Optimal Randomized Group Testing Algorithm to Determine the Number of Defectives

Nader H. Bshouty  
Department of Computer Science,  
Technion, Haifa, Israel

Catherine A. Haddad-Zaknoon  
Department of Computer Science,  
Technion, Haifa, Israel

Raghd Boulos  
The Arab Orthodox College, Haifa, Israel

Foad Moalem  
The Arab Orthodox College, Haifa, Israel

Jalal Nada  
The Arab Orthodox College, Haifa, Israel

Elias Noufi  
The Arab Orthodox College, Haifa, Israel

Yara Zaknoon  
The Arab Orthodox College, Haifa, Israel

Abstract

We study the problem of determining the exact number of defective items in an adaptive group testing by using a minimum number of tests. We improve the existing algorithm and prove a lower bound that shows that the number of tests in our algorithm is optimal up to small additive terms.

2012 ACM Subject Classification  Mathematics of computing; Mathematics of computing → Discrete mathematics; Mathematics of computing → Probabilistic algorithms; Theory of computation → Probabilistic computation

Keywords and phrases Group Testing, Randomized Algorithm

1 Introduction

Let $X$ be a set of items that contains some defective items $I \subseteq X$. In group testing, we test a subset $Q \subseteq X$ of items. An answer to the test is 1 if $Q$ contains at least one defective item, i.e., $Q \cap I \neq \emptyset$, and 0 otherwise. Group testing was initially introduced as a potential approach to the economical mass blood testing, [15]. However, it has been proven to be applicable in a variety of problems, including DNA library screening, [26], quality control in product testing, [30], searching files in storage systems, [22], sequential screening of experimental variables, [24], efficient contention resolution algorithms for multiple-access communication, [22, 34], data compression, [20], and computation in the data stream model, [12]. See a brief history and other applications in [11, 16, 17, 21, 25, 26] and references therein.

Estimating or determining exactly the number of defective items is an important problem in biological and medical applications [4, 31]. For example it is used to estimate the proportion of organisms capable of transmitting the aster-yellows virus in a natural population of leafhoppers [32], estimating the infection rate of the yellow-fever virus in a mosquito population [33] and estimating the prevalence of a rare disease using grouped samples to preserve individual anonymity [23].

In adaptive algorithms, the tests can depend on the answers of the previous ones. In non-adaptive algorithms, they are independent of the previous ones and; therefore, all tests can be done in one parallel step.

In this paper, we study the problem of determining exactly the number of defective items with adaptive group testing algorithms. We first give an algorithm that improves the number of tests in the best-known algorithm by a factor of 4. Improving constant factors in Group testing algorithms is one of the utmost important challenges in group testing since, in many
applications, tests are incredibly costly and time-consuming [3, 6, 9, 17, 18, 27]. Moreover, we give a lower bound that shows that our algorithm is optimal up to a small additive term. To the best of our knowledge, this is the first non-trivial lower bound for this problem.

1.1 Previous and New Results

Let \( X \) be a set of \( n \) items with \( d \) defective items \( I \). All the algorithms in this paper are adaptive. That is, the tests can depend on the answers to the previous ones. All the non-adaptive algorithms for determining exactly the number of defective items must ask at least \( \Omega(\frac{d^2}{\log d} \log n) \) queries and \( \Omega(\log n / \log \log n) \) for estimating their number, [1, 13, 14]. In [2], Bshouty et al. show that any deterministic or Las Vegas adaptive algorithm must ask at least \( \Omega(d \log(n/d)) \) queries. Since the query complexity depends on the number of items \( n \), which, for most applications, is extremely large, non-adaptive algorithms and Las Vegas (and deterministic) algorithms are not desirable for solving this problem.

In [5], Cheng gave a randomized Monte Carlo adaptive algorithm that for any constant \( c \), asks \( 4dc \log d \) queries\(^1\) and, with probability at least \( 1 - \delta = 1 - 1/d^{c-1} \), determines exactly the number of defective items. His algorithm, with the technique used in this paper\(^2\), gives a randomized Monte Carlo algorithm that asks \( 4d \log(d/\delta) \) queries with success probability at least \( 1 - \delta \) for any \( \delta \).

In this paper, we first give lower bounds for the number of queries. The first lower bound is \( d \log(1/d\delta) \) for any \( n, d \) and \( \delta > 1/(2(n - d + 1)) \). See Theorem 4 in Section 3. This shows that Cheng’s algorithm is almost optimal (up to a multiplicative factor of 4 and an additive term of \( 4d \log d \)). For \( \delta < 1/(2(n - d + 1)) \), we give the tight bound \( d \log(n/d) \), which, in particular, is the number of tests required for any deterministic algorithm. This bound matches the tight bound for finding all the defective items. We also give better lower bound of \( d \log(1/\delta) \) for any large enough\(^3\) \( n \). See Theorem 5 in Section 3.

