

Witness Finding in the Black-Box Setting

Akinori Kawachi*, Benjamin Rossman*, and Osamu Watanabe*

*Department of Mathematical and Computing Sciences,
Tokyo Institute of Technology, Tokyo 152-8552, Japan
{kawachi, rossman, watanabe}@is.titech.ac.jp

Abstract

We propose an abstract framework for studying search-to-decision reductions for NP. Specifically, we study the following *witness finding problem*: for a hidden nonempty set $W \subseteq \{0,1\}^n$, the goal is to output a witness in W with constant probability by making randomized queries of the form “is $Q \cap W$ nonempty?” where $Q \subseteq \{0,1\}^n$. Algorithms for the witness finding problem can be seen as a general form of search-to-decision reductions for NP. This framework is general enough to express the average-case search-to-decision reduction of Ben-David et al., as well as the Goldreich-Levin algorithm from cryptography.

We show that the witness finding problem requires $\Omega(n^2)$ non-adaptive queries with the error-free oracle, matching the upper bound of Ben-David et al. We also give a new witness finding algorithm that achieves an improved error tolerance of $O(1/n)$ with $O(n^2)$ non-adaptive queries. Further, we investigate a list-decoding version of the witness finding problem, where a witness is unique, i.e., $|W| = 1$, and answers from the oracle may contain some errors. For this setting, it has been known that an improved version of the Goldreich-Levin algorithm with $O(n/\varepsilon^2)$ non-adaptive queries and $O(1/\varepsilon^2)$ list size solves the problem with any $(1/2 - \varepsilon)$ -error bounded oracle. We show that this query complexity is optimal up to a constant factor (if we want to keep the list size polynomially bounded) even if queries are adaptive.

1 Introduction

We propose an abstract framework for studying the relationship between the search and decision versions of NP problems. As a generalization of search-to-decision reductions, we study the following *witness finding problem*: for a hidden nonempty subset W of $\{0,1\}^n$, the goal is to produce an element in W by asking NP-type queries of the form “is $Q \cap W$ nonempty?” where $Q \subseteq \{0,1\}^n$. Algorithms solving the witness finding problem can be seen as generic search-to-decision reductions that apply to any NP problem. This witness finding problem is also relevant to cryptography. The algorithms of Ben-David, Chor, Goldreich, and Luby (hereafter, Ben-David et al.) [BCGL92] and Goldreich and Levin [GL89] can be seen as witness finding algorithms. We discuss the query complexity of these algorithms. We also consider the situation where queries may be answered incorrectly, and we study witness finding algorithms from the perspective of error tolerance as well.

The complexity class NP is characterized as the class of sets with a polynomial-size and polynomial-time checkable witness system. That is, for any set $L \subseteq \{0,1\}^*$, it is in NP if and only if it is characterized by

$$L = \{ x \mid \exists w \text{ such that } |w| \leq q(|x|) \text{ and } R(x, w) \}$$

with some polynomial q and some polynomial-time computable predicate R . For any set $L \in \text{NP}$, a pair (q, R) characterizing L as above is called a *witness system*, and for any $x \in L$, a string w satisfying $|w| \leq q(|x|) \wedge R(x, w)$ is called a *witness* (for $x \in L$). The *decision problem* for L is the task of deciding, for a given $x \in \{0,1\}^n$, whether there exists some witness w for $x \in L$. The *search problem* for L is the task of producing a witness for a given $x \in L$.

The relationship between decision and search problems has been investigated in several contexts. In order to be specific, let us consider the 3SAT problem and the standard witness system that uses, for any 3CNF formula φ , a satisfying assignment as a witness for $\varphi \in \text{3SAT}$. Here the search problem is the task of finding a satisfying assignment for a given $\varphi \in \text{3SAT}$. A question that has been often asked is how to compute *one of* the satisfying assignments of φ by asking queries to 3SAT. It is easy to obtain a satisfying assignment for $\varphi \in \text{3SAT}$ (e.g. the lexicographically first one) deterministically by asking queries to 3SAT *adaptively*. That is, the search problem for 3SAT is P^{NP} -computable. On the other hand, Ben-David et al. [BCGL92] used the “isolation technique” to give a randomized algorithm that solves this NP-type search problem in polynomial-time by asking queries to 3SAT *nonadaptively*; that is, the witness finding problem for 3SAT is $\text{ZPP}_{\parallel}^{\text{NP}}$ -computable. In this paper, we focus on the query complexity of such computations. Consider any $\varphi \in \text{3SAT}$ with n variables. Then a witness, i.e. satisfying assignment, can be expressed as a binary string of length n . For computing such a witness, the above mentioned P^{NP} -algorithm can be implemented with n adaptive queries, whereas $O(n^2)$ nonadaptive queries are needed in the $\text{ZPP}_{\parallel}^{\text{NP}}$ -algorithm. We investigate in this paper whether this difference in query complexity is inherent between the adaptive and nonadaptive ways of asking queries. As an abstract framework for discussing this type of query complexity, we introduce the witness finding problem, a problem of searching for a witness in an unknown set $W \subseteq \{0,1\}^n$ (as a generic NP-type search problem) by using queries of the form “is $Q \cap W$ nonempty?” for some $Q \subseteq \{0,1\}^n$ (as instances of a generic NP-type decision problem). Here the set W is called a *witness set* and every $w \in W$ is regarded as a *witness*; on the other hand, the set Q is regarded as a specification of a *query*. A *randomized algorithm* for solving the witness finding problem in this abstract setting is a pair $(\mathcal{Q}, \mathcal{F})$, where \mathcal{Q} is a procedure to generate queries Q_1, Q_2, \dots, Q_m (either adaptively or nonadaptively) and $\mathcal{F} : \{0,1\}^m \rightarrow \{0,1\}^n$ is a procedure to give a witness in W based on answers to these queries. Let $A_W : 2^{\{0,1\}^n} \rightarrow \{0,1\}$ denote the oracle answering decision queries. An algorithm is successful if it produces a witness in W with constant probability for every witness set W . That is, W (or more precisely, the oracle A_W) is considered as a blackbox. By contrast, an algorithm for solving a concrete NP-type search problem only needs to succeed on a specific class of witness sets. However, even in our abstract setting, both the standard P^{NP} -algorithm and the $\text{ZPP}_{\parallel}^{\text{NP}}$ -algorithm of Ben-David et al. succeed in solving the witness finding problem for every witness set W . In particular, the P^{NP} -algorithm uses only n queries whereas the $\text{ZPP}_{\parallel}^{\text{NP}}$ -algorithm uses $O(n^2)$ queries. On the other hand, we prove that any

randomized nonadaptive query algorithm that solves the witness finding problem for every $W \subseteq \{0,1\}^n$ (with probability $\Omega(1)$) needs to ask $\Omega(n^2)$ queries. That is, in this abstract framework, we show that the nonadaptive query complexity for the witness finding problem is $\Theta(n^2)$ and illustrate the difference between adaptive and nonadaptive query complexity.

Valiant and Vazirani [VV86] gave a procedure, known as the “isolation technique”, to modify a given NP instance to some other NP instance that reduces the number of witnesses to exactly one with a certain probability. To be specific, consider 3SAT. The isolation technique of Valiant-Vazirani is a randomized reduction f_{iso} from 3SAT to L_{iso} with the following property: each instance $\phi \in 3\text{SAT}$ (that may have more than one satisfying assignment) is reduced to $\psi = f_{\text{iso}}(\phi)$ such that, with a certain probability, $\psi \in L_{\text{iso}}$ and this is witnessed by exactly one satisfying assignment of ψ . Recently, Dell, Kabanets, van Melkebeek, and Watanabe [DKvMW12] showed the optimality of the isolation technique of Valiant-Vazirani. They showed that the “isolation probability”, i.e. the probability of a unique satisfying assignment, is at most $O(1/n)$ by any randomized reduction under a certain blackbox computation model. While the isolation technique of Valiant-Vazirani is a many-one reduction, the witness finding algorithm of Ben-David et al. [BCGL92] can be regarded as a more general truth-table reduction, and by this generalized reduction, we can achieve $1 - o(1)$ isolation probability. On the other hand, in Section 3 we show a limitation of this type of isolation by proving under a similar blackbox model that any truth-table reduction type isolation needs $\Omega(n^2)$ queries. This can be viewed as a multi-query version of the result of Dell et al. [DKvMW12], which discussed the success probability of black-box isolation with a single query. In fact, it is easy to see that the isolation probability upper bound of [DKvMW12] follows from our $\Omega(n^2)$ query lower bound.

