Lower bounds for witness-finding algorithms

Jeffrey Finkelstein

November 17, 2015

1 Introduction

Can a procedure that decides whether a Boolean formula has a satisfying assignment help to find such an assignment? The naïve adaptive “search-to-decision” reduction uses a linear number of (quite weak) queries. Is there a lower bound on the number of queries required for a nonadaptive search-to-decision reduction? We report on lower bounds for various classes of queries.

Most interesting types of queries require at least $\Omega(n^2)$ queries to find a witness. We were hoping to use lower bounds on these types of witness-finding algorithms to show that randomized reductions to $\mathsf{C=P}$ are less powerful than randomized reductions to $\mathsf{P_P}$, but we were unable to do so.

Figure 1 contains a summary of the upper and lower bounds for nonadaptive witness-finding algorithms. In addition, adaptive $Q_0$ algorithms use $\Theta(n)$ queries, and adaptive $Q_1$ witness-finding algorithms with high success probability use $\Theta(n^2)$ queries. TODO Currently the lower bound for the query complexity of adaptive $Q_1$ witness-finding algorithms with success probability $\frac{1}{2}$ is $\Omega(n)$. Can we either improve this to $\Omega(n^2)$ or provide a $O(n)$ upper bound?

TODO Compare these randomized witness-finding algorithms with “bounded query hierarchy” algorithms, in which the query generator and the witness generator are not randomized (but possibly nondeterministic), and the goal is to decide a language, not necessarily output a witness.

2 Definitions

Let $\Sigma = \{0, 1\}$. For any natural number $n$, the set $\Sigma^n$ denotes the set of binary strings of length $n$. For each $b \in \{0, 1\}$, let $\bar{b} = 1 - b$. Let $[n] = \{1, \ldots, n\}$.
Figure 1: When considering nonadaptive witness-finding algorithms, $Q_{\text{direct}}$ algorithms require $\Omega(2^n)$ queries, $Q_{\text{mono}}$, $Q_{\text{NP}}$, and $Q_k$ algorithms require $\Omega(n^2)$ queries, and $Q_{\text{all}}$ algorithms require $\Omega(n)$ queries. In this diagram, lower bounds propagate downward and upper bounds propagate upward.
Denote by \( \log n \) the base-2 logarithm.

A language \( L \subseteq \Sigma^* \) is a **witnessed language** if there exists a language \( L' \subseteq \Sigma^* \times \Sigma^* \), called the **witness relation**, such that for each \( x \) of length \( n \), we have \( x \in L \) if and only if there is a \( w \in \Sigma^n \) such that \( (x, w) \in L' \). (We don’t consider strings \( w \) of length polynomial in \( n \) because a language in any “well-behaved” complexity class can be converted to an equivalent language in which all witnesses have length exactly \( n \) by padding the string \( x \) by an appropriate amount.)

Suppose \( n \) is a positive integer. A **witness set** is a nonempty subset of \( \Sigma^n \) denoted by \( W \), and each element of \( W \) is a witness. A **query** is a Boolean predicate \( Q : \text{pow}(\Sigma^n) \rightarrow \{0, 1\} \).

**Definition 2.1** ([8]). Let \( W \) be a collection of witness sets and \( Q \) a collection of queries. A **(nonadaptive) \( Q \)-query witness-finding algorithm** is a joint distribution \( (Q_1, \ldots, Q_m) \) on \( Q^m \), for some natural number \( m \), and a function \( f : \{0, 1\}^m \rightarrow \Sigma^n \) such that \( \Pr[f(Q_1(W'), \ldots, Q_m(W')) \in W] > \frac{1}{2} \) for each \( W \in W \). The number \( m \) is the **query complexity** of the algorithm.

This definition can be generalized to allow for success probabilities other than \( \frac{1}{2} \), and to allow for adaptive queries. Unless otherwise specified, the success probability is assumed to be \( \frac{1}{2} \) and the queries are assumed to be nonadaptive.

