58th Annual IEEE Symposium on Foundations of Computer Science

# White-Box vs. Black-Box Complexity of Search Problems: Ramsey and Graph Property Testing

Ilan Komargodski, Moni Naor, Eylon Yogev

*Department of Computer Science and Applied Mathematics,*
*Weizmann Institute of Science, Rehovot 76100, Israel*
*Emails: {ilan.komargodski,moni.naor,eylon.yogev}@weizmann.ac.il*

***Abstract*—Ramsey theory assures us that in any graph there is a clique or independent set of a certain size, roughly logarithmic in the graph size. But how difficult is it to find the clique or independent set? If the graph is given explicitly, then it is possible to do so while examining a linear number of edges. If the graph is given by a black-box, where to figure out whether a certain edge exists the box should be queried, then a large number of queries must be issued. But what if one is given a program or circuit for computing the existence of an edge? This problem was raised by Buss and Goldberg and Papadimitriou in the context of TFNP, search problems with a guaranteed solution.**

**We examine the relationship between black-box complexity and white-box complexity for search problems with guaranteed solution such as the above Ramsey problem. We show that under the assumption that collision resistant hash function exist (which follows from the hardness of problems such as factoring, discrete-log and learning with errors) the white-box Ramsey problem is hard and this is true even if one is looking for a much smaller clique or independent set than the theorem guarantees.**

**In general, one cannot hope to translate all black-box hardness for TFNP into white-box hardness: we show this by adapting results concerning the random oracle methodology and the impossibility of instantiating it.**

**Another model we consider is the succinct black-box, where there is a known upper bound on the size of the black-box (but no limit on the computation time). In this case we show that for all TFNP problems there is an upper bound on the number of queries proportional to the description size of the box times the solution size. On the other hand, for promise problems this is not the case.**

**Finally, we consider the complexity of graph property testing in the white-box model. We show a property which is hard to test even when one is given the program for computing the graph. The hard property is whether the graph is a two-source extractor.**

***Keywords*-Search problems; the Ramsey problem; white-box complexity; black-box complexity; collision-resistant hashing**

## I. Introduction

Consider a setting where one is given a large object (e.g., a graph) and the goal is to find some local pattern (e.g., a certain subgraph) in the object or determine whether it satisfies some property. We investigate the relationship between the black-box setting, where access to the object is via oracle queries, and the white-box setting, where access to the object is given by a program or a circuit, in the context of search problems in which a solution is guaranteed[1] to exist and in the context of property testing.

**The Ramsey problem:** The Ramsey number $R(n)$ is the minimal number such that any graph on $R(n)$ vertices contains a clique or independent set of size $n$. The Ramsey theorem states that for any $n$, it holds that $R(n)$ is finite and moreover that $R(n) \leq 2^{2n}$. This guarantee raises the following question: *Given a graph with $2^{2n}$ nodes, how difficult is it to find $n$ nodes that are either a clique or an independent set?*

The standard proof of Ramsey's theorem is actually constructive and yields an algorithm that finds the desired clique or independent set, but explores a linear (in the graph size) number of nodes and edges. Is it necessary to explore a large portion of the graph? This of course depends on the representation of the graph and the computational model. In the black-box model, where the access to the graph is merely by oracle queries, Impagliazzo and Naor [1] observed that any randomized algorithm must make at least $\Omega(2^{n/2})$ queries before finding the desired clique or independent set. This was based on the fact that a random graph on $2^{2n}$ vertices has no clique or independent set of size $4n$ with high probability (see Section II-B).

In this work we are interested in the white-box model[2], where the above question is phrased as: *Given a Boolean circuit encoding the edges of a graph with $2^{2n}$ nodes, how difficult is it to find $n$ nodes that are either a clique or an independent set?* This question has been explicitly asked by Buss [2] and Goldberg and Papadimitriou [3] in the context of search problems in the complexity class TFNP. The class TFNP, defined by Megiddo and Papadimitriou [4],

This work is supported in part by a grant from the Israel Science Foundation No. 950/16. Ilan Komargodski is supported in part by a Levzion Fellowship. Moni Naor is the incumbent of the Judith Kleeman Professorial Chair.

[1] We are not talking about promise problems, but rather when there is a proof that the pattern exists.

[2] An example of a graph given as a white-box is the Hadamard graph, where the two inputs are treated as vectors over GF[2] and there is an edge if and only if the inner product between them is 1.

0272-5428/17 $31.00 © 2017 IEEE
DOI 10.1109/FOCS.2017.63

is the class of all search problems for which a solution is guaranteed to exist for every instance and verifying a solution can be done efficiently. Thus, the problem where the input is a graph defined by a circuit and the target is to find a clique or an independent set (of appropriate sizes) belongs to the class TFNP.

Our first result is an answer to this question. We show that under the assumption that collision resistant hash functions[3] exist, there exists an efficiently samplable distribution of circuits (circuits on $4n$ inputs representing graphs on $2^{2n}$ vertices), for which finding a clique or independent set of size $n$ is impossible for any polynomial-time (in $n$) algorithm.

We also prove a white-box lower bound of a similar flavor for a related problem known as the colorful Ramsey problem. While a graph can be viewed as the edges colored in one color and the non-edges in another, (a simple version of) the colorful Ramsey theorem says that given the complete graph on $2^{2n}$ vertices and any coloring of its edges using roughly $n/\log n$ colors, there must exist a monochromatic triangle (see Section II-B for the precise statement). The question is: given a circuit that represents such a colored graph, what is the computational complexity of finding a monochromatic triangle? We show that this is also hard: assuming collision resistant hash functions, finding a monochromatic triangle is impossible for polynomial-time (in $n$) algorithms.

Finally, we consider the bipartite version of the Ramsey problem and prove similar hardness results in the white-box setting. Specifically, we show a hardness result for finding a bi-clique or bi-independent set in a bipartite graph based on the assumption that multi-collision resistant hash functions exist. These are hash functions for which it is hard to find *multiple* inputs that hash to the same output.[4] To complement this result, we show the other direction: the hardness of the bipartite Ramsey problem *implies* the existence of multi-collision resistant hash functions.

**Impossibility of a generic transformation:** In the context of search problems, the black-box model (in which the algorithm has only query access to the function) has been extensively studied as it gives hope to prove *unconditional* query lower bounds (see Lovász et al. [5] for example). It is tempting to try and translate any query lower bound (in the black-box model) into a white-box lower bound using cryptographic assumption. We show that such a transformation is impossible to achieve in general for search problems in TFNP.[5] Specifically, we present a search problem in TFNP for which the black-box complexity is exponential but for *any* white-box implementation, there exists an algorithm that finds the solution in polynomial time. Our impossibility result is unconditional and does not rely on any cryptographic assumption. It is based on ideas stemming from Canetti et al. [6] concerning limitations of transferring cryptographic schemes that use random oracles to ones that do not appeal to them (see below). Specifically, the construction utilizes the work of Goldwasser and Kalai [7] on signature schemes using the Fiat-Shamir paradigm.

**The succinct black-box model:** In the black-box model, as we have discussed, solving the Ramsey problem requires polynomially many queries in the size of the graph (i.e. exponential in the subgraph we are looking for) and this is also the case for many other problems in TFNP, such as PPP, PLS, PPAD and CLS (see [8] and [9]). In this model, the size of the representation of the function is unbounded and the *running time* of the algorithm accessing the object via queries is unbounded. In contrast, in the white-box model the size of the representation of the object is limited. We consider the question of whether the representation of the function should indeed be unbounded in order to obtain hardness results and study the succinct black-box model (see Definition 4). In this model, the function is represented succinctly but the algorithm is unbounded and has only black-box access to the function.

