Scalable and Efficient Non-adaptive Deterministic Group Testing

Anonymous Author(s)
Affiliation
Address
email

Abstract

Group Testing (GT) is about learning a (hidden) subset K, of size k, of some large domain N, of size $n \gg k$, using a sequence of queries. A result of a query provides some information about the intersection of the query with the unknown set K. The goal is to design efficient (polynomial time) and scalable (polylogarithmic number of queries per element in K) algorithms for constructing queries that allow to decode *every* hidden set K based on the results of the queries. A vast majority of the previous work focused on randomized algorithms minimizing the number of queries; however, in case of large domains N, randomization may result in a significant deviation from the expected precision of learning the set K. Others assumed unlimited computational power (existential results) or adaptiveness of queries (next query could be constructed taking into account the results of the previous queries) – the former approach is less practical due to non-efficiency, and the latter has several drawbacks including non-parallelization. To avoid all the abovementioned drawbacks, for Quantitative Group Testing (QGT) where query result is the size of its intersection with the hidden set, we present *the first* efficient and scalable *non-adaptive deterministic* algorithms for constructing queries and decoding a hidden set K from the results of the queries – these solutions *do not* use any randomization, adaptiveness or unlimited computational power.

1 Introduction

In the Group Testing (GT) field, the goal is to identify all elements of an unknown set K by asking queries. Originally GT was applied to identifying infected individuals in large populations using pooled tests [24] which regained interest during COVID-19 pandemic [1, 46, 50]. Recently GT found applications in various areas of Machine Learning including approximating the nearest neighbor [26], simplifying multi-label classifiers [52] and accelerating forward pass of a deep neural network [45].

All we initially know about set K is that $|K| \leq k$, for some known parameter $k \leq n$, and that it is a subset of some much larger set N with $|N| = n$. The answer to a query Q depends on the intersection between K and Q and equals to $\mathsf{Result}(K \cap Q)$, where Result is some result function (also called feedback function in this paper).

The sequence of queries is a correct solution to Group Testing if and only if for any two different sets K_1, K_2 (satisfying some cardinality restriction), the sequence of answers for K_1 and K_2 is different. Note that although this allows to uniquely identify the hidden set K based on the results of the queries, in some cases decoding of set K could be a hard computational problem. The objective is: for a given deterministic feedback function $\mathsf{Result}(\cdot)$, to find a fixed sequence of queries that will identify any set K, and the length of this sequence, called the query complexity, is as short as possible. In particular, we are interested in solutions that have query complexity logarithmic in n and polynomial in k.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

α	Constructive algorithms	Lower bound
1	$O(k^2 \log n)$ [48]	$\Omega\left(k^2 \frac{\log n}{\log k}\right)$ [11]
k	$O(k^2 \log n)$ [48] $\widetilde{O}(k)$ **Thm 1** (for $\alpha = k$)	$\Omega\left(k \frac{\log n}{\log k}\right)$ (folklore)
$*$	$\widetilde{O}\left(\min\left\{\left(\frac{k}{\alpha}\right)^2, \frac{n}{\alpha}\right\} + k\right)$ (**Thm 1**)	$\Omega\left(\min\left\{\left(\frac{k}{\alpha}\right)^2, \frac{n}{\alpha}\right\} + k\frac{\log n}{\log k}\right)$ (**Thm 2**)

Table 1: Asymptotic bounds on query complexity of solutions to non-adaptive deterministic QGT with $\mathcal{F}_\alpha$ feedback. By constructive upper bound we mean query construction and decoding of hidden set in time $poly(n)$. Symbol $*$ stands for any valid value of the parameter $\alpha \in \{1, \ldots, k\}$, notations $\widetilde{O}$ and $\widetilde{\Omega}$ disregard polylogarithmic factors.

The most popular classical variant, present in the literature, considers function $\mathsf{Result}(\cdot)$ that answers whether the intersection between K and Q is empty or not, c.f., [25]; it is also known under the name of *beeping*. Another popular result function returns the intersection size; this variant has also been studied under the name of *coin weighting* [3, 22] and Quantitative Group Testing (QGT) [30, 27].