**TODO Define an “adaptive distribution sequence” (which means that \( Q_{i+1}(W) \) is actually \( Q_{i+1}(W, Q_1(W), \ldots, Q_{i-1}(W)) \)).**

- An \( (= k) \) query is a query \( Q_S \) of the form \( Q_S(W) = 1 \) if and only \( |S \cap W| = k \), for any \( S \subseteq \Sigma^n \) and any \( W \in \text{pow}(\Sigma^n) \). The collection of all \( (= k) \) queries is denoted \( Q_k \).
- A \( (\geq k) \) query is a query \( Q_S \) of the form \( Q_S(W) = 1 \) if and only \( |S \cap W| \geq k \), for any \( S \subseteq \Sigma^n \) and any \( W \in \text{pow}(\Sigma^n) \). The collection of all \( (\geq k) \) queries is denoted \( Q_{\geq k} \).
- An element of \( Q_1 \) is called an isolation query.
- An element of \( Q_0 \) is called an emptiness query.

**Definition 2.2.** Let \( W \) be a collection of witness sets. A **hitting algorithm** is an adaptive sequence of distributions \( (Q_1, \ldots, Q_m) \) on \( Q^m \), for some natural number \( m \), such that \( \Pr[Q_m(W) = 1] > \frac{1}{2} \). The number \( m \) is the **query complexity** of the algorithm.

This definition is a complexity theoretic restatement of the combinatorial “hitting game”, as described in [9, Section 3] (the original definition seems to be from [2, Definition 5]).

A hitting algorithm is always an adaptive algorithm; a nonadaptive hitting algorithm makes little sense here because it would be essentially the same as simply guessing an element of the witness set at random.

A witness-finding algorithm must output an element of \( W \), whereas a hitting algorithm need not.

**TODO define polynomial-time algorithms (provide polysize circuits that encode the query sets)**
3 Relationship among algorithms

Proposition 3.1. Let \( k \) be a positive integer.

- There is a \( Q_{\geq k} \) witness-finding algorithm if and only if there is a \( Q_{< k} \) witness-finding algorithm.
- There is a \( Q_k \) witness-finding algorithm if and only if there is a \( Q_{\neq k} \) witness-finding algorithm.

Furthermore, in both cases, the query complexity of the algorithms is exactly the same.

Proof. We prove the first equivalence; the second is similar. For each \( Q_S \) in \( Q_{\geq k} \) there is a corresponding \( Q_{\leq k} \) such that \( Q_S(W) = Q_{\leq k}(W) \) for each witness set \( W \), and furthermore, this correspondence is a bijection. Therefore if we define the function \( \bar{f} \) by

\[
\bar{f}(A_1, \ldots, A_m) = f(\overline{A_1}, \ldots, \overline{A_m}),
\]

for all possible bits \( A_1, \ldots, A_m \), we get the desired equivalence: \( (Q_1, \ldots, Q_m, f) \) is equivalent to \( (Q_1, \ldots, Q_m, \bar{f}) \).

Corollary 3.2. There is a \( Q_0 \) witness-finding algorithm if and only if there is a \( Q_{\neq 0} \) witness-finding algorithm.

Proposition 3.3. Let \( k \) and \( m \) be natural numbers. If there is a \( Q_{\leq k} \) witness-finding algorithm with query complexity \( m \), then there is a \( Q_{\geq k} \cup Q_{\geq k+1} \) witness-finding algorithm with query complexity \( 2m \).