Moreover, we give a new randomized Monte Carlo algorithm that asks \( d \log(d/\delta) \) queries. See Theorem 7 in Section 4. Our algorithm improves Cheng algorithm by a multiplicative factor of 4 and is optimal up to an additive term of \( d \log d \). Notice that, for \( \delta = 1/d^{c(1)} \) (especially when \( \delta \) depends on \( n \)), our algorithm is optimal up to a small additive term.

Estimating the number of defective items is studied in [2, 10, 13, 14, 19, 28]. The problem is to find an integer \( D \) such that \( d \leq D \leq (1 + \epsilon)d \). In [2], Bshouty et al. modified Falhatgar et al. algorithm, [19], and gave a randomized algorithm that makes expected number of \((1 - \delta) \log \log d + O((1/\epsilon^2) \log(1/\delta)) \) tests. They also prove the lower bound \((1 - \delta) \log \log d + \Omega((1/\epsilon) \log(1/\delta)) \).

2 Definitions and Preliminary Results

In this section we give some notations, definitions, the type of algorithms that are used in the literature and some preliminary results.

\(^1\) All the log in this paper are \( \log_2 \) and all the complexities in this introduction are multiplied by \( 1 + o(1) \) where the \( o(1) \) is with respect to \( d \).

\(^2\) First estimate \( d \) using the algorithm in this paper. Then determine \( c \) to get success \( \delta \) and run his algorithm

\(^3\) \( n \geq d^{c(1)} \) where \( \omega(1) \) is with respect to \( d \) for example \( \log^* d \)
2.1 Notations and Definitions

Let $X = [n] := \{1, 2, 3, \ldots, n\}$ be a set of items with some defective items $I \subseteq [n]$. In group testing, we query a subset $Q \subseteq X$ of items and the answer to the query is $Q(I) := 1$ if $Q$ contains at least one defective item, i.e., $Q \cap I \neq \emptyset$, and $Q(I) := 0$, otherwise.

Let $I \subseteq [n]$ be the set of defective items. Let $O_I$ be an oracle that for a query $Q \subseteq [n]$ returns $Q(I)$. Let $A$ be an algorithm that has access to an oracle $O_I$. The output of the algorithm with an access to an oracle $O_I$ is denoted by $A(O_I)$. When the algorithm is randomized, then we add the random seed $r$, and then the output of the algorithm is a random variable $A(O_I, r)$ in $[n]$. When $I$ is known from the context, we just write $A(r)$. Let $A$ be a randomized algorithm and let $r_0$ be a fixed seed. Then $A(r_0)$ is a deterministic algorithm that is equivalent to the algorithm $A$ with the fixed seed $r_0$. We denote by $Q(A, O_I)$ (or $Q(A, O_I, r)$) the set of queries that $A$ asks with oracle $O_I$ (and a seed $r$). We say that the algorithm determines $|I| = d$ exactly if $A(O_I, r) = |I|$.

2.2 Types of Algorithms

In this paper we consider four types of algorithms whose running time is polynomial in $n$.

1. The deterministic algorithm $A$ with an oracle $O_I$, $I \subseteq X$. The query complexity of a deterministic algorithm $A$ is the worst case complexity, i.e., $\max_{|I|=d} |Q(A, O_I)|$.

2. The randomized Las Vegas algorithm. We say that a randomized algorithm $A(r)$ is a randomized Las Vegas algorithm that has expected query complexity $g(d)$ if for any $I \subseteq X$, $A(r)$ with an oracle $O_I$ asks $g(|I|)$ expected number of queries and with probability 1 outputs $|I|$.

3. The randomized Monte Carlo algorithm. We say that a randomized algorithm $A(r)$ is a randomized Monte Carlo algorithm that has query complexity $g(d, \delta)$ if for any $I \subseteq X$, $A$ with an oracle $O_I$ asks at most $g(|I|, \delta)$ queries and with probability at least $1 - \delta$ outputs $|I|$.

4. The randomized Monte Carlo algorithm with average case complexity. We say that a randomized algorithm $A(r)$ is Monte Carlo algorithm with average case complexity that has expected query complexity $g(d, \delta)$ if for any $I \subseteq X$, $A$ asks $g(|I|, \delta)$ expected number of queries and with probability at least $1 - \delta$ outputs $|I|$.

2.3 Preliminary Results

We now prove a few results that will be used throughout the paper.

Let $s \in \bigcup_{n=0}^{\infty} \{0, 1\}^n$ be a string over $\{0, 1\}$ (including the empty string $\lambda \in \{0, 1\}^0$). We denote by $|s|$ the length of $s$, i.e., the integer $m$ such that $s \in \{0, 1\}^m$. Let $s_1, s_2 \in \bigcup_{n=0}^{\infty} \{0, 1\}^n$ be two strings over $\{0, 1\}$ of length $m_1$ and $m_2$, respectively. We say that $s_1$ is a prefix of $s_2$ if $m_1 \leq m_2$ and $s_{1,i} = s_{2,i}$ for all $i = 1, \ldots, m_1$. We denote by $s_1 \cdot s_2$ the concatenation of the two strings.