For this model we show that *any problem in TFNP is easy* (and in particular, the Ramsey problem). That is, there exist a (deterministic) algorithm that performs only a *polynomial* number of queries (in the size of the representation of the function) and finds a solution. One interesting take-away from this result is that any exponential query lower bound (in the black-box model) for a problem in TFNP must use instances of functions (i.e., "boxes") of exponential size.

**White-box graph property testing lower bounds:** Property testing studies problems of the type: given the ability to perform queries concerning local properties of an object, decide whether the object has some (predetermined) global property, or it is *far* from having such a property. The complexity of a problem is determined by the number of queries required for an algorithm to decide the above correctly.

In all classical works in this field, access to the tested object is given via queries to a black-box. We study the complexity of property testing given a white-box representation. The object is represented implicitly as a program or a circuit and is given to the solver. The solver has to decide whether the object that is encoded in the circuit has a predefined property or not.

We show that cryptographic assumptions can be useful to prove that meaningful properties of graphs are hard to test in the white-box model by any efficient algorithm. The cryptographic assumption we rely on is the existence of a *collection of lossy functions* [10]. A collection of lossy

[3] A collision resistant hash is a hash function that shrinks by one bit such that it is hard to find two inputs that hash to the same output.

[4] Any collision resistant hash function is also a multi-collision resistant hash functions, but the other direction is *not* known.

[5] We note that our impossibility result only rules out a general transformation for all search problem in TFNP. It is an interesting question to find specific problems in TFNP that admit such a transformation.

functions consists of two families of functions. Functions in the first family are injective, whereas functions in the second family are lossy, namely the size of their image is significantly smaller than the size of their domain. The security requirement is that a description of a randomly chosen function from the first family is computationally indistinguishable from a description of a randomly chosen function from the second family.

We show that there exists a graph property such that, assuming a collection of lossy functions, there exists an efficiently samplable distribution over implicitly represented graphs over $2^n$ vertices for which testing whether the graph has the property or is far from having it cannot be decided by any polynomial-time (in $n$) algorithm. The property is whether the graph is a two-source extractor.

### A. Graph-hash product

Our white-box hardness results are based on a technique we call "the graph-hash product", where we generate a new graph from an existing one by embedding the nodes of the new graph via a hash function (see Definition 8). Depending on the properties of the hash function we get various results. The key property of this product operation is that if the hash function is collision resistant, we get that the new graph looks locally as the original one. All of our hardness results, including the hardness of (all variants of) the Ramsey problem and the hardness of the graph property testing, are based on variants of this technique.

The hash product technique is not restricted to graph problems: for example, assuming collision resistant hash functions, we prove hardness for finding a sunflower configuration in a large family of sets of the same size. This is a natural (total) search problem that arises from the famous sunflower lemma of Erdös and Rado [11]. We refer to the full version [12] for more information.

A similar graph-hash product was used by Krajícek [13] relating the proof complexity of the weak pigeonhole principle and the proof complexity of the Ramsey theorem.[6]

### B. Cryptographic assumptions and white-box lower bounds

For some search problems it is known how to obtain hardness in the white-box model under certain cryptographic assumptions. One of the first examples is due to Papadimitriou [14] who showed that the hardness of the class PPP (a subclass in TFNP) can be based on the existence of one-way *permutations* (the hardness can be also based on the existence of collision resistant hash functions). We refer to [15] for more information about the assumptions that lead to white-box hardness in TFNP.

**Obfuscation:** It has been recently shown that program obfuscation is very useful for proving white-box lower bounds for search problems. An obfuscator transforms a given program (say described as a Boolean circuit) into another "scrambled" circuit which is functionally equivalent by "hiding" its implementation details. One could hope to take the underlying black-box instance, obfuscate it and use this obfuscated version as the white-box instance. Obfuscation is a strong and (still) somewhat controversial assumption (see Ananth et al. [16] for a discussion), but if it could be used for a general transformation, then we would get a large class of white-box hardness results. However, there are a few obstacles in applying such an approach: First, Canetti et al. [6] (followed by the work of Goldwasser and Kalai [7]) showed that it is impossible to generically translate security of cryptographic primitives in the random oracle model into primitives in the standard setting. Second, ideal program obfuscators ("virtual black-box") do not exist for general functionalities [17], [18], so we have to work with weaker primitives such as indistinguishability obfuscation [18], [19]. One prominent instance of using indistinguishability obfuscation in order to prove white-box lower bounds was shown in the context of PPAD-hardness [20], [21], [9], [22], but it is hard to see how to use indistinguishability obfuscation for a more general transformation from black-box hardness to white-box hardness.

Our white-box hardness results do *not* use obfuscation at all and as such bypass the above issues. Furthermore, our techniques show that weaker (and much better studied) primitives can be used to hide information in a meaningful way.

## II. Preliminaries

Unless stated otherwise, the logarithms in this paper are base 2. For a distribution $\mathcal{D}$ we denote by $x \leftarrow \mathcal{D}$ an element chosen from $\mathcal{D}$ uniformly at random. For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \ldots, n\}$.

A function $\mathsf{negl}\colon \mathbb{N} \to \mathbb{R}^+$ is *negligible* if for every constant $c > 0$, there exists an integer $N_c$ such that $\mathsf{negl}(n) < n^{-c}$ for all $n > N_c$. Two sequences of random variables $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are *computationally indistinguishable* if for any probabilistic polynomial-time algorithm $A$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that $|\Pr[A(1^n, X_n) = 1] - \Pr[A(1^n, Y_n) = 1]| \leq \mathsf{negl}(n)$ for all $n \in \mathbb{N}$.

### A. Search problems in the black-box and white-box models

Let $\mathcal{F}_n = \{f\colon \{0,1\}^n \to \{0,1\}^n\}$ be the class of all circuits $f$ mapping $n$ bits into $n$ bits. We give a definition of a search problem for the family $\mathcal{F}_n$.[7]

**Definition 1.** *A* search problem $\mathcal{S}$ *is a relation on* $2q(n)$ *tuples. More precisely,* $\mathcal{S} = \cup_{n=1}^{\infty} \mathcal{S}_n$*, where* $\mathcal{S}_n \subseteq (\{0,1\}^n)^{q(n)} \times (\{0,1\}^n)^{q(n)}$ *for a polynomial* $q(\cdot)$*, such that: (i) for all* $f \in \mathcal{F}_n$*, there exist* $x_1, \ldots, x_{q(n)} \in \{0,1\}^n$

[6] We thank Pavel Hubácek for telling us about [13].

[7] We restrict our attention to the family $\mathcal{F}_n$ of length-preserving functions for simplicity.

*for which* $\big(x_1, \ldots, x_{q(n)}, f(x_1), \ldots, f(x_{q(n)})\big) \in \mathcal{S}$*, and (ii)* $\mathcal{S}$ *is computable in polynomial time in* $n$*. The class of all such search problems is denoted* **TFNP***.*

The tuple $(x_1, \ldots, x_{q(n)})$ is called the witness (i.e., the solution). In general, a witness is not necessarily given as a sequence of points in the domain $\{0,1\}^n$ but notice that any string can be encoded as such a sequence and so our definition is without loss of generality.