Proof. Suppose \( ((Q_{S_1}, \ldots, Q_{S_m}), f) \) is the \( Q_k \) witness-finding algorithm. By Proposition 3.1 it suffices to show a \( Q_{\geq k} \cup Q_{\leq k} \) witness-finding algorithm. For each \( Q_{S_i} \), construct two queries \( Q_{\leq k} \) and \( Q_{\geq k} \) defined as \( Q_{\leq k}(W) = 1 \) if and only if \( |S_i \cap W| \geq k \) and \( Q_{\geq k}(W) = 1 \) if and only if \( |S_i \cap W| \leq k \). Define \( f' \) by

\[
f'(A_1, \ldots, A_m, B_1, \ldots, B_m) = f(A_1 \land B_1, \ldots, A_m \land B_m),
\]

for all bits \( A_1, \ldots, A_m \) and all bits \( B_1, \ldots, B_m \). Since \( Q_{\leq k}(W) \land Q_{\geq k}(W) = Q_{S_i}(W) \), the correctness of \( ((Q_{\leq k_1}, \ldots, Q_{\leq k_m}, Q_{\geq k_1}, \ldots, Q_{\geq k_m}), f') \) follows from the correctness of \( ((Q_1, \ldots, Q_m), f) \).

The problem of finding a hitting algorithm reduces to the problem of finding a \( Q_1 \) witness-finding algorithm.

Lemma 3.4. Let \( m \) be a natural number. If there is a \( Q_1 \) witness-finding algorithm with query complexity \( m \), then there is a hitting algorithm with query complexity \( m + 1 \).
Proof. Suppose \(((Q_{s_1}, \ldots, Q_{s_m}), f)\) is the \(Q_1\) witness-finding algorithm. For any witness set \(W\), let \(S_{m+1}\) be the singleton set containing the element \(f(Q_{s_1}, \ldots, Q_{s_m})\). Then \((Q_{s_1}, \ldots, Q_{s_{m+1}})\) is a hitting algorithm, since the probability that \(f(Q_{s_1}(W), \ldots, Q_{s_m}(W)) \in W\) is exactly the probability that \(Q_{s_{m+1}}(W) = 1\).

On the other hand, there does not seem to be a general reduction from a \(Q_1\) witness-finding algorithm to a hitting algorithm. In order to output a single witness knowing only that a witness is contained in a particular query set doesn’t help, unless the query set is small; there is no such guarantee on the size of that set.

Conjecture 3.5. Under an appropriate setting of parameters, if the existence of a hitting algorithm implies the existence of an isolation query algorithm, then \(\text{RP} = \text{NP}\). TODO But both types of algorithms exist with the same parameters, as shown in the next section.

Now, consider the possibility that a \(Q_k\) witness-finding algorithm implies a \(Q_{k-1}\) witness-finding algorithm. Intuitively, in order for this to work, the latter algorithm would have to simulate the queries \(Q_i\) from the former algorithm with some sort of collection of queries of the form \(Q_i \setminus \{q_{i,j}\}\), where \(q_{i,j}\) is the \(j\)th member of \(Q_i\). However, the oracle answers would be inconsistent, because we don’t know beforehand whether \(q_{i,j}\) is in the witness set or not (causing a discrepancy between the answers of the two oracles). A similar problem arises when attempting the converse simulation: the simulated queries would be something of the form \(Q_i \cup \{v_j\}\), where \(v_j\) is the \(j\)th element of \(\Sigma^n \setminus Q_i\), thus there would be a discrepancy between the answers of the two oracles. We therefore conjecture that there is evidence that such a simulation is unlikely.

Conjecture 3.6. For each \(k \in [2^n]\), if the existence of a \(Q_k\) witness-finding algorithm implies the existence of a \(Q_{k+1}\) witness-finding algorithm, then \(\text{RP} = \text{NP}\). TODO But both such algorithms exist unconditionally, as shown in the next section.

4 Upper bounds

Theorem 4.1. There is a polynomial time adaptive \(Q_0\) witness-finding algorithm with success probability 1 and query complexity \(O(n)\).

Proof idea. By Corollary 3.2 it suffices to show a \(Q_{\bar{0}}\) witness-finding algorithm. This the well-known bit-by-bit \(P^{\text{NP}}\) computable function for computing a witness for a given string.