The following lemma is proved in [1].

**Lemma 1.** Let $S = \{s_1, \ldots, s_N\}$ be a set of $N$ distinct strings such that no string is a prefix of another. Then

$$\max_{s \in S} |s| \geq E(S) := E_{s \in S} |s| \geq \log N.$$

**Lemma 2.** Let $A$ be a deterministic adaptive algorithm that asks queries. If $A(O_I) \neq A(O_J)$ then there is $Q_0 \in Q(A, O_I) \cap Q(A, O_J)$ such that $Q_0(I) \neq Q_0(J)$.

In particular, if, in addition, $I \subseteq J$ then $Q_0(I) = 0$ and $Q_0(J) = 1$. 

SWAT 2020
18:4 Algorithm to Determine the Number of Defectives

Proof. Since algorithm $A$ is deterministic, in the execution of $A$ with $O_I$ and $O_J$, $A$ asks the same queries as long as it gets the same answers to the queries. Since $A(O_I) \neq A(O_J)$, there must be a query $Q_0$ that is asked to both $O_I$ and $O_J$ for which $Q_0(I) \neq Q_0(J)$. □

Lemma 3. Let $A$ be a deterministic adaptive algorithm that asks queries. Let $C \subseteq 2^{[n]} = \{I|I \subseteq [n]\}$. If for every two distinct $I_1$ and $I_2$ in $C$ there is a query $Q \in Q(A,O_{I_1})$ such that $Q(I_1) \neq Q(I_2)$ then

$$\max_{I \in C}|Q(A,O_I)| \geq E_{I \in C}|Q(A,O_I)| \geq \log |C|.$$

That is, the query complexity of $A$ is at least $\log |C|$.

Proof. For $I \in C$ consider the sequence of the queries that $A$ with the oracle $O_I$ asks and let $s(I) \in \mathbb{Z}_{\geq 0}^n$ be the sequence of answers. The query complexity and average-case complexity of $A$ is $s(C) := \max_{I \in C}|s(I)|$ and $\bar{s}(C) := \mathbb{E}_{I \in C}|s(I)|$ where $|s(I)|$ is the length of $s(I)$. We show that for every two distinct $I_1$ and $I_2$ in $C$, $s(I_1) \neq s(I_2)$ and $s(I_1)$ is not a prefix of $s(I_2)$. This implies that $\{s(I) | I \in C\}$ contains $|C|$ distinct strings such that no string is a prefix of another. Then by Lemma 1, the result follows.

Consider two distinct sets $I_1,I_2 \subseteq [n]$. There is a query $Q_0 \in Q(A,O_{I_1})$ such that $Q_0(I_1) \neq Q_0(I_2)$. Consider the execution of algorithm $A$ with both $O_{I_1}$ and $O_{I_2}$, respectively. Since $A$ is deterministic, as long as the answers of the queries are the same both ($A$ with $O_{I_1}$ and $A$ with $O_{I_2}$) continue to ask the same query. Then, either we get to the query $Q_0$ in both execution and then $Q_0(I_1) \neq Q_0(I_2)$, or we reach some other query $Q'$, that is asked before $Q_0$, satisfies $Q'(I_1) \neq Q'(I_2)$. In both cases, $s(I_1) \neq s(I_2)$ and $s(I_1)$ is not a prefix of $s(I_2)$. □

3 Lower Bounds

In this section, we prove some lower bounds for the number of queries that are needed to determine exactly the number of defective items with a Monte Carlo algorithm.

Theorem 4. Let $\delta \geq 1/(2(n - d + 1))$. Let $A$ be a randomized Monte Carlo adaptive algorithm that for any set of defective items $I$ of size $|I| \in \{d,d + 1\}$, with probability at least $1 - \delta$, determines exactly the number of defective items $|I|$. Algorithm $A$ must ask at least

$$d \log \frac{1}{2d\delta} - 1$$

queries.

In particular, when $\delta \leq 1/(2(n - d + 1))$ then $A$ must ask at least $d \log(n/d) - O(d)$ queries which is the query complexity (up to additive term $O(d)$) of finding the defective items (and therefore, in particular, finding $|I|$) with $\delta = 0$ error.