In this paper we study the problem of Group Testing under a more general *capped quantitative result*, where the result (feedback) is the size of the intersection up to some parameter $\alpha \in \{1, \ldots, k\}$, and α for larger intersections. It subsumes and generalizes the two previously described classical result functions: the smallest case of $\alpha = 1$ corresponds to the classical empty/non-empty feedback (beeping), while the case $\alpha = k$ captures the (full) quantitative feedback (classical QGT).

Our focus is on *non-adaptive deterministic* solutions, in which queries are fixed (i.e., the same sequence applies to all hidden sets) and allow to discover (decode) any hidden set based on α-capped results. The primary goal is to minimize the number of queries, called *query complexity*, and the second objective is to compute the queries and decode the hidden set in polynomial time. All existing polynomial construction/decoding algorithms for beeping or quantitative feedbacks use $\Theta(\min\{n, k^2 \log n\})$ queries, c.f., [48]. General constructions developed in this work employ the concept of *dispersion*, c.f., [51], and combine it appropriately with other techniques. When instantiated for a specific feedback parameter $\alpha \in \{1, \ldots, k\}$, (e.g., $\alpha = \sqrt{k}$), they shrink the gap for efficiently constructed query systems nearly exponentially: from linear to polylogarithmic gap. Together with our lower bound, they explain why sometimes a much smaller feedback is sufficient for decoding sets with similar efficiency. We show that parameter α has also different interpretations – it inversely relates to the number of occurrences of an element in the queries and to fault-tolerance.

1.1 Our Results (see also Table 1)

We generalize the classical beeping and QGT models of Group Testing by considering an *α-capped results*, also called an *α-capped feedback* function, for an $\alpha \in \{1, \ldots, k\}$: for a hidden set K and query Q it returns $\mathcal{F}_\alpha(Q \cap K) = \min\{|Q \cap K|, \alpha\}$. We study the query complexity of non-adaptive deterministic QGT in this general model. We focus on scalable (polylogarithmic number of queries per element) and efficient (polynomial-time) constructing/decoding algorithms, and we show that it could be achieved *if and only* if the information cap $\alpha = \widetilde{\Omega}(\sqrt{k})$. In particular, it holds for classical QGT.

Main result – Polynomial-time construction/decoding algorithms using almost optimal number of queries (Algorithms 1 and 2 in Section 4). Here almost-optimality means that the length of the constructed query sequence is only polylogarithmically longer than the shortest possible sequence. The previous best polynomial-time solution used $\Theta(\min\{n, k^2 \log n\})$ queries for all $\alpha \geq 1$ [48], and we shrink it by factor $\Theta(\min\{\alpha^2, k\} \operatorname{polylog}^{-1} n)$. To achieve this goal, we define and build new types of selectors, called (Strong) Selectors under Interference based on the concept of dispersion. We also generalize the concept of Round-Robin query systems, where each query is a singleton, to α-Round-Robin query systems, containing sets of size at most α. Such sequences are shorter than the simple Round-Robin, *i.e.,* have length $O((n/\alpha) \operatorname{polylog} n)$, and, unlike a simple Round-Robin singletons' structure, are challenging to construct in a way to allow correct decoding based on α-capped feedback.

Theorem 1. *There is an explicit polynomial-time algorithm constructing non-adaptive queries $Q_1, \ldots, Q_m$, for $m = O\left(\min\left\{\left(\frac{k}{\alpha}\right)^2, \frac{n}{\alpha}\right\} \operatorname{polylog} n + k \operatorname{polylog} n\right)$, that solve Group Testing un-*

α	Upper bound	First work with asymptotic bound	Difference from this paper
k	$O\left(k\frac{\log n}{\log k}\right)$	[34]	existential (super-exponential construction time)
	$O\left(k\frac{\log n}{\log k}\right)$	[3]	adaptive
	$O\left(k\frac{\log n}{\log k}\right)$	[49]	randomized (only with some probability)

Table 2: Other related work on *adaptive*, *randomized* or *existential* solutions to QGT for $\alpha = k$.

der feedback $\mathcal{F}_\alpha$ with polynomial-time decoding. Moreover, every element occurs in $O(\frac{k}{\alpha}\operatorname{polylog} n)$ queries, and the decoding time is $O(m + \frac{k^2}{\alpha}\operatorname{polylog} n + k\operatorname{polylog} n)$.