For each \(i \in [n]\) and each \(b \in \{0, 1\}\), define the query \(S_{i,b}\) inductively (or in other words, adaptively) as follows. If \(i = 1\), then \(S_{i,b} = \{bv \mid v \in \{0, 1\}^{i-1}\}\). For each subsequent pair of queries, let \(u = b_1 \cdots b_{i-1}\), where \(b_j\) is a bit that caused the oracle to answer “yes” on query \(S_{j,b_j}\) for each \(j \in [i-1]\), and let \(S_{i,b} = \{uvb \mid v \in \{0, 1\}^{n-1}\}\). Finally, given the responses to \(S_{n,0}\) and \(S_{n,1}\), which
are singleton sets, the algorithm simply outputs the sole element of the query set that caused the oracle to answer “yes”.

The total number of queries is $2^n$, which is in $O(n)$.

This upper bound means that any class of queries containing $Q_0$ as a subset also satisfies this very strong upper bound.

**Theorem 4.2** ([8, Theorem 1.4] (citing [3])). There is a polynomial time $Q_0$ witness-finding algorithm with query complexity $O(n^2)$.

**Proof idea.** By Corollary 3.2 it suffices to show a $Q_{\neq 0}$ witness-finding algorithm. This algorithm uses the Valiant–Vazirani Isolation Lemma in each query to guess each bit of a possible witness (in parallel).

First, consider a simpler set of queries. Let $s$ be a string in $\Sigma^r$. For each $i \in [n]$ and each $b \in \Sigma$, let $S_{i,b} = \{w \mid w \in C \land w_i = b\}$, where $C$ is the isolating set from the Valiant–Vazirani Isolation Lemma (for example, the set of all strings whose image is zero under a pairwise independent hash function). The Valiant–Vazirani Isolation Lemma proves that this algorithm has success probability $\Omega(\frac{1}{n})$ and query complexity $O(n)$. In order to get success probability $\Omega(1)$, we create $O(n)$ collections of $O(n)$ queries, with an independent set $C_{s_k}$ for each collection. Now our queries are $S_{k,i,b} = \{w \mid w \in C_{s_k} \land w_i = b\}$, for each $k \in [O(n)]$, each $i$, and each $b$.

In [7], the authors call the above theorem the “truth-table reduction” analog of the “many-one reduction” result of [4, Theorem 4.2]. However, this is not strictly precise, since the former involves $Q_0$ queries, whereas the latter involves $Q_1$ queries. Fortunately, the proof uses the Valiant–Vazirani Isolation Lemma (that is, the sets $C_{s_k}$), so the probability that $|S \cap W| \neq 0$ equals the probability that $|S \cap W| = 1$. (On the other hand, there is no other known relationship between $Q_0$ witness-finding algorithms and $Q_1$ witness-finding algorithms, other than the fact that $Q_0 \cup Q_1 \subseteq \cup_{k \in \mathbb{N}} Q_k$.)

**Corollary 4.3.** There is a polynomial time $Q_1$ witness-finding algorithm with query complexity $O(n^2)$.

TODO Double check this logic.

**Corollary 4.4.** There is a polynomial time hitting algorithm with query complexity $O(n^2)$.

**Proof.** Combine the previous corollary with Lemma 3.4.

---

## 5 Lower bounds

A query $Q$ is a *monotone query* if for all $W$ and $W'$, subsets of $\Sigma^n$, if $W \subseteq W'$, then $Q(W) = 1$ implies $Q(W') = 1$. The collection of all monotone queries is denoted $Q_{\text{mono}}$. 
Theorem 5.1 ([8, Theorem 1.3]). Each $Q_{\text{mono}}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

The proof of this theorem relies on the existence of a distribution on witness sets that places all witnesses far apart, so it is hard to guess both the number and location of the witnesses. This distribution comes from [4, Theorem 4.2].

An infinite sequence of queries $(Q_1, Q_2, \ldots)$ is an NP query if there is a non-deterministic oracle Turing machine $M$ such that for each $n$, we have $Q^n(W) = 1$ if and only if $M^W(1^n)$ accepts (where $Q^n$ denotes a query on witness sets of strings of size $n$). The collection of all NP queries is denoted $Q_{\text{NP}}$.