Proof. Let $A(O_I,r)$ be a randomized Monte Carlo algorithm that for $|I| \in \{d,d + 1\}$, determines $|I|$ with probability at least $1 - \delta$ where $r$ is the random seed of the algorithm. Let $X(I,r)$ be a random variable that is equal to 1 if $A(O_I,r) \neq |I|$ and 0 otherwise. Then, for any $I \subseteq [n]$, $E_r[X(I,r)] \leq \delta$. Let $m = \lfloor 1/(2\delta) \rfloor + d - 1 \leq n$. Consider any $J \subseteq [m]$, $|J| = d$. For any such $J$, let

$$Y_J(r) = X(J,r) + \sum_{i \in [m] \setminus J} X(J \cup \{i\},r).$$
Then, for every $J \subseteq [m]$ of size $d$, $E_r[Y_J(r)] \leq (m - d + 1)\delta \leq \frac{1}{2}$. Therefore, for a random uniform $J \subseteq [m]$ of size $d$, we have $E_r[E_J[Y_J(r)]] = E_J[E_r[Y_J(r)]] \leq 1/2$. Thus, there is $r_0$ such that for at least half of the sets $J \subseteq [m]$, of size $d$, $Y_J(r_0) = 0$. Let $C$ be the set of all $J \subseteq [m]$, of size $d$, such that $Y_J(r_0) = 0$. Then

$$|C| \geq \frac{1}{2} \left( \frac{m}{d} \right) = \frac{1}{2} \left( \frac{1}{2d\delta} + d - 1 \right).$$

Consider the deterministic algorithm $A(r_0)$. We now claim that for every two distinct $J_1, J_2 \subseteq C$, there is a query $Q_0 \in Q(A(r_0), O_{J_1})$ such that $Q_0(J_1) \neq Q_0(J_2)$. If this is true then, by Lemma 3, the query complexity of $A(r_0)$ is at least

$$\log |C| \geq \log \left( \frac{1}{2} \left( \frac{1}{2d\delta} + d - 1 \right) \right) \geq d \log \frac{1}{2d\delta} - 1.$$

Proof. We now prove the claim. Consider two distinct $J_1, J_2 \subseteq C$. There is w.l.o.g $j \in J_2 \setminus J_1$. Since $Y_{J_2}(r_0) = 0$, we have $X(J_1, r_0) = 0$ and $X(J_1 \cup \{j\}, r_0) = 0$ and, therefore, $A(O_{J_1}, r_0) = d$ and $A(O_{J_1 \cup \{j\}}, r_0) = d + 1$. Thus, by Lemma 2, there is a query $Q_0 \in Q(A(r_0), O_{J_1}) \cap Q(A(r_0), O_{J_1 \cup \{j\}})$ for which $Q_0(J_1) = 0$ and $Q_0(J_1 \cup \{j\}) = 1$. Therefore, $Q_0(\{j\}) = 1$ and then $Q_0(J_1) = 0$ and $Q_0(J_2) = 1$.

In the Appendix, we prove this lower bound for any randomized Monte Carlo algorithm with average-case complexity.

For large enough\(^4\) $n$, $n = d^{\omega(1)}$, the following result gives a better lower bound.

\[\textbf{Theorem 5.} \text{ Any randomized Monte Carlo adaptive algorithm that with probability at least } 1 - \delta \text{ determines the number of defectives must ask at least } \left( 1 - \frac{\log d + \log(1/\delta) + 1}{\log n + \log(1/\delta)} \right) d \log \frac{1}{2\delta} \text{ queries.} \]

\[\text{In particular, when } n = d^{\omega(1)} \text{ then the number of queries is at least } \left( 1 - o(1) \right) d \log \frac{1}{2\delta}. \]

Proof. Let $A(r)$ be a randomized Monte Carlo algorithm that determines the number of defective items with probability at least $1 - \delta$ where $r$ is the random seed of the algorithm. Let $X'(I, r)$ be a random variable that is equal to 1 if $A(O_I, r) \neq |I|$ and 0 otherwise. Then, for every $I$, $E_r[X'(I, r)] \leq \delta$. For every set $I$ and $i \in [n] \setminus I$, let $X(I, i, r) = X'(I, r) + X'(I \cup \{i\}, r)$. Then, for every $I \subseteq [n]$ and $i \in [n] \setminus I$, $E_r[X(I, i, r)] \leq 2\delta$. For $I$ of size $d$ chosen uniformly at random and $i \in [n] \setminus I$ chosen uniformly at random, we have $E_r[E_r[E_r[X(I, i, r)]] = E_r[E_r[X(I, i, r)]] \leq 2\delta$. Therefore, there exists a seed $r_0$ such that $E_r[E_r[X(I, i, r_0)] \leq 2\delta$. We now choose $q$ permutations $\phi_1, \ldots, \phi_q : [n] \to [n]$ uniformly and independently at random where