We also consider the situation where queries may be answered incorrectly. In fact, the motivation of Ben-David et al. was to solve a given witness finding problem *on average* by using a polynomial-time algorithm \mathcal{A} that solves the corresponding NP-type decision problem with high probability on random instances. The algorithm \mathcal{A} can be seen as an erroneous oracle; on the other hand, the algorithm of Ben-David et al. is tolerant against a small fraction of errors and solves the witness finding problem with high probability by using \mathcal{A} as an oracle. We investigate the question of whether the algorithm of Ben-David et al. is optimal also in terms of its error tolerance. For this, we extend our abstract framework to include a notion of “ ε -error tolerance.” The algorithm of Ben-David et al. is $O(1/n^2)$ -error tolerant in our framework. In Section 3 we show how to improve this error tolerance parameter by presenting a randomized nonadaptive algorithm that is $O(1/n)$ -error tolerant while still making only $O(n^2)$ queries.

Finally, in Section 4 we consider the setting where the witness is unique, that is, the witness set consists of one element. This situation is typical for solving decoding problems. For example, for analyzing the hardcore property of one-way functions, Goldreich and Levin gave a randomized polynomial-time algorithm that solves the unique witness finding problem by using some NP oracle; see, e.g., [Gol01]. For any singleton witness set $W = \{w\}$, the Goldreich-Levin algorithm gives a list of candidates for this witness w by making queries to an erroneous oracle whose error probability is bounded by $1/2 - \varepsilon$. This may be regarded as a “list decoding algorithm” where the unique witness w is a “message” and the erroneous oracle is a “corrupted codeword”. We again would like to discuss the limitations of such algorithms. For this, we relax our notion of an abstract witness finding algorithm to allow the output to be a list of candidate witnesses. It is easy to see that, for any $\varepsilon = n^{-O(1)}$, the Goldreich-Levin algorithm achieves $(1/2 - \varepsilon)$ -error tolerance with $O(n^2/\varepsilon^2)$ query complexity and $O(n/\varepsilon^2)$ list size; furthermore, this can be improved to one with $O(n/\varepsilon^2)$ query complexity and $O(1/\varepsilon^2)$ list size. We show that the query complexity of the improved version is close to the optimal in the following sense: There exists some small $c_1 > 0$ such that for any $\varepsilon = n^{-O(1)}$, no randomized query algorithm (even an adaptive one) exists with $c_1 n/\varepsilon^2$ query complexity that has $(1/2 - \varepsilon)$ -error tolerance and polynomially bounded list size. This can be also interpreted as a lower bound of the query complexity for list decoding algorithms.

2 Preliminaries

We use standard notions and notations in computational complexity theory. Throughout this paper, we assume strings are encoded in $\{0, 1\}^*$, and we use $\Omega_n = \{0, 1\}^n$ (or more simply Ω when n is clear by the context) to denote the universe of witnesses for a given length parameter n . For any set X and any distribution \mathcal{D} on X , by “ $\Pr_{x:\mathcal{D}}[\Phi(x)]$ ” we mean the probability that $\Phi(x)$ holds when x is chosen under the distribution \mathcal{D} . When \mathcal{D} is the uniform distribution on X , we simply write it as “ $\Pr_{x:X}[\Phi(x)]$.”

Definition 1 A witness set is a nonempty subset of Ω_n denoted by W , and each element of W is a witness. A query is a set $Q \subseteq \Omega_n$ interpreted as the question “is $Q \cap W$ nonempty?” Let $A_W(Q) \in \{0, 1\}$ denote the answer from the (error-free) oracle to the query Q ; that is, $A_W(Q) = 1$ if and only if $Q \cap W \neq \emptyset$.

The *witness finding problem* is the problem of obtaining any one of the witnesses in W by asking queries and using oracle answers to those queries. Here we define the following abstract notion of “algorithm” for this task.

Definition 2 A randomized witness finding query algorithm (hereafter, we omit “randomized” and “query” for simplicity) is a pair $(\mathcal{Q}, \mathcal{F})$ of randomized algorithms where, for every witness length n and random seed $s \in \{0, 1\}^{r(n)}$, \mathcal{Q} produces a sequence of queries $Q_1, \dots, Q_{m(n)} \subseteq \Omega_n$ (where Q_i may depend on the oracle answers to queries Q_1, \dots, Q_{i-1}) and \mathcal{F} outputs an element in Ω_n based on the answers to queries $Q_1, \dots, Q_{m(n)}$. For a witness set W , the algorithm succeeds w.r.t. to W whenever \mathcal{F} correctly outputs an element of W . The success probability of the algorithm is the probability $\Pr_{s:\{0,1\}^{r(n)}}[\mathcal{F}(s, \tilde{A}_W(\mathcal{Q})) \text{ succeeds}]$ for the worst witness set. Parameters $r(n)$ and $m(n)$ are called the seed length and query complexity.

We may sometimes use the term “success probability” in a more general way meaning the probability $\Pr[\mathcal{F}(s, \tilde{A}_W(\mathcal{Q})) \text{ succeeds}]$ under some distribution defined in each context. An algorithm is said to be *nonadaptive* if the distribution of queries Q_1, \dots, Q_m does not depend on the answers to these queries; otherwise, the algorithm is *adaptive*.

When n is fixed, we write simply r and m . We write $A_W(\mathcal{Q})$ as a shorthand for the sequence $A_W(Q_1), \dots, A_W(Q_m)$ of answers to queries Q_1, \dots, Q_m issued by \mathcal{Q} , and we write $\mathcal{F}(A_W(\mathcal{Q}))$ for the output of \mathcal{F} . This notation suppresses the random seed s ; to make this dependence explicit, we write $\mathcal{F}(s, A_W(\mathcal{Q}))$.

This abstract definition of witness finding algorithms is suitable for our information-theoretic lower bounds. Of course, we should consider more concrete algorithms for our upper bounds, including an appropriate definition of polynomial-time computability.

Definition 3 A witness finding algorithm $(\mathcal{Q}, \mathcal{F})$ is polynomial-time if the seed length $r(n)$ is bounded by a polynomial in n , and \mathcal{Q} and \mathcal{F} are polynomial-time algorithms taking $(1^n, s)$ as input where $s \in \{0, 1\}^{r(n)}$; \mathcal{Q} outputs a sequence of Boolean circuits C_1, \dots, C_m such that C_i computes a function $\{0, 1\}^n \times \{0, 1\}^{i-1} \rightarrow \{0, 1\}$ with $Q_i = \{x \mid C_i(x, A_W(Q_1), \dots, A_W(Q_{i-1})) = 1\}$, and \mathcal{F} outputs a Boolean circuit computing $\mathcal{F}(A_W(\mathcal{Q}))$ as a function $\{0, 1\}^m \rightarrow \{0, 1\}^n$.

Remark 1 In many situations, a witness set W is determined by an “input” x to a witness finding algorithm. In the case of 3SAT, for example, x is a satisfiable 3CNF formula and W_x is the set of its satisfying assignments. However, it is not essential to consider such inputs in our abstract framework, so we omit specifying inputs to witness finding algorithms.