Theorem 5.2 ([8, Theorem 1.6]). Each $Q_{\text{NP}}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

Corollary 5.3 ([7, Theorem 2]). Each $Q_{0}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

Proof. This is a special case of both Theorem 5.1 and Theorem 5.2.

The next corollary is a generalization of [7, Remark 2].

Corollary 5.4. Suppose $k \in [2^n]$. Each $Q_{\geq k}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

Proof. This is a special case of Theorem 5.1.

Corollary 5.5. Suppose $k \in [2^n]$. Each $Q_{k}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

Proof. This is a proof by contradiction. Assume there is a $Q_{k}$ witness-finding algorithm with query complexity $o(n^2)$. By Proposition 3.3 there is a $Q_{\geq k} \cup Q_{\geq k+1}$ witness-finding algorithm with query complexity $o(n^2)$. Since $Q_{\geq k}$ and $Q_{\geq k+1}$ are both subsets of the collection of all monotone queries, each $Q_{\geq k} \cup Q_{\geq k+1}$ witness-finding algorithm has query complexity $\Omega(n^2)$, by Theorem 5.1. This is a contradiction.

Corollary 5.6. Each $Q_{1}$ witness-finding algorithm has query complexity $\Omega(n^2)$.

Proof. This is a special case of the previous corollary.

Theorem 5.7 ([9, Theorems 3.3 and 3.4]). Each hitting algorithm with success probability $\Omega(1 - 2^{-n})$ has query complexity $\Omega(n^2)$. Furthermore, this is true even when restricted to witness sets of cardinality two.

The proof of this theorem relies on the probabilistic method to show the existence of a witness set that is hard to “hit”, found in [1].

Corollary 5.8. Each adaptive $Q_{1}$ isolation query algorithm with success probability $\Omega(1 - 2^{-n})$ has query complexity $\Omega(n^2)$. Furthermore, this is true even when restricted to witness sets of cardinality two.
Proof. This is a proof by contradiction. Assume there is an adaptive $Q_1$ witness-finding algorithm with success probability $\Omega(1 - 2^{-n})$ and query complexity $o(n^2)$. By Lemma 3.4 there is a hitting algorithm with the same success probability and query complexity (up to constant factors). This would contradict the previous theorem, hence no such algorithm exists.

Theorem 5.9 ([9, Theorem 3.2]). Each hitting algorithm with expected success probability $\Omega(1 - 2^{-n})$ has expected query complexity $\Omega(n)$.

TODO I can’t understand what the random variable is for which there is an expectation here.

6 Comparing equality to counting queries

Now we consider more general types of queries, queries to $PP$ or $C=P$ oracles. In [9, Section 7], Fred asks, “Does the hypothesis $P^{C=P} = PP$ have any dramatic consequences?” Our intuition is that $P^{C=P}$ is less powerful than $PP$, because the latter can determine the exact number of accepting paths in a computation (or in other words, the number of witnesses in a witness set) by a binary search, whereas the former does not seem to admit this possibility. On the contrary, we know that $NP^{C=P} = NP^{PP}$. Perhaps a first step toward comparing $PP$ and $P^{C=P}$ is comparing $RP^{PP}$ and $RP^{C=P}$.

There is a superficial similarity to the question “is $RP^{PP}$ a subset of $RP^{C=P}$?” and the question “does a witness-finding algorithm with oracle $Q_{\geq k}$ imply a witness-finding algorithm with oracle $Q_k$?” However, we run into a few problems when trying to follow this approach. First, similar to the issues with Conjecture 3.6, the lower bounds for $Q_k$ and $Q_{\geq k}$ are identical, as seen in Corollary 5.4 and Corollary 5.5. Furthermore, if we attempt such a proof, we run into several problems.

Conjecture 6.1. If $RP^{PP} \subseteq RP^{C=P}$, then a $Q_{\geq k}$ witness-finding algorithm implies a $Q_k$ witness-finding algorithm.