One of the consequences is that having the result function capped at $\alpha = \sqrt{k}$, we obtain similar number of queries as with the actual result (i.e., returning the size of the whole intersection, up to k), which together with our next lower bound establishes an interesting inherited property of GT. Section 6 summarizes (and Appendix B describes in detail) several non-straightforward extensions and applications of Theorem 1, such as: fault-tolerance, GT on multisets, online maintenance of dynamically changing graphs/sets, Private Parallel Information Retrieval and coding.

Lower bound (Section 5). The almost-optimality of our algorithms from Theorem 1 is justified by proving an absolute lower bound on the length of sequences allowing to decode a hidden set based on feedback $\mathcal{F}_\alpha$ to the queries. Here by "absolute" we mean that it holds for all query systems that allow for decoding of the hidden sets based on feedback $\mathcal{F}_\alpha$, not restricted to polynomially constructed queries with polynomial decoding algorithm. Even more, some components of the lower bound are general: they hold for *any α-capped* feedback function, which will be formally defined later in Section 3. The lower bound has three components: $(k/\alpha)^2$, n/α, and $k\frac{\log\frac{n}{k}}{\log\alpha}$. For different ranges of k, different components determine the value of the lower bound. Note that these components match the corresponding components in our constructive upper bound (Main result in Theorem 1), up to a polylogarithmic factor.

Theorem 2. *Any non-adaptive algorithm solving Group Testing needs:*

- $\Omega\left(\min\left\{\left(\frac{k}{\alpha}\right)^2, \frac{n}{\alpha}\right\} + k\frac{\log\frac{n}{k}}{\log\alpha}\right)$ *queries under feedback $\mathcal{F}_\alpha$.*
- $\Omega\left(\min\left\{\left(\frac{k}{\alpha}\right)^2, \frac{n}{\alpha}\right\}\right)$ *queries under any feedback capped at α.*

Document structure. We discuss related work on various variants of GT in Section 2. In Section 3 we formally define the QGT problem and the generalized α-capped-results model. In Section 4 we present our polynomial-time query construction and decoding algorithms. Section 5 proves the lower bound. Discussion of extensions, applications and future directions is given in Section 6. Additional Appendix contains more details: Appendix A discusses other related work; Appendix B shows details of extensions and applications; Our new selector tools are constructed and analyzed in Appendix C.

2 Previous and Related Work (see also Tables 1 and 2)

In the standard feedback model, considered in most of the Group Testing literature [25], the feedback tells whether the intersection between query Q and set K is empty or not (sometimes it is also called a *beeping model*). It it is a special case $\mathcal{F}_1$ of our feedback function. In this feedback model, Group Testing is known to be solvable using $O(k^2\log(n/k))$ queries [20] and an explicit polynomial-time construction of length $O(k^2\log n)$ exists [48]. Best known lower bound (for $k < \sqrt{n}$) is $\Omega(k^2\log n/\log k)$ [11].

The setting considered in this paper is also a generalization of an existing problem of coin weighting. In the coin weighting problem, we have a set of n coins of two distinct weights w_0 (true coin) and w_1 (counterfeit coin), out of which up to k are counterfeit ones. We are allowed to weigh any subset of coins on a spring scale, hence we can deduce the number of counterfeit coins in each weighting. The task is to identify all the counterfeit coins. Such a feedback is a special case $\mathcal{F}_k$ of our feedback function. The problem is solvable with $O(k\log(n/k)/\log k)$ [34] non-adaptive (i.e., fixed in advance) queries and matching a standard information-theoretic lower bound of $\Omega(k\log(n/k)/\log k)$, as

well as its stronger version proved for randomized strategies [22]. [3] considers the problem of explicit polynomial-time construction of $O(k\log(n/k)/\log k)$ queries that allows for polynomial time identification of the counterfeit coins. However, the algorithm presented in [3] is adaptive, which means that the subsequent queries can depend on the feedback from the previous ones. The only existing, constructive, non-adaptive solution would be using the explicit construction of the superimposed codes [42] but the resulting query complexity would be $O(k^2 \operatorname{polylog} n)$. Thus, the solution in our paper is the first explicit polynomial time algorithm constructing non-adaptive queries allowing for fast decoding of set K, with $O(k \operatorname{polylog} n)$ fixed queries.

