}(U_{T_0}^{k+1}, U_{T_1}^k)); (T_0, T_1, \mathsf{A}(U_{T_0}^k, U_{T_1}^k, x))\right) \\
&= \mathop{\mathbb{E}}_{(T_0,T_1)\leftarrow K} \left[\Delta\left(\mathsf{A}(U_{T_0}^{k+1}, U_{T_1}^k); \mathsf{A}(U_{T_0}^k, U_{T_1}^k, x)\right)\right] \\
&\leq \delta + 2m/\ell + \nu = \delta + 3/4n^c
\end{aligned}$$

where the first inequality follows from the data processing inequality, and the second from Proposition 38. The analogous guarantee for $x \in L_Y \cap \{0,1\}^n$ and $\mathsf{Y}(1^n)$ is shown in the same way. This proves the lemma. ◀

Proof of Proposition 37. We prove this by reduction to the simpler distributional stability lemma proved by Drucker [13]. Recall that $\mathsf{F}(D_0^{k+1}, D_1^k)$ is computed by sampling $x_1, \ldots, x_{k+1} \leftarrow D_0$ and $y_1, \ldots y_k \leftarrow D_1$, and a random permutation π over $[2k+1]$, and then computing the output as $\mathsf{F}(\pi(x_1, \ldots, x_{k+1}, y_1, \ldots, y_k))$. Alternatively, we can write this as the random variable $F(\Pi(\overline{X}, \overline{Y}))$. Similarly, we can write $\mathsf{F}(D_0^k, D_1^k, x)$ as $\mathsf{F}(\Pi(x, \overline{X'}, \overline{Y'}))$. The distance we are interested in is now:

$$\Delta\left(\mathsf{F}(\Pi(\overline{X}, \overline{Y})); \mathsf{F}(\Pi(x, \overline{X'}, \overline{Y}))\right)$$

We split the process of sampling Π into two parts. First, we sample the part of the permutation that maps the last k inputs (corresponding to $\overline{Y}$) to get an intermediate random variable $\widehat{\Pi}$, which is uniform over the set of permutations that all map their last k inputs to the same places, and then sample the final permutation π from this variable. We have:

$$\begin{aligned}\Delta\left(\mathsf{F}(\Pi(\overline{X}, \overline{Y})); \mathsf{F}(\Pi(x, \overline{X'}, \overline{Y}))\right) &\leq \Delta\left((\widehat{\Pi}, \mathsf{F}(\Pi(\overline{X}, \overline{Y}))); (\widehat{\Pi}, \mathsf{F}(\Pi(x, \overline{X'}, \overline{Y})))\right) \\ &= \mathop{\mathbb{E}}_{\widehat{\pi} \leftarrow \widehat{\Pi}} \left[\Delta\left(\mathsf{F}(\Pi_{\widehat{\pi}}(\overline{X}, \overline{Y})); \mathsf{F}(\Pi_{\widehat{\pi}}(x, \overline{X'}, \overline{Y}))\right)\right]\end{aligned}$$

where the first inequality is by the data processing inequality, and in the last expectation $\Pi_{\widehat{\Pi}}$ represents Π sampled conditioned on the last k inputs being mapped according to $\widehat{\pi}$.

For any $\widehat{\pi}$ sampled from $\widehat{\Pi}$, define the mapping $\mathsf{G}_{\widehat{\pi}}$ on input $(x_1, \ldots, x_{k+1})$ as the one obtained by sampling $(y_1, \ldots, y_k) \leftarrow \overline{Y}$, and running F with the y_i's in the positions assigned by $\widehat{\pi}$, and the x_i's in lexicographic order in the rest. We can now apply [13, Lemma 6.2], which is paraphrased below, to the quantity inside the first expectation with $\mathsf{G}_{\widehat{\pi}}$ as the mapping to get the proposition.

▶ **Proposition 39** ([13, Lemma 6.2])**.** *Let $m, n \in \mathbb{N}$, and $t \in \mathbb{R}^+$. Any (possibly randomized) mapping $\mathsf{F} : (\{0,1\}^n)^m \to \{0,1\}^*$ that is t-lossy on nm-bit inputs satisfies the following for any distribution D over $\{0,1\}^n$:*

$$\mathop{\mathbb{E}}_{x \leftarrow D}\left[\Delta\left(\mathsf{F}(D^k, x); \mathsf{F}(D^{k+1})\right)\right] \leq \delta$$

where $\delta = \min\left\{\sqrt{\frac{\ln 2}{2} \cdot \frac{t+1}{m}}, 1 - 2^{-\frac{t}{m}-3}\right\}$.