Broken proof. Let $W$ be any witness set. Let $((Q_1, \ldots, Q_m), f)$ be a $Q_{\geq k}$ witness-finding algorithm. Let $L$ be a witnessed language with witness relation $L'$. The problem of deciding whether $|Q \cap W| \geq k$ given any query set $Q$ can be decided by a $PP$ oracle. Construct machine $M$ with access to that oracle as follows on random string $s$ of length $r$.

- Make queries $(Q_1, \ldots, Q_m)$ to the $PP$ oracle (which is behaving like the $Q_k$ oracle) to get answer bits $A_1, \ldots, A_m$.
- Run $f(A_1, \ldots, A_m)$ to get potential witness $w$.
- Decide if $(x, w) \in L'$.

TODO Problem 1: need to show that this is a correct $RP^{PP}$ algorithm. By our hypothesis that $RP^{PP} \subseteq RP^{C=P}$, there is also a randomized oracle machine.
that decides $L$ with access to a $C=\mathbb{P}$ oracle. Now we wish to use $M'$ to construct a $Q_k$ witness-finding algorithm $((Q'_1,\ldots,Q'_{m'}),f')$. Define the joint distribution $(Q'_1,\ldots,Q'_{m'})$ as follows.

- Run $M'(x,s)$ and intercept each query $q_1,\ldots,q_{m'}$ that it would make to the $C=\mathbb{P}$ oracle, where $x$ is an input whose witness set is $W$ and $s$ is a random seed for the randomized Turing machine $M'$. **TODO Problem 2:** $M'$ needs access to a particular input string $x$; we may be able to modify the definition of the witness-finding algorithms to allow this. Each query $q_i$ is a question of the form “is $|W_{q_i}| = k$?”, where $W_{q_i}$ is a witness set that corresponds to the input string $q_i$. **TODO Problem 3:** this won’t work, since $Q_k$ only answers questions of the form “is $|W_{q_i} \cap W| = k$?”, but we need it to answer questions like “is $|W_{q_i}| = k$?”, and we don’t get to choose the arbitrary witness set $W$.

- Let $Q'_i = q_i$ for each $i \in [m']$.

Define the function $f'$ as follows on oracle answer bits $A_1,\ldots,A_{m'}$.

- Run $M'(x,s)$ and when $M'$ makes query $q_i$, provide answer $A_i$ to it. **TODO Problem 4:** at this point, all $M'$ tells us is that there is a witness (or none); it gives no indication as to what the witness is. If one of the queries causes the oracle to answer yes and the size of the query set is not too large, we could simply output a random element of the query set for a small penalty in success probability, but in general we have no such guarantees. $\square$

Although this starts to have little to do with the witness-finding algorithm results given in the following sections, another approach to showing that $\mathbb{P}^{C=\mathbb{P}} \neq \mathbb{P}^{\mathbb{P}}$ may be to show that adaptive queries to $C=\mathbb{P}$ are no more powerful than nonadaptive queries, whereas adaptive queries to $\mathbb{P}^{\mathbb{P}}$ are more powerful than adaptive queries. What is known about $\mathbb{P}^{\mathbb{P}}$?

- $\mathbb{P}^{\mathbb{P}}$ is closed under complement, union, and intersection.

- $\mathbb{P}^{\mathbb{P}}$ is closed under polynomial time $k$-round truth-table reductions, for any $k \in O(1))$.

- $\mathbb{P}^{\mathbb{P}} = \mathbb{P}^{\mathbb{GapP}}$ [5, Theorem 4.14].

What is known about $C=\mathbb{P}$?

- $\mathbb{P}^{C=\mathbb{P}[O(1)]} = \mathbb{P}^{C=\mathbb{P}[O(\log n)]}$ (see [6, Theorem 5] and [10, Corollary 4.6]).
7 Entropy on bad witness distributions

In this section $H(X)$ denotes the Shannon entropy of random variable $X$ chosen from a distribution on binary strings, and $H_b(p)$ denotes the binary entropy function on a Bernoulli process with success probability $p$.