$$q = \left\lceil \frac{1 + \log n}{\log \frac{1}{2\delta}} \right\rceil.$$

\(^4\) The $\omega(1)$ is with respect to $d$. For example, $n > d^{\omega(1)}$.\[SWAT 2020\]
Then, for any \( I \) and \( i \in [n]\setminus I \), \( \phi_1(I), \ldots, \phi_q(I) \) are uniform and independent random sets of size \( d \) and \( \phi_1(i), \ldots, \phi_q(i) \) are uniform and independent random integers where \( \phi_j(i) \not\in \phi_j(I) \) for all \( j \in [q] \). Hence,

\[
E_{(\phi_j)_j} \left[ \prod_{j=1}^q X(\phi_j(I), \phi_j(i), r_0) \right] = \prod_{j=1}^q E_{\phi_j} [X(\phi_j(I), \phi_j(i), r_0)] \leq (2\delta)^q
\]

and,

\[
E_{(\phi_j)_j, E_I} \left[ \prod_{j=1}^q X(\phi_j(I), \phi_j(i), r_0) \right] = E_{E_I} E_{(\phi_j)_j} \left[ \prod_{j=1}^q X(\phi_j(I), \phi_j(i), r_0) \right] \\
\leq (2\delta)^q.
\]

Therefore,

\[
E_{(\phi_j)_j, E_I} \left[ \sum_{i \in [n]\setminus I} \prod_{j=1}^q X(\phi_j(I), \phi_j(i), r_0) \right] \leq (n-d)(2\delta)^q < \frac{1}{2}.
\]

Thus, there are permutations \( \{\phi'_j\}_{j \in [q]} \) such that

\[
E_I \left[ \sum_{i \in [n]\setminus I} \prod_{j=1}^q X(\phi'_j(I), \phi'_j(i), r_0) \right] < \frac{1}{2}
\]

Since \( X \) takes values in \( \{0, 1\} \), this implies that for at least half of the sets \( I \subseteq [n] \), of size \( d \), and all \( i \in [n]\setminus I \), there exists \( j \in [q] \) such that \( X(\phi'_j(I), \phi'_j(i), r_0) = 0 \). Let \( \mathcal{C} \) be the class of such sets \( I \) for which the later statement is true. Then,

\[
|\mathcal{C}| \geq \frac{1}{2} \left( \frac{n}{d} \right) - \frac{1}{2}
\]

and

\[
(\forall I \in \mathcal{C})(\forall i \in [n]\setminus I)(\exists j \in [q]) X(\phi'_j(I), \phi'_j(i), r_0) = 0. \tag{1}
\]

Consider the following deterministic algorithm \( A^* \):

**Algorithm \( A^* \)**

For \( j = 1, \ldots, q \)
- Run \( A(r_0) \) with oracle \( O_I \)
- If \( A(r_0) \) asks \( Q \) then ask the query \( \phi'^{-1}_j(Q) \).

First notice that, if algorithm \( A(r_0) \) has query complexity \( M \), then \( A^* \) has query complexity at most \( qM \).

Since \( \phi'^{-1}_j(Q) \cap I = \emptyset \) if and only if \( Q \cap \phi'_j(I) = \emptyset \), at iteration \( j \), the algorithm \( A(r_0) \) runs as if the defective items are \( \phi'_j(I) \). Therefore, at iteration \( j \), the queries that are asked by \( A(r_0) \) are \( Q(A(r_0), O_{\phi'_j(I)}) \) and the queries that are asked by \( A^* \) are \( \phi'^{-1}_j(Q(A(r_0), O_{\phi'_j(I)})) \).

Hence,

\[
Q(A^*, O_I) = \bigcup_{j=1}^q \phi'^{-1}_j(Q(A(r_0), O_{\phi'_j(I)})). \tag{2}
\]

We now show that,
Claim 6. For every two distinct sets $I_1, I_2 \in C$ there is a query $Q' \in \mathcal{Q}(A^*, \mathcal{O}_I)$ that gives different answers for $I_1$ and $I_2$.

If this Claim is true then, by Lemma 3, the query complexity $qM$ of $A^*$ is at least $\log |C|$ and then, since $\binom{n}{d} \geq (n/d)^d$,

$$M \geq \frac{\log |C|}{q} \geq \frac{\log \frac{1}{d} \binom{n}{d}}{1 + \log \frac{1}{d} \log 1/(1/\delta)} + 1$$

$$\geq \frac{\log n - \log d - 1}{\log n + \log \frac{1}{\delta}} \cdot d \log \frac{1}{2\delta}$$

$$= \left(1 - \frac{\log d + \log(1/\delta) + 1}{\log n + \log(1/\delta)}\right) d \log \frac{1}{2\delta}.$$