Remark 2 In actual witness finding algorithms for NP sets, we may relax some of the above conditions. For example, we may allow queries of the form “ $|W \cap Q| \geq k$?” for some k that is polynomially bounded by n . The following discussion does not change much if such generalized queries are allowed. Another relaxation is to allow an algorithm to output a list of polynomially many candidate witnesses; the algorithm succeeds if W contains any element in this list. We call an algorithm of this type a witness-list finding algorithm, and for any witness-list

finding algorithm, we write $\ell(n)$ to denote its list size, a function bounding the number of elements in the list that the algorithm produces for any witness set in Ω_n . For the search version of an NP problem, this relaxation does not make any difference since one can check the correctness of a witness candidate in polynomial-time.

Definition 4 For a witness finding algorithm $(\mathcal{Q}, \mathcal{F})$, we denote by Dom the set of queries produced by \mathcal{Q} for all possible random seeds and oracle answers. (For a polynomial-time algorithm, note that $|Dom|$ is bounded by $2^{\text{poly}(n)}$.) For a witness set W , a function $\tilde{A}_W : Dom \rightarrow \{0, 1\}$ is ε -error bounded if its error probability is at most ε when queries are chosen uniformly from Dom ; that is,

$$\Pr_{Q:Dom} [\tilde{A}_W(Q) \neq A_W(Q)] < \varepsilon$$

holds. For an algorithm $(\mathcal{Q}, \mathcal{F})$ with random seed length r , its success probability with ε -error bounded oracle is the probability $\Pr_{s:\{0,1\}^r}[\mathcal{F}(s, \tilde{A}_W(\mathcal{Q})) \text{ succeeds}]$ with the worst ε -error bounded oracle for the worst witness set. An algorithm $(\mathcal{Q}, \mathcal{F})$ is ε -error tolerant if it has $\Omega(1)$ success probability.

Remark 3 For measuring the oracle's error probability, we assume a certain distribution \mathcal{D} over Dom instead of the distribution of queries made by the witness finding algorithm. This is because the error model is usually given independently from algorithms. In fact, in the context of using the Goldreich-Levin algorithm, the uniform distribution is used to measure oracle's error probability that is different from the distribution of queries of the algorithm. Throughout this paper, we consider the case where \mathcal{D} is the uniform distribution over Dom , and we omit specifying it.

3 Witness Finding for Isolation

In this section we consider nonadaptive algorithms for the general witness finding problem. As mentioned in Section 1, these algorithms can be regarded as truth-table type witness isolation reductions.

We first state the algorithm of Ben-David et al. in our framework. Here we consider the error-free oracle A_W (we will consider error tolerance later in this section).

Proposition 1 There is a polynomial-time nonadaptive witness finding algorithm $(\mathcal{Q}_{BD}, \mathcal{F}_{BD})$ with $\Omega(1)$ success probability and $O(n^2)$ query complexity.

We first show that $(\mathcal{Q}_{BD}, \mathcal{F}_{BD})$ is optimal in terms of query complexity.

Theorem 2 There is no nonadaptive witness finding algorithm with $\Omega(1)$ success probability and $o(n^2)$ query complexity.

Proof. Consider any nonadaptive witness finding algorithm $(\mathcal{Q}, \mathcal{F})$ which makes $m = o(n^2)$ queries. We will show that there exists a witness set W such that $\Pr_{s:\{0,1\}^r}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W] = o(1)$. For our analysis, we use Yao's principle [Yao77]. That is, we have

$$\begin{aligned} \min_W \Pr_{s:\{0,1\}^r}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W] &\leq \max_{\rho} \min_W \Pr_{s:\rho}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W] \\ &= \min_W \max_{s \in \{0,1\}^r} \Pr_{W:\mathcal{W}}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W] \leq \max_{s \in \{0,1\}^r} \Pr_{W:\mathcal{W}_*}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W], \end{aligned}$$

where W is any witness set, ρ is any distribution on $\{0, 1\}^r$, \mathcal{W} is any distribution on witness sets (i.e., on nonempty subsets of Ω), and \mathcal{W}_* is a particular distribution on witness sets that we define below. Note that the lefthand term of the first inequality is what we would like to estimate. To prove the theorem, we need to show that $\Pr_{W:\mathcal{W}_*}[\mathcal{F}(s, A_W(\mathcal{Q})) \in W] = o(1)$ for any fixed $s \in \{0, 1\}^r$.

Our distribution \mathcal{W}_* on witness sets is defined by the following procedure to generate a witness set W : first we choose K uniformly at random from $[n]$, then we define W by including each $w \in \Omega$ in W independently with probability 2^{-K} .¹ A small technicality is that W may possibly be the empty set; however, since $W = \emptyset$ occurs with probability $o(1)$ for $W : \mathcal{W}_*$, we can ignore this degenerate case.² Note that, for $k \in [n]$, the expected size of W conditioned on $K = k$ is 2^{n-k} . Below we keep using K and W to denote these random variables.

We now fix an arbitrary seed $s \in \{0, 1\}^r$. Let Q_1, \dots, Q_m be the queries given by $\mathcal{Q}(s)$, and let f be the function from $\{0, 1\}^m$ to Ω given by $\mathcal{F}(s, \dots)$ (which attempts to output a witness in W given the answers to queries Q_1, \dots, Q_m). For $i \in [m]$, let $A_i = A_W(Q_i)$. That is, $A_i \in \{0, 1\}$ is the indicator random variable for the event that $Q_i \cap W \neq \emptyset$.

Below by $\Pr[\dots]$, we mean $\Pr_{W: \mathcal{W}_*}[\dots]$. Then $\Pr[f(A_1, \dots, A_m) \in W]$ is the success probability of the algorithm $(\mathcal{Q}, \mathcal{F})$ with respect to the distribution \mathcal{W}_* when running with a fixed seed s . Our goal is to show that $\Pr[f(A_1, \dots, A_m) \in W] = o(1)$. Toward that end, we view $f(A_1, \dots, A_m)$ as a random variable over Ω and estimate its entropy. Below we use the standard notations for discussing entropy.³

Claim 1 $H(f(A_1, \dots, A_m)) \leq \log n + O(m/n)$.

Proof. We have

$$\begin{aligned} H(f(A_1, \dots, A_m)) &\leq H(A_1, \dots, A_m) \leq H(A_1, \dots, A_m, K) = H(K) + H(A_1, \dots, A_m | K) \\ &\leq \log n + \sum_{i=1}^m H(A_i | K) = \log n + \frac{1}{n} \sum_{i=1}^m \sum_{k=1}^n H(A_i | K = k). \end{aligned}$$

To finish the proof of the claim, we show that $\sum_{k=1}^n H(A_i | K = k) = O(1)$ for all $i \in [m]$ (where $O(1)$ is some universal constant). Fix arbitrary $i \in [m]$. For $k \in [n]$, let $p_k = \Pr[A_i = 1 | K = k]$ ($= \Pr[Q_i \cap W \neq \emptyset | K = k]$). We have $H(A_i | K = k) = H(p_k)$ where $H : [0, 1] \rightarrow [0, 1]$ is the binary entropy function $H(p) = -p \log p - (1-p) \log(1-p)$. Note the inequality

$$\min(p, 1-p) \leq q \leq 1/2 \implies H(p) \leq -2q \log q.$$

We use this inequality to bound $H(p_k)$.

Let $\lambda = \log |Q_i|$. We consider three cases depending on $k \in [n]$.

- Case $k \leq \lambda - 1$: We have $1 - p_k = (1 - 2^{-k})^{|Q_i|} \leq e^{-2^{\lambda-k}} (< 1/2)$. Hence, we have $H(p_k) \leq -2 \ln(e^{-2^{\lambda-k}}) e^{-2^{\lambda-k}} = 2^{\lambda-k+1} - \log(e) 2^{\lambda-k} = 2^{-\Omega(2^{\lambda-k})}$.
- Case $\lambda - 1 < k < \lambda + 1$ (at most two k 's): We have $H(p_k) \leq 1$.
- Case $k \geq \lambda + 1$: We have $p_k \leq |Q_k| 2^{-k} = 2^{\lambda-k}$. Hence, $H(p_k) \leq (k - \lambda) 2^{\lambda-k+1}$.