Define $W_n^*$, a distribution on witness sets (that is, subsets of $\Sigma^n$), by the following sampling procedure.

- Let $W = \emptyset$.
- Choose $D$ uniformly at random from $[n]$.
- For each $w \in \Sigma^n$, add $w$ to $W$ with probability $2^{-D}$.
- Output $W$.

(Technically, $W$ may be the empty set, but this occurs with probability $o(1)$, so for our purposes, we ignore it. TODO a brief explanation can be found in [8], I believe.)

Lemma 7.1. For any positive integer $n$, for any $k \in [2^n]$, for any $d \in [n]$, and for any $S \subseteq (\{0, 1\}^n \setminus \emptyset)$,

$$
\Pr_{W \sim W_n^*}[|S \cap W| = k|D = d] < \frac{e^{\frac{1}{12}}}{\sqrt{2\pi}} \left( \frac{s}{k(s-k)} \right)^{\frac{1}{2}},
$$

where $s$ denotes the cardinality of $S$.

Proof. For the sake of brevity, let $s$ denote $|S|$, and let $p$ denote the probability $\Pr_{W \sim W_n^*}[|S \cap W| = k|D = d]$. Without loss of generality, assume $s \geq k$, since if $s$ is smaller than $k$, the probability $p$ is zero. Since each element in $W$ is chosen independently with probability $2^{-d}$, the probability that $|S \cap W| = k$ is the probability that exactly $k$ of the elements of $S$ are in $W$ and exactly $s-k$ of the elements are not in $W$. This probability fits the binomial distribution with $s$ trials, $k$ successes, and probability of success $2^{-d}$. Thus

$$
p = \binom{s}{k} (2^{-d})^k (1 - 2^{-d})^{s-k}.
$$

By finding the zeros of the first derivative of $p$ with respect to $d$, we see that $p$ achieves a maximum value at $d = -\log \frac{k}{s}$. TODO Cite a text giving the first derivative of the binomial distribution. When $d$ takes this value,

$$
p = \binom{s}{k} \frac{k^k}{s^k} (s-k)^{s-k}.
$$

Expanding the binomial coefficient as $\frac{s!}{k!(s-k)!}$ and applying Stirling’s approximation,

$$
\sqrt{2\pi m} \left( \frac{m}{e} \right)^m < m! < e^{\frac{1}{12}} \sqrt{2\pi m} \left( \frac{m}{e} \right)^m,
$$
yields the upper bound

\[ p < \frac{e^{\frac{s}{2\pi}}}{\sqrt{2\pi}} \left( \frac{s}{k(s-k)} \right)^{\frac{1}{2}}. \]

Under what relationship between \( k \) and \( |S| \) does the probability become at most one half?

**Lemma 7.2.** Let \( n, k, d, S, \) and \( s \) be as in the hypothesis of Lemma 7.1. If \( s \geq 4 \) and \( k \) satisfies either of the inequalities

\[
\begin{align*}
& \quad k \leq \frac{1}{2} \left( s - \frac{1}{\alpha} \sqrt{s(s-4)} \right) \\
& \quad k \geq \frac{1}{2} \left( s + \frac{1}{\alpha} \sqrt{s(s-4)} \right),
\end{align*}
\]

where \( \alpha = \frac{20}{103} \), then

\[
\frac{e^{\frac{s}{2\pi}}}{\sqrt{2\pi}} \left( \frac{s}{k(s-k)} \right)^{\frac{1}{2}} \leq \frac{1}{2}.
\]