We mainly focus on three models of computation that differ either by the representation type of the function $f \in \mathcal{F}_n$ or by the complexity measure of the solver. The models that we define and study are the *black-box* model, the *white-box* model, and a new hybrid model we call *succinct black-box*. We also mention a fourth model we call the *efficient-succinct black-box* model. For the rest of this subsection, fix a polynomial $q = q(n)$ and a search problem $\mathcal{S} \subseteq (\{0,1\}^n)^q \times (\{0,1\}^n)^q$.

In the *black-box model*, an algorithm is required to solve the search problem $\mathcal{S}$ while given only oracle access to the function $f$. That is, the algorithm provides queries $x$ and gets back the results $y = f(x)$. The black-box complexity of a search problem $\mathcal{S}$ is the number queries needed to solve a search problem in the worst-case, while the running time is unbounded. This model was introduced and studied by Lovász et al. [5].

**Definition 2** (Black-box complexity)**.** *The black-box complexity of* $\mathcal{S}$*, denoted by* $\mathsf{bbc}(\mathcal{S})$*, is bounded by a function* $T(\cdot)$ *if there exists an algorithm* $A$ *that for sufficiently large* $n$ *and any* $f \in \mathcal{F}_n$*, makes at most* $T(n)$ *queries to* $f$ *and outputs* $x_1, \ldots, x_q$ *such that* $(x_1, \ldots, x_q, f(x_1), \ldots, f(x_q)) \in \mathcal{S}$*.*

In the *white-box model*, an algorithm is required to solve the search problem $\mathcal{S}$ while given an explicit representation of the function $f$ (as a circuit). The white-box complexity of $\mathcal{S}$ is the *running time* (as opposed to number of queries) needed (measured as a function of the size of the representation) to solve a search problem in the worst-case. In the white-box setting, we are mostly interested in solvers that run in polynomial-time in the size of the function.

**Definition 3** (White-box complexity)**.** *The white-box complexity of* $\mathcal{S}$*, denoted by* $\mathsf{wbc}(\mathcal{S})$*, is bounded by a function* $T(\cdot)$ *if there exists an algorithm* $A$ *that for sufficiently large* $n$*, given* $f \in \mathcal{F}_n$ *(as a circuit) runs in time* $T(|f|)$*, and outputs* $x_1, \ldots, x_q$ *such that* $(x_1, \ldots, x_q, f(x_1), \ldots, f(x_q)) \in \mathcal{S}$*.*

In the *succinct black-box model*, an algorithm is required to solve the search problem $\mathcal{S}$ while given only oracle access to the function $f$, however, as opposed to the black-box model, the succinct black-box complexity of a search problem $\mathcal{S}$ is measured by the number of queries required to solve the problem as a function of the *size of the representation* of $f$. In particular, if $f$ is represented succinctly by a polynomial-size (in $n$) circuit, then an efficient algorithm can perform only a polynomial number of queries (but its running time is unbounded).

**Definition 4** (Succinct black-box complexity)**.** *The succinct black-box complexity of* $\mathcal{S}$*, denoted by* $\mathsf{sbbc}(\mathcal{S})$*, is bounded by the function* $T(\cdot)$ *if there exists an algorithm* $A$ *that for sufficiently large* $n$ *and any* $f \in \mathcal{F}_n$*, makes at most* $T(|f|)$ *queries to* $f$ *and outputs* $x_1, \ldots, x_q$ *such that* $(x_1, \ldots, x_q, f(x_1), \ldots, f(x_q)) \in \mathcal{S}$*.*

We also consider a model we call the *efficient-succinct black-box model*, which is similar to the succinct black-box model, except that the solver's running is bounded (in the representation size).

### B. Ramsey theory

In this section we recall some basic definitions and facts from Ramsey theory and derive several bounds that will be useful for us later. We refer to Graham et al. [23] for a thorough introduction and history of Ramsey theory.

A Ramsey graph is a graph that contains no clique or independent set of some predefined sizes.

**Definition 5** (Ramsey graphs)**.** *A graph on* $N$ *vertices is called* $(s,t)$*-Ramsey if it contains no independent set of size* $s$ *and no clique of size* $t$*. A graph is called* $k$*-Ramsey if it is* $(k,k)$*-Ramsey.*

The classical result of Ramsey gives an upper bound on the size of a graph that does not contain either an independent set or a clique of some predefined size.

**Proposition 1.** *Every graph on* $N$ *vertices has either a clique or an independent set of size* $\frac{1}{2}\log N$*.*

A well-known (*non-explicit*) construction of a Ramsey graph was given by Erdös [24] as one of the first applications of the probabilistic method. He showed that most graphs on $N$ vertices are $(2\log N)$-Ramsey (see also the book of Alon and Spencer [25]). It was observed by Naor [26] that Erdös's proof actually gives a stronger statement: not only are most graphs $(2\log N)$-Ramsey, but such graphs can actually be sampled with relatively few bits of randomness (i.e., via a limited-independent family[8] or a small-bias probability space [27]). For completeness, the proof of the next statement is given in the full version [12]. No explicit construction of graphs matching these parameters is known. For a function $g\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ we define the corresponding graph $G$ on $n$ vertices where for any $u < v$ (lexicographic order) it holds that $(u,v)$ is an edge in $G$ iff $g(u,v) = 1$.

[8] A function family $\mathcal{H} = \{h\colon \mathcal{D} \to \mathcal{R}\}$ is $k$-wise independent, if $\Pr_{h \leftarrow \mathcal{H}}[h(x_1) = y_1 \vee h(x_2) = y_2 \vee \ldots \vee h(x_k) = y_k] = 1/|\mathcal{R}|^k$, for every distinct $x_1, x_2, \ldots, x_k \in \mathcal{D}$ and every $y_1, y_2, \ldots, y_k \in \mathcal{R}$.

**Proposition 2.** *A graph on $N$ vertices sampled via a $(2\log^2 N)$-wise independent hash function is a $(2\log N)$-Ramsey graph with probability $1 - 1/N^{\Omega(\log\log N)}$.*

Given that there are constructions of $k$-wise independent functions mapping $\{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ that are succinct (the size of the representation is polynomial in $n$ and $k$ even for $n$ output bits), the proposition implies that it is possible to sample a Ramsey graph (w.h.p.) with a succinct representation, i.e., the description length of the graph is polynomial in $n$. Furthermore, since computing a $(2\log^2 N)$-wise independent function can be done in time proportional to the size of the description, it is possible to sample a circuit that implicitly represents the graph.

The property of a graph being $(s,t)$-Ramsey can be equivalently phrased as a coloring property of the complete graph $K_N$ on $N$ vertices with two colors. Specifically, the function that defines whether there is an edge or not can be thought of a coloring of the full graph with two colors and the existence of a clique or an independent set of size $k$ is equivalent to the existence of a monochromatic subgraph of size $k$. This raises a natural generalization of the Ramsey property for graphs with multiple colors.

**Definition 6** (Colorful Ramsey graphs)**.** *A coloring $\psi\colon \binom{N}{2} \to [m]$ of the full graph $K_N$ with $m$ colors is called $(k_1,\ldots,k_m)$-Ramsey if there is no monochromatic subgraph of size $k_i$ colored with the color $i$, for every $i \in [m]$.*

The colorful Ramsey theorem provides, for a given number of colors, an upper bound on the size of a clique such that any coloring must result with a monochromatic subgraph of a predefined size.