Now it follows that

$$\begin{aligned} \sum_{k=1}^n H(A_i | K = k) &= \sum_{k=1}^n H(p_k) \leq \sum_{k=1}^{\lfloor \lambda-1 \rfloor} 2^{-\Omega(2^{\lambda-k})} + 2 + \sum_{k=\lceil \lambda+1 \rceil}^n (k - \lambda) 2^{\lambda-k+1} \\ &< O\left(\sum_{j=1}^{\infty} 2^{-\Omega(2^j)}\right) + 2 + O\left(\sum_{j=1}^{\infty} j 2^{-j}\right) = O(1). \end{aligned}$$

¹The same distribution was considered in [DKvMW12] to show an $O(1/n)$ upper bound on the *success probability* of witness finding algorithms.

²Precisely speaking, we should consider a distribution \mathcal{W}'_* conditioned that the empty set is never obtained. But the difference, which is negligible, is ignored here.

³For discrete random variables X and Y , $H(X) = \sum_x -\Pr[X = x] \log \Pr[X = x]$ and $H(X|Y) = \sum_y \Pr[Y = y] H(X|Y = y)$, where $H(X|Y = y) = \sum_x -\Pr[X = x|Y = y] \log \Pr[X = x|Y = y]$. For a sequence of random variables X_1, \dots, X_j , $H(X_1, \dots, X_j)$ is the entropy of the joint distribution (X_1, \dots, X_j) .

□ (Claim 1)

Using this entropy bound, we now show that $\Pr[f(A_1, \dots, A_m) \in W] = o(1)$. Fix an arbitrary constant $\epsilon > 0$. For $w \in \Omega$, let $p(w)$ denote the probability that $f(A_1, \dots, A_m)$ takes value w . Let $U = \{w \mid p(w) \geq 2^{-\epsilon n}\}$ and note that $|U| \leq 2^{\epsilon n}$. For all $w \notin U$, we have $-\log p(w) > \epsilon n$ and thus (by Claim 1)

$$\begin{aligned} & \Pr[f(A_1, \dots, A_m) \notin U] \\ &= \sum_{w \notin U} p(w) \leq \sum_{w \notin U} \frac{-p(w) \log p(w)}{\epsilon n} \leq \frac{H(f(A_1, \dots, A_m))}{\epsilon n} = o(1). \end{aligned}$$

Next we have

$$\begin{aligned} \Pr[U \cap W \neq \emptyset] &\leq \Pr[K \leq 2\epsilon n] + \Pr[U \cap W \neq \emptyset \mid K > 2\epsilon n] \\ &\leq 2\epsilon + \sum_{u \in U} \Pr[u \in W \mid K > 2\epsilon n] \\ &< 2\epsilon + |U|2^{-2\epsilon n} \leq 2\epsilon + 2^{-\epsilon n} = 2\epsilon + o(1). \end{aligned}$$

Combining these inequalities, we have

$$\begin{aligned} \Pr[f(A_1, \dots, A_m) \in W] &\leq \Pr[f(A_1, \dots, A_m) \notin U \vee U \cap W \neq \emptyset] \\ &\leq \Pr[f(A_1, \dots, A_m) \notin U] + \Pr[U \cap W \neq \emptyset] \leq 2\epsilon + o(1). \end{aligned}$$

Since ϵ is arbitrarily small, we have $\Pr[f(A_1, \dots, A_m) \in W] = o(1)$. □

Next we consider error tolerance. Note that the algorithm $(\mathcal{Q}_{\text{BD}}, \mathcal{F}_{\text{BD}})$ is $O(1/n^2)$ -error tolerant since it makes only $O(n^2)$ queries, which are uniformly distributed over the hash functions. Here we present a new polynomial-time algorithm based on some standard error-correcting code that has a better $O(1/n)$ -error tolerance while still making only $O(n^2)$ queries. (Note that in our framework we do not assume any error-free checking procedure for the obtained witnesses, as in [BT06]. One interesting point in the following algorithm is that we can indeed implement the checking procedure with erroneous oracles.)

Theorem 3 *There is a polynomial-time, nonadaptive, and $O(1/n)$ -error tolerant witness finding algorithm with $O(n^2)$ query complexity.*

Proof. We first prepare some polynomial-time encodable and decodable code with some specific property suitable for our usage. Below for any binary strings x and y , we denote by $|x|$ the Hamming weight of x and by $\text{dist}(x, y)$ the Hamming distance between x and y .

Lemma 4 *There exists a polynomial-time encodable and decodable code with the following property for some constants $c > 1$ and $\delta > 0$: For any n , let $C \subseteq \{0, 1\}^{2cn}$ denote the set of codewords encoding messages in $\{0, 1\}^n$. Then we have (i) $|y| = cn$ for all $y \in C$, and for all distinct $y, y' \in C$, we have (ii) $\text{dist}(y, y') \geq 2\delta cn$, and (iii) $|y \vee y'| = cn + \frac{1}{2} \text{dist}(y, y') \geq (1 + \delta)cn$, where $y \vee y'$ is the bit-wise OR of y and y' . Below we use C also to denote a function mapping any message in $\{0, 1\}^n$ to the corresponding codeword in $\{0, 1\}^{2cn}$.*

Proof. First consider any polynomial-time encodable and decodable binary code of some constant rate $1/c$ and constant relative minimum distance δ . That is, for each n , every message in $y \in \{0, 1\}^n$ is encoded by some codeword $\hat{y} \in \{0, 1\}^{cn}$, and we have $\text{dist}(\hat{y}, \hat{y}') \geq \delta cn$ for any two distinct codewords $\hat{y}, \hat{y}' \in \{0, 1\}^{cn}$. For example, Justesen code satisfies all these requirements (see, e.g., [Rot06]). Now for any $x \in \{0, 1\}^n$, our $C(x)$ is defined by

$$C(x) = (\hat{y}_1, \dots, \hat{y}_{cn}, 1 - \hat{y}_1, \dots, 1 - \hat{y}_{cn}),$$

where $\hat{y} = \hat{C}(x)$. It is easy to see that C satisfies properties (i), (ii) and (iii) of the lemma. □ (Lemma 4)

We present our algorithm $(\mathcal{Q}, \mathcal{F})$ as a modification of the algorithm of Ben-David et al. [BCGL92] by using this code. For this, we recall their algorithm first. The key tool of their algorithm is the isolation technique of Valiant and Vazirani [VV86]. In the abstract setting, their technique is a polynomial-time algorithm \mathcal{Q}_{VV} that generates/recognizes a random subset $\mathcal{Q}_{\text{VV}}(s)$ of $\Omega (= \{0, 1\}^n)$ from a random seed $s \in \{0, 1\}^{r_{\text{iso}}}$ for some polynomial r_{iso} (again we write simply r_{iso} for $r_{\text{iso}}(n)$); the important property here is that for any nonempty $W \subseteq \Omega$, we have $|\mathcal{Q}_{\text{VV}}(s) \cap W| = 1$ with probability $\geq \frac{1}{4n}$ [VV86]. For $\mathcal{Q}_{\text{VV}}(s)$ that *achieves isolation* (i.e., $|\mathcal{Q}_{\text{VV}}(s) \cap W| = 1$), we can ask each bit of the unique witness $w \in \mathcal{Q}_{\text{VV}}(s) \cap W$ nonadaptively. In our framework, this is implemented by asking queries $\{v \mid v \in \mathcal{Q}_{\text{VV}}(s) \wedge v_i = 1\}$ for all $i \in [n]$. The algorithm of Ben-David et al. asks a set of these n queries for $4n$ random seeds for s to achieve $\Omega(1)$ success probability. Clearly, in order for the algorithm to work, no error should occur for these $O(n^2)$ queries, and for this, we need an $O(1/n^2)$ -error bounded oracle.