**Proof.** Since \( s \geq 4 \), we have \( \frac{1}{12s} \leq \frac{1}{48} \), so \( e^{\frac{1}{12s}} \leq e^{\frac{1}{48}} \leq 1.03 \). Also, the value of \( \sqrt{\frac{1}{2\pi}} \) is less than 0.4. Thus it suffices to show that

\[
\frac{1.03}{0.4} \left( \frac{s}{k(s-k)} \right)^{\frac{1}{2}} \leq \frac{1}{2},
\]

or equivalently,

\[
\frac{s}{k(s-k)} \leq \left( \frac{20}{103} \right)^2 = \alpha^2.
\]

We can rearrange this inequality so that it looks like the more familiar quadratic inequality in the variable \( k \),

\[
-\alpha^2k^2 + \alpha^2ks - s \geq 0.
\]

The discriminant of this quadratic polynomial is \( \alpha^2s^2 - 4(-\alpha^2)(-s) \), which equals \( \alpha^2s(s-4) \). Since \( s \geq 4 \) by hypothesis, the polynomial has real roots. Since this polynomial is convex, the inequality is satisfied when \( k \) lies outside the open interval whose endpoints are the two (possibly equal) roots. The quadratic formula proves that the roots of the polynomial are

\[
\frac{s}{2} \pm \frac{1}{2\alpha^2} \sqrt{\alpha^2s(s-4)}.
\]

We conclude that the upper bound in the conclusion of the lemma is satisfied if \( k \) is outside the interval

\[
\left( \frac{1}{2} \left( s - \frac{1}{\alpha} \sqrt{s(s-4)} \right), \frac{1}{2} \left( s + \frac{1}{\alpha} \sqrt{s(s-4)} \right) \right).
\]
If the probability of a Bernoulli trial is less than one half, this provides an upper bound on the binary entropy of the probability.

**Lemma 7.3.** Let \( n, k, d, S, \) and \( s \) be as in the hypothesis of Lemma 7.2. Suppose \( W \leftarrow W_n^* \) and let \( A \) denote the Bernoulli random variable for the event \(|S \cap W| = 1\), conditioned on the event \( D = d\). We have

\[
H_b(A) \leq -2e^{1/12} \left( \frac{s}{k(s-k)} \right)^{1/2} \log \left( \frac{s}{k(s-k)} \right)^{1/2}.
\]

**Proof.** If \( p \) is at most \( q \) and \( q \leq 1/2 \), then \( H_b(p) \leq -2q \log q \). Apply this fact with the previous two lemmas, choosing \( q \) to be \( e^{1/12} \left( \frac{s}{k(s-k)} \right)^{1/2} \).

**Lemma 7.4.** Let \( n, d, S, \) and \( s \) be as in the hypothesis of the previous lemma. For any \( k \in [2^n] \), if

\[
p^k = \Pr_{W \leftarrow W_n^*} [|S \cap W| = k | D = d]
\]

and

\[
p^{\geq k} = \Pr_{W \leftarrow W_n^*} [|S \cap W| \geq k | D = d],
\]

then

\[
p^{\geq k} = \sum_{i=k}^{s} p^i.
\]

**Proof.** The first equality holds because

\[
p^{\geq k} = \Pr_{W \leftarrow W_n^*} [|S \cap W| \geq k | D = d]
\]

\[
= \Pr_{W \leftarrow W_n^*} \left[ \sum_{i=k}^{s} |S \cap W| = i | D = d \right]
\]

\[
= \sum_{i=k}^{s} \Pr_{W \leftarrow W_n^*} [|S \cap W| = i | D = d]
\]

\[
= \sum_{i=k}^{s} p^i,
\]

where the third equality holds because the events \(|S \cap W| = i\) and \(|S \cap W| = j\) are mutually exclusive for any distinct \( i \) and \( j \) in \([2^n]\).

**8 Todo**

Does the lower bound for adaptive \( Q_1 \) witness-finding algorithms with high probability of success [Corollary 5.8] hold for any \( Q_k \) witness-finding algorithm, for any fixed positive integer \( k \)? (It certainly doesn’t hold when \( k = 0 \), because of Theorem 4.1.) This question may be answered by adapting the proof by probabilistic method given in [1].
References