An important line of work on non-adaptive *randomized* solutions to Quantitative Group Testing, started by [49] and being continued until recently [30, 13, 27, 2], resulted in a number of algorithms nearing the lower bound of $2k\frac{\ln(n/k)}{\ln k}$ from [23]. However, these results always assume some restriction on k (typically $k \sim n^\theta$ for some $0 < \theta < 1$), and similarly as above, they may result in significant deviation from the actual set if n is large.

The bounds obtained in this paper match (up to polylogarithmic factors) the best existing results for the extreme cases of $\alpha = 1$ and $\alpha = k$. The paper also bounds how the query complexity depends on the value of α between these extremes. Interestingly we show, that the shortest-possible query complexity of $k \operatorname{polylog} n$ is already possible for $\alpha = \sqrt{k}$ and increasing α from $\sqrt{k}$ to k does not result in further decrease of the query complexity.

Other interesting feedback function is a *Threshold Group Testing* [19], where the feedback model includes a set of thresholds and the feedback function returns whether or not the size of the intersection is larger or smaller than each threshold. In [21] the authors show that it is possible to define an interval of $\sqrt{k \log k}$ thresholds resulting in an algorithm with $O(k\log(n/k)/\log k)$ queries. Note that both those feedbacks are "inefficient" in view of our setting of α-capped feedbacks, because their feedback functions are not capped at any $\alpha < k$, but they achieve similar query complexity as our capped $\mathcal{F}_{\sqrt{k}}$ feedback.

Superimposed codes and dispersers were already used in Group Testing. However, they either led to a super-quadratic (in k) number of queries, c.f., [42, 8], or decoded only a fraction of elements of the hidden set, see [40]. Recall that in solutions where query complexity depends on the number of identified elements, decoding of all the elements requires over k^2 queries, as proved in [20, 9]. [41] present the first Group Testing solution with $poly(k, \log n)$ decoding time, but super-quadratic query complexity. It is worth noting that our generalized solution achieves almost-linear number of queries (for certain values of α) and our decoding algorithm identifies all the elements in time polynomial in k and logarithmic in n. Our use of the known tools is different then in previous approaches: we define new properties (SuI, SSuI), which we prove to be satisfied by some combinations of those tools, and lead to efficient solutions in both query complexity and construction/decoding time.

3 The Model and the Problem

We assume that the universe of all elements N, with $|N| = n$, is enumerated with integers $1, 2, \ldots, n$. Throughout the paper, we will associate an element with its identifier. Let K, with $|K| \leq k$, denote a hidden subset of N chosen arbitrarily by an adversary. Let $\mathcal{Q} = \langle Q_1, \ldots, Q_m \rangle$ be a non-adaptive algorithm, represented by a sequence of m queries fixed prior to an execution.

Consider a feedback function $\mathcal{F}_\alpha$, also called a result function, that returns the size of an intersection if it is at most α and α for larger intersections, *i.e.,* $\mathcal{F}_\alpha(Q \cap K) = \min\{|Q \cap K|, \alpha\}$. Parameter α in feedback $\mathcal{F}_\alpha$ is called a *feedback cap*. A general class of feedback functions (used in our lower bound) with feedback cap α includes all deterministic functions that take subsets of N as input and for sets with more than α elements output some arbitrary fixed value.

We say that $\mathcal{Q}$ solves Quantitative Group Testing (QGT), if the feedback vector allows for unique identification of set K. The feedback vector is defined as $\langle \mathcal{F}_\alpha(Q_1 \cap K), \mathcal{F}_\alpha(Q_2 \cap K), \ldots, \mathcal{F}_\alpha(Q_t \cap K) \rangle$. Thus, in order to solve QGT, the feedback vectors for any two sets K_1 and K_2 have to be different. We say that $\mathcal{Q}$ solves QGT with polynomial-time reconstruction if there exists a polynomial-time algorithm that, given the feedback vector outputs all the identifiers of the elements from K. Finally we say that $\mathcal{Q}$ is constructible in polynomial time if there exists a polynomial-time algorithm, that given parameters n, k, α outputs an appropriate sequence of queries.

We assume that both coupled algorithms, construction and decoding, know $n, k, \mathcal{F}_\alpha$. W.l.o.g., in order to avoid rounding in the presentation, we assume that n and other crucial parameters are powers of 2.