In our algorithm, instead of asking for each bit of the isolated witness directly, we ask for bits of the codeword of the isolated witness. That is, for each $s \in \{0, 1\}^{r_{\text{iso}}}$, we consider queries

$$Q_{s,j} = \{v \mid v \in \mathcal{Q}_{\text{VV}}(s) \wedge C(v)_j = 1\}$$

for all $j \in [2cn]$. Again, to achieve $\Omega(1)$ success probability, the query algorithm \mathcal{Q} asks a set of these $2cn$ queries for $4n$ independent random seeds $s_1, \dots, s_{4n} \in \{0, 1\}^{r_{\text{iso}}}$. Thus, \mathcal{Q} is an algorithm that takes random seeds $s_1, \dots, s_{4n} \in \{0, 1\}^{r_{\text{iso}}}$, and asks queries $Q_{s_k,j}$ for all $k \in [4n]$ and $j \in [2cn]$. Hence, the algorithm makes $8cn^2$ nonadaptive queries. Let Dom denote the set $\{Q_{s,j} \mid s \in \{0, 1\}^{r_{\text{iso}}}, j \in [2cn]\}$ of all such queries.

For explaining the algorithm \mathcal{F} , we first see answers we can expect from an erroneous oracle. Let $\tilde{A}_W : 2^\Omega \rightarrow \{0, 1\}$ be any $\frac{\delta}{32n}$ -error bound oracle for some unknown witness set W . That is, we assume that

$$\Pr_{Q:\text{Dom}} [\tilde{A}_W(Q) \neq A_W(Q)] (= \Pr_{s:\{0,1\}^r, j:[2cn]} [\tilde{A}_W(Q_{s,j}) \neq A_W(Q_{s,j})]) \leq \frac{\delta}{32n}.$$

For $s \in \{0, 1\}^{r_{\text{iso}}}$, we define

$$\alpha(s) = \frac{1}{2cn} \sum_{j \in [2cn]} A_W(Q_{s,j}) \quad \text{and} \quad \tilde{\alpha}(s) = \frac{1}{2cn} \sum_{j \in [2cn]} \tilde{A}_W(Q_{s,j}).$$

Let $\text{Good}(s)$ be the event that $|\alpha(s) - \tilde{\alpha}(s)| < \delta/4$. By Markov's inequality we have

$$\begin{aligned} \Pr_s[\neg \text{Good}(s)] &= \Pr_s[\Pr_j[\tilde{A}_W(Q_{s,j}) \neq A_W(Q_{s,j})] \geq \delta/4] \\ &\leq \frac{\text{Es}[\Pr_j[\tilde{A}_W(Q_{s,j}) \neq A_W(Q_{s,j})]]}{\delta/4} \leq \frac{1}{8n}. \end{aligned} \tag{1}$$

Let $\text{Isolated}(s)$ be the event that $|\mathcal{Q}_{\text{VV}}(s) \cap W| = 1$.

Claim 2 *If $\text{Good}(s)$ holds, then $\text{Isolated}(s) \Leftrightarrow |\frac{1}{2} - \tilde{\alpha}(s)| < \frac{\delta}{4}$. Moreover, if $\text{Good}(s) \wedge \text{Isolated}(s)$ holds, then given $\tilde{A}_W(Q_{s,1}), \dots, \tilde{A}_W(Q_{s,2cn})$, the unique element in $\mathcal{Q}_{\text{VV}}(s) \cap W$ can be computed in polynomial-time.*

Proof. The claim follows from the properties of C . If $\text{Good}(s)$, then

$$\begin{aligned} |\mathcal{Q}_{\text{VV}}(s) \cap W| = 0 &\Rightarrow \alpha(s) = 0 &\Rightarrow \tilde{\alpha}(s) < \frac{\delta}{4} (< \frac{1}{2} - \frac{\delta}{4}), \\ |\mathcal{Q}_{\text{VV}}(s) \cap W| = 1 &\Rightarrow \alpha(s) = \frac{1}{2} &\Rightarrow \frac{1}{2} - \frac{\delta}{4} < \tilde{\alpha}(s) < \frac{1}{2} + \frac{\delta}{4}, \\ |\mathcal{Q}_{\text{VV}}(s) \cap W| \geq 2 &\Rightarrow \alpha(s) \geq \frac{1}{2} + \frac{\delta}{2} &\Rightarrow \tilde{\alpha}(s) > \frac{1}{2} + \frac{\delta}{4}. \end{aligned}$$

Thus, $\text{Good}(s)$ implies $\text{Isolated}(s) \Leftrightarrow |\frac{1}{2} - \tilde{\alpha}(s)| < \frac{\delta}{4}$. In the event that both $\text{Good}(s)$ and $\text{Isolated}(s)$ hold, we can decode $(\tilde{A}_W(Q_{s,1}), \dots, \tilde{A}_W(Q_{s,2cn})) \in \{0, 1\}^{2cn}$ in polynomial-time to recover the original message $w \in \{0, 1\}^n$; this w is then the unique element of $\mathcal{Q}_{\text{VV}}(s) \cap W$. \square (Claim 2)

Now the following description of algorithm \mathcal{F} is clear from this claim: For given answers from the oracle, compute $\tilde{\alpha}(s_i)$ for all $i \in [4n]$. If there exists $i \in [4n]$ such that $|\frac{1}{2} - \tilde{\alpha}(s_i)| < \frac{\delta}{4}$, then for the first such i , output the unique element of $\mathcal{Q}_{VV}(s_i) \cap W$ according to Claim 2 under the (possibly false) assumption that $\text{Good}(s_i)$ holds. (If $|\frac{1}{2} - \tilde{\alpha}(s_i)| \geq \frac{\delta}{4}$ for all $i \in [4n]$, the algorithm simply fails.)

To analyze its success probability, note that it successfully outputs a witness in W whenever $\bigwedge_{i \in [4n]} \text{Good}(s_i) \wedge \bigvee_{i \in [4n]} \text{Isolated}(s_i)$ holds. Using the bound (1) and the fact that $\Pr_{s: \{0,1\}^{r_{\text{iso}}}} [\text{Isolated}(s)] \geq \frac{1}{4n}$, we have

$$\begin{aligned} & \Pr_{s_1, \dots, s_{4n}} \left[\bigwedge_{i \in [4n]} \text{Good}(s_i) \wedge \bigvee_{i \in [4n]} \text{Isolated}(s_i) \right] \\ & \geq 1 - \Pr_{s_1, \dots, s_{4n}} \left[\bigvee_{i \in [4n]} \neg \text{Good}(s_i) \right] - \Pr_{s_1, \dots, s_{4n}} \left[\bigwedge_{i \in [4n]} \neg \text{Isolated}(s_i) \right] \\ & \geq 1 - 4n \Pr_s [\neg \text{Good}(s)] - \left(1 - \Pr_s [\neg \text{Isolated}(s)]\right)^{4n} \geq 1 - \frac{1}{2} - \frac{1}{e} > 0.13. \end{aligned}$$

Therefore, the algorithm succeeds with constant probability > 0.13 for any $\frac{\delta}{32n}$ -error bounded oracle. \square

4 Witness Finding for Decoding

We consider the case where a witness set is restricted to a singleton set, the situation typical for decoding problems. Throughout this section, we consider only singleton witness sets, and the conditions of, e.g., Definition 2 are modified for this restriction. Also, since our target witness set is singleton, we specify the target by a witness $w \in \Omega$ instead of a witness set $W = \{w\}$ in the following. Throughout this section, we use ε to denote any function on \mathbf{N} such that $0 < \varepsilon(n) < 1/2$ holds for any n .

As explained in Introduction, the original Goldreich-Levin algorithm is a $(1/2 - \varepsilon)$ -error tolerant witness-list finding algorithm with $O(n^2/\varepsilon^2)$ query complexity and $O(n/\varepsilon^2)$ list size, and this has been improved as follows⁴ (see for example [Gol01]).