Proof. We now prove the claim. Let $I_1, I_2 \in C$ be two distinct sets of size $d$. Then there is $i_0 \in I_2 \setminus I_1$. By (1) there is $j_0 \in [q]$ such that for $\phi := \phi^{j_0}$, $X(\phi(I_1), \phi(i_0), r_0) = 0$. Therefore $A(\mathcal{O}_{\phi(I_1)}, r_0) = |\phi(I_1)| = |I_1|$ and

$$A(\mathcal{O}_{\phi(I_1) \cup \phi(i_0)}, r_0) = |\phi(I_1) \cup \{\phi(i_0)\}| = |I_1| + 1.$$ 

Therefore, by Lemma 2, there exists a query $Q_0 \in \mathcal{Q}(A(r_0), \mathcal{O}_{\phi(I_1)})$ that satisfies $Q_0(\phi(I_1)) = 0$ and $Q_0(\phi(I_1) \cup \{\phi(i_0)\}) = 1$. That is, $Q_0 \cap \phi(I_1) = \emptyset$ and $Q_0 \cap (\phi(I_1) \cup \{\phi(i_0)\}) \neq \emptyset$. This implies that $\phi(i_0) \in Q_0$. Since $\phi(i_0) \in \phi(I_2)$, we get that $Q_0 \cap \phi(I_2) = \emptyset$ and $Q_0 \cap \phi(I_2) \neq \emptyset$. Thus, $\phi^{-1}(Q_0) \cap I_1 = \emptyset$ and $\phi^{-1}(Q_0) \cap I_2 \neq \emptyset$. That is, the query $Q' := \phi^{-1}(Q_0)$ satisfies

$$Q'(I_1) \neq Q'(I_2).$$

Since $Q_0 \in \mathcal{Q}(A(r_0), \mathcal{O}_{\phi(I_1)})$, by (2), we have (recall that $\phi := \phi^{j_0}$)

$$Q' = \phi^{-1}(Q_0) \in \phi^{-1}(\mathcal{Q}(A(r_0), \mathcal{O}_{\phi(I_1)})) \subseteq \mathcal{Q}(A^*, \mathcal{O}_I)$$

and therefore, $Q' \in \mathcal{Q}(A^*, \mathcal{O}_I)$. This completes the proof of the claim.

4 Upper Bound

In this section we prove:

**Theorem 7.** There is a Monte Carlo adaptive algorithm that asks

$$d \log \frac{d}{\delta} + O \left( d \log d \log \frac{1}{\delta} \right) = (1 + o(1))d \log \frac{d}{\delta}$$

queries and, with probability at least $1 - \delta$, finds the number of defective items.

This improves the bound $4d \log(d/\delta)$ achieved in [5]. By Theorem 5, this bound is optimal up to the additive term $(1 + o(1))d \log d$.

We will use the following.

**Lemma 8 ([7, 8, 29]).** There is a deterministic algorithm, **Find-Defectives**, that without knowing $d$, asks $d \log(n/d) + O(d)$ queries and finds the defective items.

Our algorithm, at the first stage, calls the procedure **Estimate** that, with probability at least $1 - \delta/2$ finds an estimate $D$ of the number defective items where $d \leq D \leq 8d$. This procedure makes $O(d + \log d \log(1/\delta))$ queries.
At the second stage, it uniformly at random partitions the set of items \([n]\) into \(t = D^2/\delta\) disjoint sets \(B_1, B_2, \ldots, B_t\). We show that, with probability at least \(1 - \delta/2\), each set contains at most one defective item. We call a set that contains a defective item a **defective set**. Therefore, with probability at least \(1 - \delta/2\), the number of defective items is equal to the number of defective sets. Then, we treat each set \(B_i\) as one item \(i\) and call the algorithm **Find-Defectives** in Lemma 8 on \(t\) items to find the defective sets. To do that, each test \(Q \subseteq [t]\) in **Find-Defectives** is simulated by the query \(S(Q) := \bigcup_{i \in Q} B_i\) in our algorithm.

Obviously, the set \(Q\) contains an index of a defective set if and only if \(S(Q)\) contains a defective item. Therefore, the algorithm will return the number of defective sets which, with probability at least \(1 - \delta/2\), is equal to the number of defective items. By Lemma 8, the number of queries asked in the second stage is

\[
d \log \frac{t}{d} + O(d) = d \log \frac{D^2/\delta}{d} + O(d) = d \log \frac{d}{\delta} + O(d).
\]

We now prove:

**Lemma 9.** There is a Monte Carlo adaptive algorithm **Estimate** that asks

\[
O\left(d + \log d \log \frac{1}{\delta}\right)
\]

queries and returns an integer \(D\) that, with probability at least \(1 - \delta\), \(D \geq d\) and, with probability \(1\), \(D \leq 8d\).