**Proposition 3.** *For every $k > 2$ and $m > 1$, it holds that $R(\underbrace{k,\ldots,k}_{m \text{ times}}) \le m^{mk}$.*

As a corollary of Proposition 3, we obtain a bound on the number of colors that ensure the existence of a monochromatic subgraph of size $k$.

**Proposition 4.** *Consider the full graph on $N$ vertices. For every $k < \log N$, and every coloring $\psi\colon \binom{N}{2} \to [m]$, where $m = \frac{(\log N)/k}{\log\log N - \log k}$, there exists a monochromatic subgraph of size $k$.*

The proofs of Propositions 3 and 4 appear in the full version [12].

### C. Randomness extractors

We consider random variables supported on $n$-bit strings. A random variable $X$ is said to have min-entropy $k$ if for every $x \in \mathsf{Supp}(X)$ it holds that $\Pr[X = x] \le 2^{-k}$. Two random variables $X$ and $Y$ are said to be $\epsilon$-close if

$$\Delta(X,Y) \triangleq \frac{1}{2} \cdot \left( \sum_x |\Pr[X = x] - \Pr[Y = x]| \right) \le \epsilon$$

We say that a function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a $(k,\epsilon)$*-two-source extractor* if given any two independent distributions $X$ and $Y$ with min-entropy $k$ (each), then the distribution $\mathsf{Ext}(X,Y)$ is $\epsilon$-close to the uniform distribution on one bit [28].

It is known that every $(k,\epsilon)$-two-source extractor gives a $2^n \times 2^n$ Boolean matrix in which every minor of size at least $2^k \times 2^k$ has roughly the same number of 1's and 0's, namely, it has $1/2 \pm \epsilon$ fraction of 1's and 0's (and vice versa).

The probabilistic method shows that most functions are two-source extractors with very good parameters (in particular, they work for min-entropy $\log n + 2\log(1/\epsilon) + 1$), but obtaining explicit constructions for such functions has been a major open problem for a long time. In the last couple of years there has been remarkable progress and nearly optimal constructions are now known.

We will actually use the first construction of a two-source extractor given by Chor and Goldreich [28, Theorem 9]. They showed that the inner product function (also known as a Hadamard matrix) acts as a good two-source extractor for $k$ which is roughly $n/2$:

**Proposition 5.** *Let $k = k(n)$ and $\epsilon = \epsilon(n)$ be such that $2k \ge n + 2\log(1/\epsilon) + 2$. Then, the inner-product function is a $(k,\epsilon)$-two-source extractor.*

*In other words, the $2^n \times 2^n$ inner-products matrix has the property that every minor of size at least $2^k \times 2^k$ has $1/2 \pm \epsilon$ fraction of 1's and 0's.*

### D. Lossy functions and collision resistant hash functions

**Collision resistant hash:** Recall that a family of collision resistant hash (CRH) functions is one such that it is hard to find two inputs that hash to the same output. More formally, a sequence of families of functions $\mathcal{H}_n = \{h\colon \{0,1\}^{\ell_1(n)} \to \{0,1\}^{\ell_2(n)}\}$, where $\ell_1$ and $\ell_2$ are two functions such that $\ell_1(n) > \ell_2(n)$ for every $n \in \mathbb{N}$, is $\mathsf{collision\ resistant}$ if for every probabilistic polynomial-time algorithms $A$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\Pr_{h \leftarrow \mathcal{H}_n}[(x,x') \leftarrow A(1^n,h);\ h(x) = h(x')] \le \mathsf{negl}(n).$$

CRH functions are known to exist under a variety of hardness assumptions such as factoring, discrete-log, and Learning with Errors (LWE). They are *not* known to exist under the assumption that one-way functions exist[9], and there are oracle separation results for the two primitives [29].

By default, unless we say otherwise, when we assume the existence of CRH functions, then we assume a family as above in which every function shrinks its input by one bit. It is known that such an assumption is equivalent to a family in which every function shrinks by any fixed polynomial factor (by iteratively applying the hash polynomially-many times).

[9] In contrast, UOWHFs, Universal One-Way Hash Functions, where there is a fixed target $x$ and the goal is to find $x'$ that collides with it are known to exist under the assumption that one-way functions exist.

**Lossy functions:** A collection of lossy functions consists of two families of functions. Functions in the first family are injective, whereas functions in the second family are lossy, namely the size of their image is significantly smaller than the size of their domain. The security requirement is that a description of a randomly chosen function from the first family is computationally indistinguishable from a description of a randomly chosen function from the second family.

Lossy functions were introduced by Peikart and Waters [10] and shown to be useful for a variety of fundamental cryptographic applications. In particular, they were shown to imply collision resistant hash functions, oblivious transfer protocols, and chosen ciphertext-secure cryptosystems. Since their introduction they have found numerous other applications (see [30] for references).

**Definition 7** ([10])**.** *A collection of $(n, \ell)$-lossy functions is defined by a pair of algorithms $(G, F)$ such that:*

1) *$G(1^n, b)$, where $b \in \{0,1\}$, outputs a string $s \in \{0,1\}^{p(n)}$ for some fixed polynomial $p(\cdot)$. If $b = 0$, then the algorithm $F(s,\cdot)$ computes an injective function $f_s(\cdot)$ over $\{0,1\}^n$, and if $b = 1$, then the algorithm $F(s,\cdot)$ computes a function $f_s(\cdot)$ over $\{0,1\}^n$ whose image size is at most $2^{n-\ell}$.*
2) *The distribution of $G(1^n, 0)$ is computationally indistinguishable from the distribution of $G(1^n, 1)$.*

Lossy functions are known to exist under a variety of hardness assumptions such as Decisional Diffie-Hellman (DDH), Learning with Errors (LWE), and factoring related assumptions (Quadratic Residuosity and Phi-hiding) with different parameters [10], [31], [32], [30]. In our constructions, we will rely on lossy functions with polynomial shrinkage (e.g., $(n, n - n^{0.1})$-lossy functions). Such functions are known to exist based on LWE [10], DDH [30] and Phi-hiding assumptions [31] (but not based on Quadratic Residuosity). The construction of [31] gives a family of functions which are length-preserving.

## III. Hardness of The Ramsey Problem

We show a hard distribution for the Ramsey problem. In this problem, one is given an implicit and efficient representation of the adjacency matrix of a graph on $2^n$ vertices, and the goal is to find either a clique of size $n/2$ or an independent set of size $n/2$. The implicit representation of the graph is by a circuit $C \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ that represents the adjacency matrix of a graph on $2^n$ vertices.

In terms of Definition 1, we have the $q(n) = \binom{n/2}{2}$ and the relation $\mathcal{S}$ is such that $\big(x_1, \ldots, x_{q(n)}, f(x_1), \ldots, f(x_{q(n)})\big) \in \mathcal{S}$ if and only if the edges $x_1, \ldots, x_{q(n)}$ form a clique or an independent set of size $n$. That is, the set of vertices touched by some edge in $x_1, \ldots, x_{q(n)}$ is of size exactly $n$, and $f(x_1) = \ldots = f(x_{q(n)}) = b$ for some $b \in \{0,1\}$.[10]

**Hardness of the Ramsey problem:** We say that the Ramsey problem is *hard* if there exists an efficiently samplable distribution $\mathcal{D} = \{C \colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}\}$ over circuits of size polynomial in $n$ that represent graphs on $2^n$ vertices, such that for every probabilistic polynomial-time algorithm $A$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\Pr_{C \leftarrow \mathcal{D}}[v_1, \ldots, v_{n/2} \leftarrow A(1^n, C) \; ; \; v_1, \ldots, v_{n/2} \text{ form a clique or an independent set}] \leq \mathsf{negl}(n),$$

where the probability is over the uniform choice of $C \leftarrow \mathcal{D}$ and the randomness of $A$. All the efficiency requirements are polynomial in $n$.