4 Polynomial-time Constructions and Decoding

4.1 Combinatorial Tools

In this section we introduce two new combinatorial tools (Selectors-under-Interference and Strong-Selectors-under-Interference) for our QGT solutions, and recall Dispersers and Balanced IDs.

For given sets $K_1, K_2 \subseteq N$ and an element $v \in K_1$, we say that $S \subseteq N$ *selects* v *from* K_1 *under* α*-interference from* K_2 if $S \cap K_1 = \{v\}$ and $|S \cap K_2| < \alpha$. Intuitively, v is a unique representative of K_1 in S and the number of representatives of K_2 in S is smaller than α.

Definition 1 (Selector under Interference (SuI))**.** *An $(n, \ell, \epsilon, \kappa, \alpha)$-Selector-under-Interference, $(n, \ell, \epsilon, \kappa, \alpha)$-SuI for short, is a sequence of queries $\mathcal{S} = (S_1, \ldots, S_x)$ satisfying: for every set $K_1 \subseteq N$ of at most ℓ elements and set $K_2 \subseteq N$ of at most κ elements, there are at most $\epsilon\ell$ elements $v \in K_1$ that* are not *selected from K_1 under α-interference from K_2 by any query $S_i \in \mathcal{S}$, i.e., set $\{v \in K_1 \ : \ \forall_{i \le x}\ S_i \cap K_1 \neq \{v\}$ or $|S_i \cap K_2| \ge \alpha\}$ has less than $\epsilon\ell$ elements.*

Disperser. Consider a bipartite graph $G = (V, W, E)$, where $|V| = n$, which is an $(\ell^* = \epsilon\ell, d, \epsilon)$-*disperser with entropy loss* δ, i.e., it has left-degree d, $|W| = \ell d/\delta$, and satisfies the following dispersion condition: for each $L \subseteq V$ such that $|L| \ge \ell^*$, the set $N_G(L)$ of neighbors of L in graph G is of size at least $(1 - \epsilon)|W|$. Note that it is enough for us to take as ϵ in the dispersion property the same value as in the constructed $(n, \ell, \epsilon, \kappa, \alpha)$-SuI $\mathcal{S}$.[1] An explicit construction (i.e., in time polynomial in n) of dispersers was given by [51], for any $n \ge \ell$, and some $\delta = O(\log^3 n)$, where $d = O(\text{polylog } n)$.[2] In the following the notation d and δ will always denote the parameters of the disperser. Note that any improved construction of dispersers will immediately reduce the complexities of our algorithms. In Appendix C.1 we will describe two disperser-based polynomial-time constructions of SuI and prove:

Theorem 3. *There is an explicit polynomial-time construction of an $(n, \ell, \epsilon, \kappa, \alpha)$-SuI, for any ℓ, any $\alpha \le k$ such that $\ell\alpha/(3\delta) > \kappa$, for any constant $\epsilon \in (0, 1/3)$, of size $O(\min\{n, \ell d\delta \log^2 n\})$. Moreover, every element occurs in $O(d\delta \log n)$ queries.*

Theorem 4. *There is an explicit polynomial-time construction of an $(n, \ell, \epsilon, \kappa, \alpha)$-SuI, for any ℓ, any $\alpha \le k$ such that $\ell \le 3\delta\kappa/\alpha$ for any constant $\epsilon \in (0, 1/3)$, of size $O\left(\min\left\{n, \frac{\kappa d\delta}{\alpha} \log^2 n\right\} + \frac{nd\delta}{\alpha} \log n\right)$. Moreover, every element occurs in $O(d\delta \log n)$ queries.*

Definition 2 (Strong Selector under Interference (SSuI))**.** *An $(n, \ell, \kappa, \alpha)$-Strong-Selector-under-Interference, $(n, \ell, \kappa, \alpha)$-SSuI for short, is a sequence of queries $\mathcal{T} = (T_1, \ldots, T_x)$ satisfying: for every set $K_1 \subseteq N$ of at most ℓ elements and set $K_2 \subseteq N$ of at most κ elements,* every *element $v \in K_1$ is selected from K_1 under α-interference from K_2 by some query $T_i \in \mathcal{T}$, i.e., set $\{v \in K_1 \ : \ \forall_{i \le x}\ T_i \cap K_1 \neq \{v\}$ or $|T_i \cap K_2| \ge \alpha\}$ is empty.*

An $(n, \ell, \kappa, \alpha)$-Strong-Selector-under-Interference could be also viewed as $(n, \ell, 0, \kappa, \alpha)$-Selector-under-Interference. In Appendix C.2 we describe a polynomial-time construction of SSuI, which essentially is a [42] construction for adjusted parameters, and prove that it satisfies the SSuI property.