While the lemma in [13] was stated for compression as measured by output length, it may be verified that its proof there only uses lossiness. ◀

In order to prove Proposition 38, we need the following proposition that is proven in the same way as [13, Lemma 6.3].

▶ **Proposition 40.** *Suppose F is δ-DS over all pairs of distributions over $\{0,1\}^n$. For a pair of distributions (D_0, D_1), denote by $\widehat{D}_{0,\ell}$ (respectively $\widehat{D}_{1,\ell}$) the empirical distribution formed by taking ℓ samples from D_0 (respectively D_1). (Note that these are themselves random variables.) Then,*

$$\mathop{\mathbb{E}}_{\substack{\widehat{D}_{0,\ell}, \widehat{D}_{1,\ell} \\ x \leftarrow D_0}}\left[\Delta\left(\mathsf{F}(\widehat{D}_{0,\ell}^k, \widehat{D}_{1,\ell}^k, x); \mathsf{F}(\widehat{D}_{0,\ell}^{k+1}, \widehat{D}_{1,\ell}^k)\right)\right] \leq \delta + 2m/\ell$$

Proof of Proposition 38. We proceed by considering the following two-player zero-sum game:

- Player 1 chooses a pair of multisets $T_0 \subseteq S_0$ and $T_1 \subseteq S_1$, both of size ℓ
- Player 2 chooses a string $x \in S_0 \cup S_1$

- If $x \in S_0$, Player 2 gets a payoff of $\Delta\left(\mathsf{F}(U^k_{T_0}, U^k_{T_1}, x); \mathsf{F}(U^{k+1}_{T_0}, U^k_{T_1})\right)$, and if $x \in S_1$, it gets a payoff of $\Delta\left(\mathsf{F}(U^k_{T_1}, U^k_{T_0}, x); \mathsf{F}(U^{k+1}_{T_1}, U^k_{T_0})\right)$

The rest of the proof follows that of [13, Lemmas 6.4 and 6.5]. For any randomized strategy of Player 2, which is a distribution X over $S_0 \cup S_1$, consider the randomized strategy of Player 1 (T_0, T_1) where each element of T_0 (respectively T_1) is sampled from the distribution X restricted to S_0 (respectively S_1), denoted $X|_{S_0}$ (denoted $X|_{S_1}$). In the case where X is not supported in S_0 (respectively S_1), the set T_0 is chosen arbitrarily from S_0 (respectively T_1 from S_1). The payoff of this strategy for Player 2 is calculated as follows:

$$\begin{aligned}\Pr_{x \leftarrow X}\left[x \in S_0\right] \mathop{\mathbb{E}}_{x \leftarrow X|_{S_0}, T_0, T_1} \left[\Delta\left(\mathsf{F}(U^k_{T_0}, U^k_{T_1}, x); \mathsf{F}(U^{k+1}_{T_0}, U^k_{T_1})\right)\right] \\ + \Pr_{x \leftarrow X}\left[x \in S_1\right] \mathop{\mathbb{E}}_{x \leftarrow X|_{S_1}, T_0, T_1} \left[\Delta\left(\mathsf{F}(U^k_{T_1}, U^k_{T_0}, x); \mathsf{F}(U^{k+1}_{T_1}, U^k_{T_0})\right)\right]\end{aligned}$$

Noting that the elements of T_0 and T_1 are sampled from $X|_{S_0}$ and $X|_{S_1}$, respectively, and that F is δ-DS over all pairs of distributions over $\{0,1\}^n$, Proposition 40 implies that both the expectations above are at most $\delta + 2m/\ell$.

Thus, for any strategy of Player 2, there is a strategy of Player 1 such that Player 2's payoff is at most $\delta + 2m/\ell$. By the minimax theorem, there is a randomized strategy of Player 1 – a distribution over multisets (T_0, T_1) – such that for every move $x \in S_0 \cup S_1$ of Player 2, its payoff is at most $\delta + 2m/\ell$. Further, by the results of Lipton and Young [22, Theorem 2], there is a randomized strategy of Player 1 that corresponds to uniformly sampling from a set of pure strategies of size $O(n/\nu^2)$ and restricts Player 2's payoff to $\delta + 2m/\ell + \nu$. This proves the proposition. ◀