The above problem is indeed in $\mathsf{TFNP}$ as it is guaranteed by Proposition 1 that there always exists a monochromatic clique or independent set of size $n/2$. We show that under a certain cryptographic assumptions, the existence of collision resistant hash (CRH) functions (see Section II-D), there exists an efficiently samplable distribution over instances of the Ramsey problem for which no efficient algorithm can find a solution. Recall that if CRH functions compressing by one bit exist, then CRH functions compressing by any polynomial factor (i.e. from $n$ bits to $n^\delta$ for any fixed constant $\delta > 0$) exist. We will use a collision resistant hash function family $\mathcal{H} = \{h \colon \{0,1\}^n \to \{0,1\}^{n/4}\}$.

**Theorem 1.** *The Ramsey problem is hard assuming the existence of collision resistant hash functions.*

In the proof of Theorem 1 we use a construction of Ramsey graphs given in Proposition 2 as well as a type of graph product operation: the operation takes as input a graph $G$ on $2^n$ vertices and a hash function $h \colon \{0,1\}^{n+\ell} \to \{0,1\}^n$, where $\ell \geq 1$ and outputs a graph $G \otimes h$ on $2^{n+\ell}$ vertices, whose edges depend on the edges in $G$ and the hash function.

**Definition 8** (The graph-hash product)**.** *Given a graph $G = (V, E)$, where $|V| = \{0,1\}^n$, and a hash function $h \colon \{0,1\}^{n+\ell} \to \{0,1\}^n$, define the graph $G \otimes h = (V', E')$ as a graph on vertices $V' = \{0,1\}^{n+\ell}$ with edges $E'$ such that $(u, v) \in E'$ if and only if $(h(u), h(v)) \in E$.*

Observe that given an efficient representation of $G$ and an efficient representation of $h$, we have an efficient representation of the graph $G \otimes h$.

*Proof of Theorem 1:* Let $k = n/4$, let $\mathcal{H}$ be a family of collision resistant hash function from $n$ bits to $k$ bits; such a family $\mathcal{H}$ exists under the assumption that collision resistant hash functions that compress by one bit exist. Let $\mathcal{G} = \{g \colon \{0,1\}^k \times \{0,1\}^k \to \{0,1\}\}$ be a $2k^2$-wise

[10] We say that an edge $(u, v)$ touches the vertices $u$ and $v$.

independent hash function family, where each member $g \in \mathcal{G}$ defines a graph $G$ on $2^k$ vertices in the natural way (see below). By Proposition 2, most $g \in \mathcal{G}$ define a graph $G$ that does not contain any clique or independent set of size $2k = n/2$. The following sampling procedure yields a graph $(V', E')$, where $|V'| = 2^n$:

1) Sample a collision resistant hash function $h \leftarrow \mathcal{H}$ and a function $g \leftarrow \mathcal{G}$.
2) Set $G = (V, E)$ to be the graph with $|V| = 2^k$ vertices induced by $g$ (see Proposition 2).
3) Output $h$ and $g$ as representing the graph-hash product $G \otimes h = (V', E')$. That is, for any $x, y \in V'$ s.t. $x < y$ we have that edge $(x, y)$ exists iff $g(h(x), h(y)) = 1$.

The Ramsey challenge on $(V', E')$ is to find a clique or independent set of size $n/2$ (since $|V'| = 2^n$). We reduce the ability of an adversary to solve the Ramsey problem to an adversary that breaks the collision resistance of $h \leftarrow \mathcal{H}$.

Suppose that there exists an adversary $A$ that, given an instance of the distribution above, finds a clique or an independent set of size $n/2 = 2k$ in $G \otimes h$ with probability $\epsilon > 0$ (over the choice of $h$, $g$, and the randomness of $A$). Denote this event by $\mathsf{Win}(A, g, h)$. That is, $\Pr[\mathsf{Win}(A, g, h)] \geq \epsilon$. Let $(v_1, \ldots, v_{2k})$ the solution found by $A$, and let $v'_i \triangleq h(v_i)$ for $i \in [2k]$. Let $\mathsf{Distinct}$ be the event in which in the solution output by $A$, the values $v'_1, \ldots, v'_{2k}$ are distinct. Then, by the assumption it holds that

$$\begin{aligned}&\Pr[\mathsf{Win}(A,g,h)] = \\ &\quad \Pr[\mathsf{Win}(A,g,h) \mid \mathsf{Distinct}] \cdot \Pr[\mathsf{Distinct}] + \\ &\quad \Pr[\mathsf{Win}(A,g,h) \mid \neg\mathsf{Distinct}] \cdot \Pr[\neg\mathsf{Distinct}] \geq \epsilon\end{aligned}$$

We first argue that $\Pr[\mathsf{Win}(A, g, h) \mid \mathsf{Distinct}] \leq \exp(-n)$. Indeed, by the definition of the event $\mathsf{Distinct}$, it holds that $v'_1, \ldots, v'_{2k}$ are distinct, and by the definition of our graph-hash product, the sequence of vertices $(v'_1, \ldots, v'_{2k})$ must form a clique or an independent set of size $2k$ in $G$. However, by Proposition 2 we know that with probability $1 - \exp(-n)$ over $g$, the graph $G$ *does not contain* any such independent set or clique.

Plugging this back into the above equation, we get

$$\Pr[\neg\mathsf{Distinct}] \geq \epsilon - \exp(-n)$$

Recall that $\epsilon$ is a non-negligible term and thus we obtain an algorithm $A'$ that finds a collision in $h$ with probability $\epsilon - \exp(-n)$, which contradicts the collision resistance property of the hash function $h$. To summarize, the algorithm $A'$ gets as input a hash function $h$, samples a function $g \leftarrow \mathcal{G}$, as above, and simulates the execution of $A$ on the graph-hash product graph $G \otimes h$. Given the output of $A$, it searches the output for a pair of values that form a collision relative to $h$ and outputs them (it outputs $\perp$ in case no such pair was found). By the above, two such colliding values will appear in the output with non-negligible probability, resulting in a collision relative to $h$. ■

**Hardness for finding a smaller clique or independent set:** We showed that it is hard to find a clique or independent set of size $n/2$ in an implicitly represented graph of size $2^n$. We can show that *finding a clique or independent set of size* $n^\delta$ for any $0 < \delta \leq 1$ is hard, by following the proof of Theorem 1 and using a hash function that maps $n$ bits into $n^\delta$ bits (which is implied by the existence of the hash function we used in Theorem 1).

We can even go below a fixed $\delta$ to, say, $n^{1/\sqrt{\log n}}$ by using a hash function that compresses a super-polynomial amount (from $n$ bits to $n^{1/\sqrt{\log n}}$ bits). This is known to be implied by a hash function that compresses a single bit albeit with a super-poly loss in security, but it is not known with only a poly loss.