Theorem 5. *There is an explicit polynomial-time construction of an $(n, \ell, \kappa, \alpha)$-SSuI of length $O(\ell^2 \log_\ell^2 n)$, provided $\ell \ge 3\delta\kappa/\alpha$. Moreover, every element occurs in $O(\ell \log_\ell n)$ queries.*

Balanced IDs. Each element i in N has a unique ID represented by $2\log_2 n$ bits, in which the number of 1's is the same as the number of 0's; e.g., take a binary representations of elements i and $n - i$, each in $\log_2 n$ bits, and concatenate them (recall $n = |N|$). Balanced IDs have previously been used in algorithms for decoding elements in Group Testing (see e.g., [44]).

4.2 Construction of Queries, Decoding and Analysis (Proof of Theorem 1)

Algorithm constructing queries. Assume $\alpha \ge 2$ and $k > 3\delta k/(\alpha - 1)$ (the opposite case will be considered later) and let us take $(n, \ell, 1/2, k, \alpha - 1)$-SuI $\mathcal{S}^{(\ell)}$, for ℓ being a power of 2 ranging down from k to $3\delta k/(\alpha - 1)$ (w.l.o.g. we could also assume that $3\delta k/(\alpha - 1)$ is a power of 2). Next, for each set S in these selectors we add the following family $\mathcal{R}(S) = \{R_i(S)\}_{i=1}^{2\log_2 n}$ of sets

[1] If someone considers $\epsilon \ge 1/3$, we could use construction for $\epsilon = 1/3$ to get solution with better guarantees.

[2] Optimal dispersers have $\delta = O(1)$ and $d = O(\log n)$, but their polynomial-time construction is an open problem.

$R_i(S) = \{v \in S \ : \ \lfloor v/2^{i-1} \rfloor = 1 \mod 2\}$. Intuitively $R_i(S)$ is the set of elements from S that have 1 on i-th least significant bit of Balanced ID. Let us call the obtained enhanced selectors (i.e., with additional families $\mathcal{R}(S)$, for every set S in the original selector) $\bar{\mathcal{S}}^{(\ell)}$. Then we concatenate selectors $\mathcal{R}(\mathcal{S}^{(\ell)})$, starting from the largest $\ell = k$, to the smallest value $\ell = 3\delta k/(\alpha - 1)$. An $(n, 3\delta k/(\alpha-1), k, \alpha-1)$-SSuI $\mathcal{T}$ is concatenated at the end, with the same replacement of bits 1 and 0 that gives $\mathcal{R}(\mathcal{T})$ as in the above $(n, \ell, 1/2, k, \alpha-1)$-SuI's. In the case, where $\alpha = 1$ or $k \leq 3\delta k/(\alpha-1)$ then the sequence equals to (n, k, k, α)-SSuI $\mathcal{T}$ concatenated as in the above with $\mathcal{R}(\mathcal{T})$. Algorithm 1 presents a pseudocode of the construction algorithm for $\alpha > 1$.

Algorithm 1: Construction of a sequence of queries solving QGT with α-capped feedback.