Proposition 5 *There is a randomized polynomial-time, nonadaptive, and $(1/2 - \varepsilon)$ -error tolerant witness-list finding algorithm with $O(n/\varepsilon^2)$ query complexity and $O(1/\varepsilon^2)$ list size.*

Our second main result shows that the above query complexity is optimal. In fact, if both constant success probability and polynomially bounded list size are required, then we show that $m > c_1 n/\varepsilon^2$ queries are necessary while the above theorem shows that $m = \hat{c}_1 n/\varepsilon^2$ is sufficient for some constants $c_1 < \hat{c}_1$.

Theorem 6 *Consider any witness-list finding algorithm with seed length r , query complexity m , and list size ℓ . For any sufficiently small $\varepsilon > 0$, let α denote its success probability with $(1/2 - \varepsilon)$ -error bounded oracle. Then for some constants $c_1 > 0$, if $m \leq c_1 n/\varepsilon^2$, then we have $\alpha = O(\ell 2^{-\Omega(n)}/\varepsilon)$. (In the proof below, we use $c_1 = 1/640$.)*

Proof. Consider any witness-list finding algorithm $(\mathcal{Q}, \mathcal{F})$ with random seed length r , query complexity m , and list size ℓ . Also consider any $\varepsilon > 0$ for the advantage parameter. We may assume that $\varepsilon < 1/4$. As usual, fix sufficiently large n , and consider the problem of finding a given witness $w \in \Omega = \{0, 1\}^n$; from now on, r , m , ℓ , and ε are some numbers determined by n . In particular, let us fix the query number bound m to $m = c_1 n/\varepsilon^2$ for $c_1 = 1/640$.

Let Dom be the set of all possible queries made by \mathcal{Q} with any random seed and any oracle answers. We consider two cases depending on $D := |\text{Dom}|$; the case (a) where $D \leq d_1 \varepsilon^{-2}$ for some sufficiently large constant

⁴By using a different technique, we can show some *deterministic* and *non-adaptive* witness-list finding algorithm of the same order of query complexity and list size respectively; that is, the upper bound can be achieved with deterministic and non-adaptive queries, while the almost tight lower bound (Theorem 6) holds even for randomized and adaptive queries. A drawback is that we do not know whether it can be executed in polynomial-time. For the completeness, we give the algorithm in Appendix, Theorem 9.

d_1 (that will be specified later), and the case (b) where $D > d_1 \varepsilon^{-2}$. (What follows is for the case (b); the proof for the case (a) is given in Appendix.)

Consider the case (b), i.e., the case where Dom is sufficiently large. Let \mathcal{OK}_w denote the set of all oracles that are $(1/2 - \varepsilon)$ -error bounded. Our goal is to estimate the success probability α defined by

$$\alpha = \min_{w \in \Omega} \min_{\tilde{A}_w \in \mathcal{OK}_w} \Pr_{s: \{0,1\}^r} [\mathcal{F}(s, \tilde{A}_w(Q)) \ni w].$$

Again by using Yao's principle, we consider some distributions on $w \in \Omega$ and on $\tilde{A}_w \in \mathcal{OK}_w$, and discuss the probability of $\mathcal{F}(s, \tilde{A}_w(Q)) \ni w$ under these distributions while any $s \in \{0,1\}^r$ is fixed.

For the distribution of witnesses, we consider the uniform distribution on Ω ; the symbol Ω is used also for denoting this distribution. For defining a distribution on \mathcal{OK}_w and for analyzing the adaptive query computation of the algorithm, we use a folklore argument attributed to Rudich (see, e.g., [GNW95]). We use, as a "noise function", a random function Δ defined on Dom ; Δ takes, for each $Q \in Dom$ independently, value 0 with probability $1/2 + 2\varepsilon$ and value 1 with probability $1/2 - 2\varepsilon$. Then we assume that our oracle is generated by $\tilde{A}_w(Q) = A_w(Q) \oplus \Delta(Q)$; this is our distribution⁵ on \mathcal{OK}_w , which is again denoted by \mathcal{OK}_w .

Now we fix an arbitrary random seed $s \in \{0,1\}^r$. Since s is fixed, the computation of the algorithm and the produced list of witness candidates are determined by a sequence of answers from oracle \tilde{A}_w that is determined by w and Δ . We use a string $a \in \{0,1\}^m$ to denote this *answer sequence* that can be also regarded as a *computation path* of the algorithm. (If the number of queries is smaller than m on some path, then a prefix a' of a is used to determine the computation. That is, in this case, we consider that $a'u$ yields the same answer for any $u \in \{0,1\}^{m-|a'|}$; for the sake of the following analysis, we regard the case $u_i = 1$ as the case that the oracle makes an error.) Note that, no matter which w is given, every answer sequence a occurs depending on Δ , and that the algorithm behaves in the same way on the same answer sequence. On the other hand, depending on w , the probability that each a occurs may differ. For each $w \in \Omega$ and $a \in \{0,1\}^m$, let $E_{a,w}$ denote the event that the algorithm receives this answer sequence a with respect to w , and let $\#_1(a,w)$ denote the number of queries Q such that $\Delta(Q) = 1$. Then the probability $p(a,w)$ that this event holds is

$$p(a,w) = \left(\frac{1}{2} - 2\varepsilon\right)^{\#_1(a,w)} \left(\frac{1}{2} + 2\varepsilon\right)^{m - \#_1(a,w)}. \quad (2)$$

Here we assume, without loss of generality, that all queries on each computation path are different and hence errors occur independently along each computation path.

We fix any $w \in \Omega$ and discuss the algorithm's success probability α_w . We say that an answer sequence is *good* (with respect to w) if it yields a list containing w . Though $E_{a,w}$ and $E_{a',w}$ may be correlated, they are disjoint; hence, we have

$$\alpha_w = \sum_{a:\text{good}} p(a,w) = \sum_{a:\text{good}} \left(\frac{1}{2} - 2\varepsilon\right)^{\#_1(a,w)} \left(\frac{1}{2} + 2\varepsilon\right)^{m - \#_1(a,w)}.$$

We would like to express the number M_w of good answer sequences in terms of this α_w . Consider the ordering of sequences a with respect to $p(a,w)$. Note that M_w is minimized if the set of good sequences consists of the first M_w sequences in this ordering. For the case where $\alpha_w < 1/2$, let k_w be the smallest number satisfying

$$\alpha_w \leq \sum_{i=0}^{k_w} \binom{m}{i} \left(\frac{1}{2} - 2\varepsilon\right)^i \left(\frac{1}{2} + 2\varepsilon\right)^{m-i}, \quad (3)$$

⁵Note that under this distribution some \tilde{A}_w may not be $(1/2 - \varepsilon)$ -error bounded although this probability is small since $|Dom| \geq d_1 \varepsilon^{-2}$ and d_1 is sufficiently large. Thus, precisely speaking, the distribution that we should consider is the one that defines oracles by using Δ satisfying the condition that $|\{Q \in Dom \mid \Delta(Q) = 1\}| \leq (1/2 - \varepsilon)|Dom|$. But the difference is within at most constant factor, and we argue here by using the distribution \mathcal{OK}_w defined above.

and let δ_w be defined so that $k_w = \lfloor (1/2 - 2\varepsilon - \delta_w)m \rfloor$ holds. Then we have $\delta_w > 0$, and by the Hoeffding bound the righthand side is at most $\exp(-2\delta_w^2 m)/2$, and hence,

$$\alpha_w \leq \frac{1}{2} \exp(-2\delta_w^2 m). \quad (4)$$

On the other hand, for the case where $\alpha_w \geq 1/2$, we simply set $k_w = \lfloor (1/2 - 2\varepsilon)m \rfloor$ and $\delta_w = 0$. Then from the above observation, we have

$$M_w \geq \sum_{i=0}^{k_w-1} \binom{m}{i} \geq \binom{m}{k_w} \geq \frac{c_{\text{str}} 2^m}{\sqrt{m}} \exp(-4(2\varepsilon + \delta_w)^2 m), \quad (5)$$

where the last bound is from Claim 3 below.