**Ramsey theory and proof complexity:** Ramsey theory has been extensively studied in the context of proof complexity. In particular, it is known that Ramsey's theorem has a polynomial-size bounded-depth Frege proof [33] and it is related to the weak pigeonhole principle [34].

### A. Hardness of the colorful Ramsey problem

The colorful Ramsey problem asks, given an implicit and efficient representation of a coloring using $m$ colors of the edges of a graph on $2^n$ vertices, to find a monochromatic clique of size $k$. We will see a hard distribution for the colorful Ramsey problem. We focus in the case where the goal is to find a monochromatic triangle (i.e., $k = 3$ above) for simplicity and remark that the proof generalizes for larger values of $k$.

**Hardness of the colorful Ramsey problem:** We say that the colorful Ramsey problem is *hard* if there exists an efficiently samplable distribution $\mathcal{D} = \{\psi\colon \binom{2^n}{2} \to [m]\}$ over colorings of the full graph on $2^n$ vertices, such that for every probabilistic polynomial-time algorithm $A$, there exists a negligible function $\mathsf{negl}(\cdot)$ s.t.

$$\Pr_{C \leftarrow \mathcal{D}}[v_1, v_2, v_3 \leftarrow A(1^n, C)\ ;\ v_1, v_2, v_3 \\ \text{form a monochromatic triangle}] \leq \mathsf{negl}(n),$$

where the probability is over the uniform choice of $C \leftarrow \mathcal{D}$ and the randomness of $A$.

The above problem is indeed in $\mathsf{TFNP}$ whenever $m \leq n/(3\log n)$, since it is guaranteed by Proposition 4 that there always exists a monochromatic triangle if there are only $n/(3\log n)$ colors. The theorem below shows that there exists a distribution over instances of the colorful Ramsey problem for which no efficient algorithm can find a solution. As before, the security of the distribution relies on the existence of collision resistance hash functions. The proof is given in the full version [12].

**Theorem 2.** *The colorful Ramsey problem is hard assuming the existence of a collision resistant hash function family.*

### B. The Ramsey problem and Multi-CRH

In Theorem 1 we showed that under the assumption that CRH functions exist, the Ramsey problem is hard. Here we study the bipartite version of the Ramsey problem and point out a tight relationship to a cryptographic primitive we call *multi-collision resistant hash* (MCRH) functions.[11]

A bipartite graph on two sets of $N$ vertices is a bipartite $K$-Ramsey graph if it has no $K \times K$ complete or empty bipartite subgraph. Ramsey's theorem for such graphs says that every bipartite graph on $2N$ vertices has a $\log N \times \log N$ complete or empty bipartite subgraph (see e.g., [36]).[12] The result of Erdös [24] on the abundance of $(2 \log N)$-Ramsey graphs holds as is for bipartite graphs.

Analogously to the Ramsey problem on graphs, the *bipartite Ramsey problem* is when the graphs are bipartite and the goal is to find a bi-clique or bi-independent set of a certain size. We focus on the task of finding a bi-clique or bi-independent set of size $n/4$. We say that the *bipartite* Ramsey problem is *hard* if there exists an efficiently samplable distribution $\mathcal{D} = \{C\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}\}$ over circuits of size polynomial in $n$ that represent *bipartite* graphs on $2^n \times 2^n$ vertices, such that for every probabilistic polynomial-time algorithm $A$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\Pr_{C \leftarrow \mathcal{D}}[u_1, \dots, u_{n/4}, v_1, \dots, v_{n/4} \leftarrow A(1^n, C) \ ; \\ \exists b \in \{0,1\}, \forall i, j \in [n/4] \colon C(u_i, v_j) = b] \leq \mathsf{negl}(n),$$

where the probability is over the uniform choice of $C \leftarrow \mathcal{D}$ and the randomness of $A$. All the efficiency requirements are polynomial in $n$.

Roughly, a family of *multi-collision resistant hash* functions is one such that it is hard to find multiple inputs that hash to the same output. More precisely, a sequence of families of functions $\mathcal{H}_n = \{h\colon \{0,1\}^{\ell_1(n)} \to \{0,1\}^{\ell_2(n)}\}$, where $\ell_1$ and $\ell_2$ are two functions such that $\ell_1(n) > \ell_2(n)$ for every $n \in \mathbb{N}$, is *k-multi-collision resistant* if for every probabilistic polynomial-time algorithms $A$, it holds that

$$\Pr_{h \leftarrow \mathcal{H}_n}[(x_1, \dots, x_k) \leftarrow A(1^n, h);\ h(x_1) = \dots = h(x_k)] \\ \leq \mathsf{negl}(n).$$

By default, unless otherwise stated, we assume that a family of $k$-MCRH functions maps strings of length $n$ to strings of length $n - \log k$. This assumption ensures that a $k$-multi-collision exists (but yet it is hard to find). $k$-MCRH functions are implied by standard CRH functions (but is seemingly a weaker primitive).

We show that MCRH functions are sufficient and necessary for bipartite Ramsey hardness. The proof appears in the full version [12].

**Theorem 3.** *If the bipartite Ramsey problem is hard, then there exists a family $\mathcal{H} = \{h\colon \{0,1\}^n \to \{0,1\}^{n/2}\}$ of $n/4$-MCRH functions.*

*Furthermore, if there exists a family $\mathcal{H} = \{h\colon \{0,1\}^n \to \{0,1\}^{\sqrt{n}/8}\}$ of $\sqrt{n}$-MCRH functions, then the bipartite Ramsey problem is hard.*

**Subsequent work:** Following this work, the notion of MCRH has been studied in depth showing a variety of applications such as statistically-hiding commitments with short communication and various types of efficient zero-knowledge protocols [37], [38], [39].

## IV. Hardness of Testing an Extractor

In this section we present a *graph property* which is hard to test in the white-box setting. Specifically, we present a property $\Pi$ and a distribution over succinctly-represented graphs for which efficiently deciding whether an instance in the distribution has the property $\Pi$ or is *far* from having the property $\Pi$ is impossible (under appropriate cryptographic assumptions). We briefly recall the notions related to (graph) property testing and then describe our main result. A more elaborate introduction can be found in [40] and references therein.

A property $\Pi$ is simply a set of elements in a universe of interest. A property $\Pi$ is a graph property, if it is a set of graphs closed under graph isomorphism. That is, if for every graph $G = (V, E)$ on $N$ vertices and any permutation $\pi$ on $V$ it holds that $G \in \Pi$ if and only if $\pi(G) \in \Pi$, where $\pi(G) = (V, E')$ and $E' = \{(\pi(u), \pi(v)) \mid (u, v) \in E\}$. A graph $G = (V, E)$ on $N$ vertices is said to be $\epsilon$-far from a property $\Pi$ if for every $N$-vertex graph $G' = (V', E')$ that has the property $\Pi$ (i.e., $G' \in \Pi$), it holds that $|E \triangle E'| \geq \epsilon \cdot \binom{N}{2}$ (the operator $\triangle$ denotes symmetric difference).

**Definition 9** (White-box property testing)**.** *An $\epsilon$-tester for a graph property $\Pi$ is a probabilistic machine that, on input a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ representing the adjacency matrix of a $2^n$-vertex graph $G$, outputs a binary value that satisfies:*

1) *If $G$ has the property $\Pi$, then the tester outputs 1 with probability at least $2/3$.*
2) *If $G$ is $\epsilon$-far from $\Pi$, then the tester outputs 1 with probability at most $1/3$.*

The above definition naturally generalized to bipartite graphs (and properties of bipartite graphs).