**Proof.** The algorithm is in Figure 1. For each \(k = 2^i\), the algorithm does the following \(t = \lceil 2 \log(1/\delta)/k \rceil\) times: uniformly at random divides the items into \(k\) mutually disjoint sets and then tests each set and counts (in the variable “count”) the number of sets that contain at least one defective item. First, notice that, for \(k \leq d\),

\[
\Pr[\text{count} < k/4] \leq \left(\frac{k}{k/4}\right)^d \left(\frac{1}{4}\right) \leq 2^k \left(\frac{1}{4}\right)^d \leq 2^{-d}.
\]

The probability that the algorithm outputs \(k < d\) is the probability that the event “count < \(k/4\)” happens \(t = \lceil 2 \log(1/\delta)/k \rceil\) times for some \(k = 2^i < d\). This is at most

\[
\sum_{i=1}^{\lceil \log d \rceil} 2^i \left(\frac{2 \log(1/\delta)/2^i}{2^i}\right)^d \leq \sum_{i=1}^{\lceil \log d \rceil} 2^i (1/\delta)^d \leq \delta.
\]

Therefore, with probability at least \(1 - \delta\), the output is greater or equal to \(d\). Since “count” is always less than or equal to the number of defective items \(d\), when \(4d < k \leq 8d\), we have count \(\leq d < k/4\) and the algorithm halts. Therefore, the output cannot be greater than \(8d\).

The number of queries that the algorithm asks is at most

\[
\sum_{i=1}^{\lceil \log 8d \rceil} 2^i \left(\frac{2 \log(1/\delta)/2^i}{2^i}\right)^d \leq \sum_{i=1}^{\lceil \log 8d \rceil} 2 \log(1/\delta) + 2^i = O\left(d + \log d \log \frac{1}{\delta}\right).
\]

We now prove Theorem 7.

**Proof.** The algorithm is in Figure 2. First, we run the algorithm in Figure 1 that estimates \(d\) and returns \(D\) such that, with probability at least \(1 - \delta/2\), \(d \leq D \leq 8d\). Then, the algorithm chooses a function \(f : [n] \to [N]\) uniformly at random where \(N = \lceil D^2/\delta \rceil\). This is equivalent
Estimate($n, \delta$)
1) For $k = 2^i, i = 1, 2, 3, \ldots$ do
2) For $m = 1$ to $t := \lceil 2 \log(1/\delta) \rceil$
3) Choose a function $f_m : [n] \rightarrow [k]$ uniformly at random
4) count $\leftarrow 0$
5) For $j = 1$ to $k$
6) ask the query $Q := f_m^{-1}(j)$
7) If the answer is 1 then count $\leftarrow$ count + 1
8) EndFor
9) If (count $\geq k/4$) Break (leave the For loop)
10) EndFor
11) If (count $\leq k/4$) Output($k$) and halt
12) EndFor

Figure 1 An algorithm that estimates $d$.

to uniformly at random divide the items into $N$ mutually disjoint sets $Q_i, i = 1, \ldots, N$. The probability that some $Q_i$ contains two defective items is

$$\Pr[(\exists i) \text{ } Q_i \text{ contains two defective items}] = 1 - \prod_{i=1}^{d-1} \left(1 - \frac{i}{N}\right) \leq \sum_{i=1}^{d-1} \frac{i}{N} \leq \frac{d^2}{2N} \leq \frac{\delta}{2}.$$ 

Then, the algorithm runs Find-Defectives in Lemma 8 on the $N$ disjoint sets $Q_1, \ldots, Q_N$ to find the number of sets that contain a defective item. This number is, with probability at least $1 - \delta/2$, equal to the number of defective items. Therefore, with probability at least $1 - \delta$, Find-Defectives finds $d$. This completes the proof of correctness.

The query complexity is the query complexity of Estimate($n, \delta/2$) and Find-Defectives with $N$ items. This, by Lemma 8 and 9, is equal to

$$d \log \frac{N}{d} + O(d) + O \left( d + \log d \log \frac{1}{\delta}\right) = d \log \frac{d}{\delta} + O \left( d + \log d \log \frac{1}{\delta}\right).$$

Figure 2 An algorithm that finds $d$. 

Find-d($n, \delta$)
1) $D \leftarrow$ Estimate($n, \delta/2$).
2) $N \leftarrow \lceil D^2/\delta \rceil$.
3) Choose a function $f : [n] \rightarrow [N]$ uniformly at random.
4) Define $Q_i = f^{-1}(i)$ for $i = 1, 2, \ldots, N$.
5) Run Find-Defectives with $N$ items.
6) and for each query $Q$ ask $\cup_{j \in Q} Q_j$.
7) Let $\Delta$ be the output of Find-Defectives.
8) Output $\Delta$. 

SWAT 2020
References


19 Moein Falahatgar, Ashkan Jafarpour, Alon Orlitsky, Venkataadheeraj Pichapati, and Ananda Theertha Suresh. Estimating the number of defectives with group testing. In
Theorem 10. Let $1/d^{o(1)} \geq \delta \geq 1/(2(n - d + 1))$. Let $A$ be a randomized adaptive algorithm that for any set of defective items $I$ of size $d$ or $d + 1$, with probability at least $1 - \delta$, exactly determines the number of defective items $|I|$. Algorithm $A$ must ask at least
\[(1 - o(1))d \log \frac{1}{\delta}\]
expected number of queries.