Now we bound M_w by α_w . First consider the special case where $\varepsilon \leq \delta_w$. In this case, we have $\alpha_w < 1/2$, and from (4), we immediately have $\alpha_w \leq \exp(-2\delta_w^2 m) \leq \exp(-2\varepsilon^2 m) \leq \exp(-2c_1 n) = 2^{\Omega(-n)}$, which leads to our desired bound. Thus, in the following, we consider the case $\delta_w \leq \varepsilon$. Recall that $m = c_1 n / \varepsilon^2$ for $c_1 = 1/640$. Then from (4) and (5) (and using the bound $\alpha_w \leq 1$ for the case $\alpha_w \geq 1/2$), we have, for some constant $d_2 > 0$, that

$$\begin{aligned} M_w &\geq \frac{c_{\text{str}} 2^m}{\sqrt{m}} \exp(-4(2\varepsilon + \delta_w)^2 m) \\ &\geq \frac{c_{\text{str}} 2^m}{\sqrt{m}} \exp(-32\varepsilon^2 m - 4\delta_w^2 m) \geq \frac{d_2 2^m 2^{-n/10}}{\sqrt{m}} \alpha_w^2. \end{aligned} \quad (6)$$

Note that each answer sequence cannot be good for more than ℓ witnesses, and that there are 2^m answer sequences. Hence, we have $\sum_{w \in \Omega} M_w \leq \ell 2^m$, and thus from (6) we have $\sum_{w \in \Omega} \alpha_w^2 \leq \frac{\ell \sqrt{m} 2^{n/10}}{d_2}$. Then by the Cauchy-Schwartz inequality, we have

$$\left(\sum_{w \in \Omega} \alpha_w \right)^2 \leq \left(\sum_{w \in \Omega} \alpha_w^2 \right) \cdot \left(\sum_{w \in \Omega} 1 \right) \leq \frac{\ell \sqrt{m} 2^{n/10} 2^n}{d_2} = \frac{\ell \sqrt{m} 2^{11n/10}}{d_2}.$$

Thus, we have

$$\Pr_{w: \Omega, \tilde{A}_w: \mathcal{OK}_w} [\mathcal{F}(s, \tilde{A}_w) = w] = \frac{1}{2^n} \sum_{w \in \Omega} \alpha_w \leq \frac{\sqrt{\ell \sqrt{m} 2^{-9n/20}}}{\sqrt{d_2}} = O\left(\frac{\ell 2^{-\Omega(n)}}{\varepsilon}\right).$$

Since this holds for all $s \in \{0, 1\}^r$, by Yao's principle, we have the desired bound for α . \square

Claim 3 *By the Stirling bound, for some constant $c_{\text{str}} > 0$ and for any $\varepsilon \leq 1/4$, we have*

$$\binom{u}{(\frac{1}{2} - \varepsilon)u} \geq \frac{c_{\text{str}} 2^u}{\sqrt{u}} \exp(-4\varepsilon^2 u).$$

Recall that the algorithm given in Proposition 5 outputs a list of size $O(1/\varepsilon^2)$ even with the optimal query complexity. In fact, this list size is optimal up to a constant factor subject to achieving constant success probability. More precisely, we have the following bound.

Theorem 7 *Consider any witness-list finding algorithm with list size ℓ , and for any sufficiently small $\varepsilon > 0$, let α be its success probability with $(1/2 - \varepsilon)$ -error bounded oracle. Then for some constants $c_2 > 0$, if $n > c_2 \log(1/\varepsilon)/\varepsilon^2$, then we have $\alpha = O(\ell \varepsilon^2)$.*

References

- [BCGL92] S. Ben-David, B. Chor, O. Goldreich, and M. Luby, On the theory of average-case complexity, *Journal of Computer and System Sciences*, 44(2):193–219, 1992.
- [BT06] A. Bogdanov and L. Trevisan, On worst-case to average-case reductions for NP problems, *SIAM J. Comput.*, 36(4):1119–1159, 2006.
- [DKvMW12] H. Dell, V. Kabanets, D. van Melkebeek, and O. Watanabe, Is the Valiant-Vazirani isolation lemma improvable?, in *Proc. 27th Conference on Computational Complexity*, 2012, to appear.
- [Gol01] O. Goldreich, *Foundations of Cryptography, Basic Tools*, Cambridge Univ. Press, 2001.
- [GHK11] V. Guruswami, J. Håstad, and S. Kopparty, On the list-decodability of random linear codes, *IEEE Transactions on Information Theory*, 57(2):718–725, 2011.
- [GL89] O. Goldreich and L. Levin, Hard-core predicates for any one-way function, in *Proc. 21st ACM Sympos. on the Theory of Comput.*, 25–32, 1989.
- [GNW95] O. Goldreich, N. Nisan, and A. Wigderson, On Yao’s XOR-lemma, Technical Report TR95-050, ECCC Report, 1995.
- [GV10] V. Guruswami and S. Vadhan, A lower bound on list size for list decoding, *IEEE Transactions on Information Theory*, 56(11):5681–5688, 2010.
- [Rot06] R.M. Roth, *Introduction to Coding Theory*. Cambridge Univ. Press, 2006.
- [VV86] L. Valiant and V. Vazirani, NP is as easy as detecting unique solutions, *Theoretical Computer Science*, 47:85–93, 1986.
- [Yao77] A.C. Yao, Probabilistic computations: toward a unified measure of complexity, *Proc. of the 18th IEEE Sympos. on Foundations of Comput. Sci.*, IEEE, 222–227, 1977.

Appendix

We state proofs left out of the body of the paper.

Proof. (Proof of Case (a) in the proof of Theorem 6)

We list all possible queries in Dom under some ordering, and let Q_1, \dots, Q_D be this list. Oracle answers to these queries can be specified naturally by a string $a \in \{0, 1\}^D$. For any $a \in \{0, 1\}^D$ and $w \in \Omega$, we say that a and w are $(1/2 + \varepsilon)$ -consistent if the Hamming distance between a and $A_w(Q_1)A_w(Q_2) \cdots A_w(Q_D)$ is $\leq D' := (1/2 - \varepsilon)D$. Note that each $w \in \Omega$ has $\binom{D}{D'}$ strings that are $(1/2 + \varepsilon)$ -consistent with w ; hence, the total number of pairs (a, w) that are $(1/2 + \varepsilon)$ -consistent is exactly $2^n \binom{D}{D'}$. Thus, there exists some $a_0 \in \{0, 1\}^D$ that is $(1/2 + \varepsilon)$ -consistent with at least

$$\frac{2^n \binom{D}{D'}}{2^D} \geq \frac{2^n c_{\text{str}} 2^D}{2^D \sqrt{D} \exp(4\varepsilon^2 D)} \geq \frac{c_{\text{str}} 2^n}{\sqrt{D} \exp(4d_1)} = O(\varepsilon 2^n)$$

witnesses, where we use the bound given by Claim 3 below for $\binom{D}{D'}$. Let W_0 denote the set of such witnesses.