**The property of being an extractor:** The graph property $\Pi$ we are interested in is being a *two-source extractor*: a bipartite graph $G = (U, V, E)$, where $|U| = |V| = 2^n$, is $(k, \delta)$-balanced if for every set $U' \subseteq U$ and $V' \subseteq V$ of

[11] Multiple collisions in hash functions were studied before in the context of *iterated* hash functions by Joux [35]. He showed that for such functions, finding multi-collisions (a set of $k$ messages that hash to the same value) is not much harder than finding ordinary collisions (pairs of messages that collide).

[12] Given a bipartite $K$-Ramsey graph $G$ on $2N$ vertices, one can transform it into a non-bipartite $2K$-Ramsey graph $H$ on $N$ vertices. The graph $H$ is defined by the upper triangle of the adjacency matrix of $G$.

size $|U'| = |V'| = 2^k$, the induced subgraph $G_{U',V'}$ has $1/2 \pm \delta$ fraction of edges. The induced subgraph $G_{U',V'} = (U', V', E_{U',V'})$ is defined by $(u, v) \in E_{U',V'}$ if and only if $(u, v) \in E$, $u \in U'$ and $v \in V'$.

We present a distribution over succinctly represented (bipartite) graphs for which testing the above property is hard. The hardness reduces to breaking the security of a *collection of lossy functions* described in Section II-D.

**Theorem 4.** *Assume the existence of a collection of $(n, 2n/3)$-lossy functions and consider the bipartite graph property $\Pi$ of being $(0.52n, 2^{-n/2000})$-balanced. There exist a constant $\epsilon > 0$ and a distribution over succinctly represented bipartite graphs on $2^n$ vertices for which any $\epsilon$-tester for $\Pi$ must run in super-polynomial-time.*

Observe that the existence of a collection of lossy functions directly implies white-box hardness of testing whether a given function is injective or far from being such (i.e., lossy), but the theorem proves hardness for a graph property. The proof of the theorem appears in the full version [12].

## V. Impossibility of a General Transformation

In this section, we show (unconditionally) that there cannot be a general transformation from a black-box lower bound, to a white-box lower bound. That is, we show that there exists a problem that has exponentially high black-box complexity, however, is solvable in polynomial time given any white-box implementation of the search function.

We first give an informal overview of the problem we define. Consider the problem of finding a small circuit that is consistent with a large set of pairs $(x_i, y_i)$. In particular, the set will be larger than the size of the circuit. In the black-box model, these points will be completely random and thus the task of finding a small circuit that is consistent is impossible (since one cannot compress random bits). On the other hand, in the white-box model, given any circuit that computes these points, the task becomes completely trivial: simply return the circuit in hand.

This approach raises two main difficulties. First, this problem does not always have a solution in the black-box model (which is not consistent with the definition of a search problem). Second, the solution has no a priori bound on its size.

The first problem is solved by taking any search problem with proven high black-box complexity (e.g., PPP or PWPP). Notice that this problem might have high white-box complexity as well. Then, we modify our search problem to be an OR of the two problems. That is, either find a small consistent circuit or solve the second search problem. In the black-box model, the complexity of the new problem remains high, and moreover, a solution always exists. In the white-box model, the problems remains solvable in polynomial time.

The second problem is solved as by instead of giving the circuit as a solution, giving a short *commitment* to the circuit and then proving that this commitment to a circuit is consistent on a random value. To achieve this, we use techniques such as Kilian's protocol combined with the Fiat-Shamir paradigm to remove interaction in the random oracle model (in the black-box model we have a random oracle!).

The search problem we define is the one considered by Goldwasser and Kalai [7] in the context of showing limitations for the Fiat-Shamir paradigm. They showed that there exists a 3-round public-coin identification scheme for which the Fiat-Shamir paradigm yields an insecure digital signature with any hash function in the standard model.

This signature scheme naturally gives rise to a search problem: Given the public parameters of the scheme, find a valid signature for an arbitrary message. To make this problem in TFNP, we define the problem of either finding a valid signature as above or finding a collision in a compressing function. The latter has a guaranteed solution so this defines a valid search problem in TFNP. More details can be found in the full version [12].

## VI. The Succinct Black-Box Model

We define and study a new model of computation which we call succinct black-box. In this model, as in the black-box model, the solver has only query access to the object and it is measured by the number of queries it performs in order to find a solution. However, in this model (as opposed to the black-box model), the number of queries is measured as a function of the size of the representation of the function. This is similar to the white-box model, where the running time is measured as a function of the size of the representation. In particular, if the function has a polynomial-sized representation, then an efficient algorithm in this model can perform only a polynomial number of queries (but the running-time may be arbitrary).

We show that for any problem in TFNP, there exists a deterministic procedure that performs only a *polynomial* number of queries (in the size of the representation of the function) and finds a valid solution. The proof appears in the full version [12].

**Theorem 5.** *For any search problem $\mathcal{S} \in$ TFNP it holds that* sbbc$(\mathcal{S})$ *is polynomial. In particular, if the representation size is $s$ and any solution is of size at most $k$, then the number of queries is $O(sk/\log k)$.*

The assumption that the search problem is in TFNP is essential for the theorem to hold. To see this, consider the case of *point functions* (functions that output 1 at a specific point and 0 everywhere else) where the goal is to find the hidden point. There exists a succinct representation (the point itself) but any algorithm that is only allowed to query the oracle must make exponentially many queries until it finds the hidden point.

Goldberg and Roth [41, Theorem 3.3] investigated the number of queries needed to find an $\epsilon$-well supported Nash equilibrium in multi-player games. They showed that if the game has a succinct representation, then there is an algorithm that performs a polynomial number of queries in the number of players $n$, the number of actions $m$, the description length of the target game, and finds such an equilibrium. One can view Theorem 5 as a generalization of that result.[13]

## VII. Further Research

The immediate direction this work raises is which other Ramsey-type problems are hard in the white-box model. Consider, for instance, Schur's Theorem that states that for every positive integer $m$, there exists a number $S(m)$ such that for every coloring of the numbers in the set $\{1, \ldots, S(m)\}$ with $m$ colors, there must be $x, y$ and $z$ colored with the same color such that $x + y = z$ (see [23], Chapter 3). This property naturally gives rise to the $m$-*Schur search problem*: Given an implicit representation of the coloring of the numbers $\{1, \ldots, S(m)\}$, find $x, y$ and $z$ colored with the same color and satisfy $x + y = z$. Can we argue that the $m$-Schur problem is $\mathsf{hard}$?

What are the minimal assumptions needed to obtain the hardness results for Ramsey? Are one-way functions sufficient or is there an inherent reason why collision resistant hash functions are needed? For the *bipartite* Ramsey problem, we showed that a relaxation of CRH functions (MCRH functions) is necessary and sufficient.

Our results are "obfuscation-free", in the sense that we needed much weaker primitives for obtaining them than in the recent works of [20], [21], [22]. Can we get similar results for showing the hardness of complexity classes such as $\mathsf{PLS}$ and $\mathsf{PPAD}$?