When $\delta < 1/(2(n - d + 1))$ then $A$ must ask at least $(1 - o(1))d \log n$ queries which is, asymptotically, the query complexity of finding the defective items with $\delta = 0$ error.

Proof. Let $A(r)$ be a randomized algorithm that for $I \subseteq [n]$, $|I| \in \{d, d + 1\}$ and oracle $O_I$, determines $|I|$ with probability at least $1 - \delta$ where $r$ is the random seed of the algorithm.
Algorithm to Determine the Number of Defectives

Let $X(I, r)$ be a random variable that is equal to 1 if $A(r, O_I) \neq |I|$ and 0 otherwise. Then for any $I \subseteq [n]$, $E_r[X(I, r)] \leq \delta$. Let $m = \lceil \tau/d \rceil + d - 1 \leq n$ where $\tau = 1/(d \log(1/(d\delta)))$. Consider any $J \subseteq [m]$, $|J| = d$. For any such $J$ let

$$Y_J(r) = X(J, r) + \sum_{i \in [m] \setminus J} X(J \cup \{i\}, r).$$

Then for every $J \subseteq [m]$ of size $d$, $E_r[Y_J(r)] \leq (m - d + 1)\delta \leq \tau$. Therefore for a random uniform $J \subseteq [m]$ of size $d$ we have $E_r[E_J[Y_J(r)]] = E_J[E_r[Y_J(r)]] \leq \tau$. Therefore, by Markov’s inequality, for $\eta = 1/(d \log(d\delta))$,

$$\Pr_r[E_J[Y_J(r)] > \eta] \leq \frac{\tau}{\eta} = \frac{1}{d\eta}.$$

That is, for random $r$, with probability at least $1 - 1/d$, at least $1 - \eta$ fraction of the sets $J \subseteq [m]$ of size $d$ satisfies $Y_J(r) = 0$. Let $R$ be the set of seeds $r$ such that at least $1 - \eta$ fraction of the sets $J \subseteq [m]$ of size $d$ satisfies $Y_J(r) = 0$. Then $\Pr_r[R] \geq 1 - 1/d$. Let $r_0 \in R$. Let $C_{r_0}$ be the set of all $J \subseteq [m]$ of size $d$ such that $Y_J(r_0) = 0$. Then

$$|C_{r_0}| \geq (1 - \eta) \left(\frac{m}{d}\right) = (1 - \eta) \left(\frac{\lceil \tau/d \rceil + d - 1}{d}\right).$$

Consider the deterministic algorithm $A(r_0)$. As in Theorem 4, for every two distinct $J_1, J_2 \subseteq C_{r_0}$, there is a query $Q \in Q(A(r_0), O_{J_1})$ such that $Q(J_1) \neq Q(J_2)$. Then by Lemma 3, the average-case query complexity of $A(r_0)$ is at least

$$\log |C_{r_0}| \geq \log(1 - \eta) \left(\frac{\lceil \tau/d \rceil + d - 1}{d}\right) \geq d \log \frac{\tau}{d\delta} - \log \frac{1}{1 - \eta}.$$

Let $Z(O_I, r) = |Q(A(r), O_I)|$. We have shown that for every $r \in R$,

$$E_{I \in C_r}[Z(O_I, r)] \geq d \log \frac{\tau}{d\delta} - \log \frac{1}{1 - \eta}.$$

Therefore for every $r \in R$,

$$E_I[Z(O_I, r)] \geq E_{I \in C_r}[Z(O_I, r)|I \in C_r]\Pr[I \in C_r]$$

$$\geq (1 - \eta) \left(\frac{d \log \frac{\tau}{d\delta} - \log \frac{1}{1 - \eta}}{d}\right).$$

Therefore

$$E_I E_r[Z(O_I, r)] = E_r E_I[Z(O_I, r)]$$

$$\geq E_r[E_I[Z(O_I, r)|r \in R]\Pr[r \in R]]$$

$$\geq \left(1 - \frac{1}{d}\right)(1 - \eta) \left(\frac{d \log \frac{\tau}{d\delta} - \log \frac{1}{1 - \eta}}{d}\right).$$

Therefore there is $I$ such that

$$E_r[Z(O_I, r)] \geq \left(1 - \frac{1}{d}\right)(1 - \eta) \left(\frac{d \log \frac{\tau}{d\delta} - \log \frac{1}{1 - \eta}}{d}\right)$$

and then

$$E_r[Z(O_I, r)] \geq (1 - o(1))d \log \frac{1}{\delta}. \quad \blacksquare$$