Now we again follow Yao's principle and consider some distribution on witnesses while we fix a random seed used by $(\mathcal{Q}, \mathcal{F})$ to any s in $\{0, 1\}^r$. Our distribution is simply the uniform distribution on W_0 . Note that a_0 is $(1/2 - \varepsilon)$ -error bounded for any witness $w \in W_0$. Thus, for any $w \in W_0$, we consider the execution of the algorithm with random seed s and oracle answers specified by a_0 . Then the output, i.e., the list of ℓ candidates for w , is fixed, and it can be correct for at most ℓ witnesses $w \in W_0$. Hence, under the uniform distribution on W_0 , the success probability of our algorithm (with respect to s and a_0) is at most $\frac{\ell}{|W_0|} = O(\ell 2^{-n}/\varepsilon)$, which can be used as an upper bound for the success probability of the algorithm in the worst case by Yao's principle. \square

Claim 3. By the Stirling bound, for some constant $c_{\text{str}} > 0$ and for any $\varepsilon \leq 1/4$, we have

$$\binom{u}{(\frac{1}{2} - \varepsilon)u} \geq \frac{c_{\text{str}} 2^u}{\sqrt{u}} \exp(-4\varepsilon^2 u).$$

Proof. We use the following approximation by Stirling: For some constants c_{low} and c_{up} and for any $n \geq 1$, we have

$$c_{\text{low}} \sqrt{n} \left(\frac{n}{e}\right)^n < n! < c_{\text{up}} \sqrt{n} \left(\frac{n}{e}\right)^n.$$

Let $\beta = 1/2 - \varepsilon$. Then for some constant $d > 0$, we have

$$\begin{aligned} \binom{u}{\beta u} &= \frac{u!}{((1-\beta)u)!(\beta u)!} \geq \frac{d}{\sqrt{\beta(1-\beta)u}} \cdot \frac{u^u}{((1-\beta)u)^{(1-\beta)u} (\beta u)^{\beta u}} \\ &= \frac{d}{\sqrt{\beta(1-\beta)u}} \cdot (1-\beta)^{-(1-\beta)u} \beta^{-\beta u}. \end{aligned}$$

This can be modified further to

$$\begin{aligned} \binom{u}{\beta u} &\geq \frac{d}{\sqrt{(1/4 - \varepsilon^2)u}} \cdot \left(\frac{1}{2} - \varepsilon\right)^{-(1/2 - \varepsilon)u} \left(\frac{1}{2} + \varepsilon\right)^{-(1/2 + \varepsilon)u} \\ &= \frac{d}{\sqrt{(1/4 - \varepsilon^2)u}} \cdot \left(\frac{1}{4} - \varepsilon^2\right)^{-u/2 + \varepsilon u} \left(\frac{1}{2} + \varepsilon\right)^{-2\varepsilon u} \\ &\geq \frac{d}{\sqrt{(1/4 - \varepsilon^2)u}} \cdot 4^{u/2 - \varepsilon u} 2^{2\varepsilon u} (1 + 2\varepsilon)^{-2\varepsilon u} \\ &= \frac{d 2^u}{\sqrt{(1/4 - \varepsilon^2)u}} \cdot (1 + 2\varepsilon)^{-2\varepsilon u} \geq \frac{c_{\text{str}} 2^u}{\sqrt{u}} \cdot \exp(-4\varepsilon^2 u). \end{aligned}$$

□

Theorem 7. Consider any witness-list finding algorithm with list size ℓ , and for any sufficiently small $\varepsilon > 0$, let α be its success probability with $(1/2 - \varepsilon)$ -error bounded oracle. Then for some constants $c_2 > 0$, if $n > c_2 \log(1/\varepsilon)/\varepsilon^2$, then we have $\alpha = O(\ell\varepsilon^2)$.

Proof. The proof follows immediately from the lemma stated below, which is a special case of the result of Guruswami and Vadhan [GV10] on a lower bound of list size for list-decoding, and the outline of Case (a) in the proof of Theorem 6 with Yao's principle.

Lemma 8 ([GV10]) For some constants $d_1, d_2 > 0$, the following holds: For any integer $D > 0$ and any $C \subseteq \{0, 1\}^D$, if $\varepsilon < d_1$ and $|C| > 2^{d_2/(2\varepsilon^2 \log(1/\varepsilon))}$, then there exists some $a_0 \in \{0, 1\}^D$ such that $|C \cap B(1/2 - \varepsilon, a_0)| = \Omega(\varepsilon^{-2})$.

Consider any witness-list finding algorithm $(\mathcal{Q}, \mathcal{F})$. Consider any advantage ε , $0 < \varepsilon < d_1$, and let c_2 be the constant d_2 specified in Lemma 8. Let $D = |\text{Dom}|$, and we list all possible queries Q_1, \dots, Q_D in Dom under some ordering. We view C in Lemma 8 as the set of oracle answers $A_w(Q_1), \dots, A_w(Q_D)$ to these queries for some $w \in \Omega (= \{0, 1\}^n)$. We may assume that $|C| \geq 2^{n/2}$; since otherwise, there are large number of w 's with the same oracle answers, and the success probability α becomes small and the desired upper bound for α can be shown rather easily (by using the condition that $n > c_2/\varepsilon^2 \log(1/\varepsilon)$). Thus, the condition $|C| > 2^{d_2/(2\varepsilon^2 \log(1/\varepsilon))}$ of the lemma is satisfied if $n > c_2/\varepsilon^2 \log(1/\varepsilon)$. Then by Lemma 8, there exists $a_0 \in \{0, 1\}^D$ that is $(1/2 + \varepsilon)$ -consistent with $\Omega(\varepsilon^{-2})$ witnesses. Let W_0 be the set of such witnesses. Now, we again follow Yao's principle, and by the same reasoning of Case (a) in the proof of Theorem 6, the desired upper bound $\ell/|W_0| = O(\ell\varepsilon^2)$ for α is derived. □

Finally, we give a deterministic witness finding algorithm whose performance (except its time bound) is similar to the improved Goldreich-Levin algorithm (see the footnote on page 8).

Theorem 9 There is a deterministic, nonadaptive, and $(1/2 - \varepsilon)$ -error tolerant witness-list finding algorithm with $O(n/\varepsilon^2)$ query complexity and $O(1/\varepsilon^2)$ list size.

Proof. The proof is almost trivial from the following lemma, which provides an optimal list-decodable code.

Lemma 10 ([GHK11]) For some $m = O(n/\varepsilon^2)$, there exists a set $C \subseteq \{0, 1\}^m$ of size 2^n such that for every $v \in C$, we have $|B(1/2 - \varepsilon, v) \cap C| = O(1/\varepsilon^2)$, where $B(\gamma, v) \subseteq \{0, 1\}^m$ denotes the Hamming ball of radius γm centered at v .

(Note that [GHK11] proved much stronger statement.)

We use this lemma to design our algorithm. Consider any ε and fix any n . Let C be the set given in Lemma 10. We associate every witness $w \in \Omega (= \{0, 1\}^n)$ with an element in C in any one-to-one manner. Denote by $C(w)$ an element in C corresponding to $w \in \Omega$. Our algorithm is deterministic and works as follows: The i -th query is given by $Q_i = \{v : C(v)_i = 1\}$, where $C(v)_i$ denotes the i th bit of $C(v) \in \{0, 1\}^m$. Namely, the query “ $\{w\} \cap Q_i \neq \emptyset$?” asks whether the i -th bit of $C(w)$ is 1 or not. On the answers $\tilde{A}_w(Q_1), \dots, \tilde{A}_w(Q_m)$, the algorithm outputs a list of all the elements of $\{v | C(v) \in B(1/2 - \varepsilon, \tilde{\alpha}_w) \cap C\}$, where $\tilde{\alpha}_w := (\tilde{A}_w(Q_1), \dots, \tilde{A}_w(Q_m)) \in \{0, 1\}^m$. Note that we do not know an efficient way for enumerating these elements and thus the algorithm makes such a list by brute force.

From a coding-theoretic viewpoint, w is an n -bit message, $C(w)$ is its codeword of m -bit length, and $\tilde{\alpha}_w$ is an corrupted codeword. Hence, if \tilde{A}_w makes at most $(1/2 - \varepsilon)m$ errors, then we have the target witness in $\{v | C(v) \in B(1/2 - \varepsilon, \tilde{\alpha}_w) \cap C\}$. Clearly, the algorithm makes m queries and it follows from Lemma 10 that the list size is $O(1/\varepsilon^2)$. □