We showed the general impossibility of transferring black-box results to white-box results. One direction which might be fruitful is to find conditions on the search problems that *do* allow for such general transformation from black-box to white-box. A natural candidate is when the search problem is defined over graphs, and we are looking for a graph property (i.e., the decision of $\mathcal{S}$ whether to accept or not depends solely on the presented subgraph and not on the names of the vertices). Can we prove a transformation in this case? Can we show an impossibility?

## References

[1] R. Impagliazzo and M. Naor, "Decision trees and downward closures," in *3rd Annual Structure in Complexity Theory Conference*, 1988, pp. 29–38.

[2] S. Buss, "Introduction to NP functions and local search," 2009, slides: http://www.math.ucsd.edu/ sbuss/ResearchWeb/Prague2009/talkslides1.pdf. Accessed: 2017-04-01.

[3] P. W. Goldberg and C. H. Papadimitriou, "Towards a unified complexity theory of total functions," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 24, p. 56, 2017.

[4] N. Megiddo and C. H. Papadimitriou, "On total functions, existence theorems and computational complexity," *Theor. Comput. Sci.*, vol. 81, no. 2, pp. 317–324, 1991.

[5] L. Lovász, M. Naor, I. Newman, and A. Wigderson, "Search problems in the decision tree model," *SIAM J. Discrete Math.*, vol. 8, no. 1, pp. 119–132, 1995.

[6] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, 2004.

[7] S. Goldwasser and Y. T. Kalai, "On the (in)security of the Fiat-Shamir paradigm," in *FOCS*, 2003, pp. 102–113.

[8] P. Beame, S. A. Cook, J. Edmonds, R. Impagliazzo, and T. Pitassi, "The relative complexity of NP search problems," *J. Comput. Syst. Sci.*, vol. 57, no. 1, pp. 3–19, 1998.

[9] P. Hubácek and E. Yogev, "Hardness of continuous local search: Query complexity and cryptographic lower bounds," in *SODA*, 2017, pp. 1352–1371.

[10] C. Peikert and B. Waters, "Lossy trapdoor functions and their applications," *SIAM J. Comput.*, vol. 40, no. 6, pp. 1803–1844, 2011.

[11] P. Erdös and R. Rado, "Intersection theorems for systems of sets," *Journal of London Mathematical Society*, vol. 35, pp. 85–90, 1960.

[12] I. Komargodski, M. Naor, and E. Yogev, "White-box vs. black-box complexity of search problems: Ramsey and graph property testing," *ECCC*, vol. 24, p. 15, 2017.

[13] J. Krajícek, "On the weak pigeonhole principle," *Fundamenta Mathematicae*, vol. 170, pp. 123–140, 2001.

[14] C. H. Papadimitriou, "On the complexity of the parity argument and other inefficient proofs of existence," *J. Comput. Syst. Sci.*, vol. 48, no. 3, pp. 498–532, 1994.

[15] P. Hubácek, M. Naor, and E. Yogev, "The journey from NP to TFNP hardness," *ECCC*, vol. 23, p. 199, 2016, to appear in ITCS 2017.

[16] P. Ananth, A. Jain, M. Naor, A. Sahai, and E. Yogev, "Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption," in *CRYPTO*, 2016, pp. 491–520.

[17] S. Hada, "Zero-knowledge and code obfuscation," in *ASIACRYPT*, 2000, pp. 443–457.

[18] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang, "On the (im)possibility of obfuscating programs," *J. ACM*, vol. 59, no. 2, p. 6, 2012.

[19] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits," in *FOCS*, 2013.

[13] We thank Aviad Rubinstein for referring us to [41].

[20] N. Bitansky, O. Paneth, and A. Rosen, "On the cryptographic hardness of finding a Nash equilibrium," in *FOCS*, 2015, pp. 1480–1498.

[21] S. Garg, O. Pandey, and A. Srinivasan, "Revisiting the cryptographic hardness of finding a Nash equilibrium," in *CRYPTO*, 2016, pp. 579–604.

[22] I. Komargodski and G. Segev, "From Minicrypt to Obfustopia via private-key functional encryption," in *EUROCRYPT*, 2017, pp. 122–151.

[23] R. L. Graham, B. L. Rothschild, and J. H. Spencer, *Ramsey Theory*, 2nd ed. John Wiley, 1990.

[24] P. Erdös, "Some remarks on the theory of graphs," *Bull. Amer. Math. Soc.*, vol. 53, no. 4, pp. 292–294, 04 1947.

[25] N. Alon and J. Spencer, *The Probabilistic Method*, 3rd ed. John Wiley, 2008.

[26] M. Naor, "Constructing ramsey graphs from small probability spaces," *IBM Research Report RJ 8810*, 1992, available at: http://www.wisdom.weizmann.ac.il/ naor/PAPERS/ramsey.ps. Accessed: 2017-08-01.

[27] J. Naor and M. Naor, "Small-bias probability spaces: Efficient constructions and applications," *SIAM J. Comput.*, vol. 22, no. 4, pp. 838–856, 1993.

[28] B. Chor and O. Goldreich, "Unbiased bits from sources of weak randomness and probabilistic communication complexity," *SIAM J. Comput.*, vol. 17, no. 2, pp. 230–261, 1988.

[29] D. R. Simon, "Finding collisions on a one-way street: Can secure hash functions be based on general assumptions?" in *EUROCRYPT*, 1998, pp. 334–345.

[30] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev, "More constructions of lossy and correlation-secure trapdoor functions," *J. Cryptology*, vol. 26, no. 1, pp. 39–74, 2013.

[31] E. Kiltz, A. O'Neill, and A. D. Smith, "Instantiability of RSA-OAEP under chosen-plaintext attack," in *CRYPTO*. Springer, 2010, pp. 295–313.

[32] B. Hemenway and R. Ostrovsky, "Extended-DDH and lossy trapdoor functions," in *PKC*, 2012, pp. 627–643.

[33] P. Pudlák, "Ramsey's theorem in bounded arithmetic," in *Computer Science Logic, 4th Workshop, CSL*, 1990, pp. 308–317.

[34] E. Jerábek, "Approximate counting by hashing in bounded arithmetic," *J. Symb. Log.*, vol. 74, no. 3, pp. 829–860, 2009.

[35] A. Joux, "Multicollisions in iterated hash functions. application to cascaded constructions," in *CRYPTO*, vol. 3152, 2004, pp. 306–316.

[36] D. Conlon, "A new upper bound for the bipartite ramsey problem," *Journal of Graph Theory*, vol. 58, no. 4, pp. 351–356, 2008.

[37] I. Komargodski, M. Naor, and E. Yogev, "Collision resistant hashing for paranoids: Dealing with multiple collisions," *IACR Cryptology ePrint Archive*, p. 486, 2017.

[38] N. Bitansky, Y. T. Kalai, and O. Paneth, "Multi-collision resistance: A paradigm for keyless hash functions," *IACR Cryptology ePrint Archive*, p. 488, 2017.

[39] I. Berman, A. Degwekar, R. D. Rothblum, and P. N. Vasudevan, "Multi collision resistant hash functions and their applications," *IACR Cryptology ePrint Archive*, p. 489, 2017.

[40] O. Goldreich, "Introduction to testing graph properties," in *Studies in Complexity and Cryptography*, 2011, vol. 6650, pp. 470–506.

[41] P. W. Goldberg and A. Roth, "Bounds for the query complexity of approximate equilibria," *ACM Trans. Economics and Comput.*, vol. 4, no. 4, pp. 24:1–24:25, 2016.