Input: Disperser G of degree d and entropy loss δ

$\ell \leftarrow k, \mathcal{Q} \leftarrow \langle\rangle$;
while $\ell > \frac{3\delta l}{\alpha-1}$ **do**
  $\mathcal{S} \leftarrow (n, \ell, 1/2, k, \alpha-1)$-SuI;
  foreach $S \in \mathcal{S}$ **do**
    $\mathcal{Q}$.**append**(S);
    for $i \leftarrow 1$ **to** $2\log_2 n$ **do**
      /* Add a set of elements from S that have 1 on i-th least significant bit of Balanced ID. */
      $R_i(S) \leftarrow \{v \in S \ : \ \lfloor v/2^{i-1} \rfloor = 1 \mod 2\}$;
      $\mathcal{Q}$.**append**$(R_i(S))$;
  $\ell \leftarrow \ell/2$;
$\mathcal{T} \leftarrow (n, 3\delta k/(\alpha-1), k, \alpha-1)$-SSuI;
foreach $T \in \mathcal{T}$ **do**
  $\mathcal{Q}$.**append**(T);
  for $i \leftarrow 1$ **to** $2\log_2 n$ **do**
    $R_i(T) \leftarrow \{v \in T \ : \ \lfloor v/2^{i-1} \rfloor = 1 \mod 2\}$;
    $\mathcal{Q}$.**append**$(R_i(T))$;
return $\mathcal{Q}$

Decoding algorithm. During the decoding algorithm we process, in subsequent iterations, the feedbacks from enhanced selectors $\bar{\mathcal{S}}^{(\ell)}$ for $\ell = k, k/2, k/4, \ldots, 3\delta k/(\alpha-1)$. We will later prove, by induction, that during processing $\bar{\mathcal{S}}^{(\ell)}$, $\ell/2$ new elements from K are decoded. To show this, we consider any iteration and let set K_1 be the set of the elements that have been decoded in previous iterations while set $K_2 = K \setminus K_1$ be the set of unknown elements. We treat K_1 as the interfering set and, by the properties of $\mathcal{S}^{(\ell)}$, we know that for at least $\ell/2$ elements v, there exists a query $S \in \mathcal{S}^{(\ell)}$, such that $v \in S$, $|K_1 \cap S| < \alpha - 1$, $|K_2 \cap S| = 1$. We observe that since we already know the identifiers of all the elements from the interfering set K_1, then using feedbacks from the additional queries $\mathcal{R}(S)$ (corresponding to balanced IDs) we can exactly decode the identifier of v. We do this for all $\ell/2$ elements that are possible to decode in this iteration and we proceed to the next iteration. After considering all Selectors-under-Interference, we have only at most $3\delta k/(\alpha-1)$ unknown elements. To complete the decoding we use a Strong-Selector-under-Interference, where the decoding procedure is exactly the same as in the case of SuI (the interfering set is also the set of already decoded elements). The properties of SSuI guarantee that we decode the identifiers of all the remaining elements from K. See the pseudocode of decoding Algorithm 2 for details of deocding for $\alpha > 1$. In the case where $\alpha = 1$ the decoding is straightforward, as the SSuI guarantees in this case no interference, hence the results (due to our enhancement $\bar{\mathcal{T}}$) contain exactly the identifiers of the elements from set K.

Lemma 1. *There is an explicit polynomial-time algorithm constructing non-adaptive queries $Q_1, \ldots, Q_m$ and decoding any hidden set K of size at most $k \leq n$, from the feedback vector in polynomial time, under feedback $\mathcal{F}_\alpha$ and for $m = O((\frac{k}{\alpha})^2 \delta^2 \log^3 n + kd\delta \log^3 n)$ queries. Moreover, every element occurs in $O(\frac{k}{\alpha}\delta \log^2 n + d\delta \log^3 n)$ queries.*

Proof. We start from describing a procedure of revealing elements in any given set K of at most k elements, together with a formal (inductive) argument of its correctness. Our first goal is to show that by the beginning of $\bar{\mathcal{S}}^{(\ell)}$, for ℓ stepping down from k to $3\delta k/(\alpha-1)$, we have not learned about the identity of at most ℓ elements from the hidden set K.

The proof is by induction – it clearly holds in the beginning of the computation, as the set K has at most $\ell = k$ elements. We prove the inductive step: by the end of $\bar{\mathcal{S}}^{(\ell)}$, at most $\ell/2$ elements are

Algorithm 2: Decoding of the elements for QGT with α-capped feedback.

Input: $\mathsf{Result}(Q)$ for queries $Q \in \mathcal{Q}$ constructed by Algorithm 1, based on SuI $\mathcal{S}^{(\ell)}$ for ℓ being powers of 2 stepping down from k to $\frac{3\delta l}{\alpha-1}$ and SSuI $\mathcal{T}$.

$\ell \leftarrow k, K_{acc} \leftarrow \emptyset$; /* Set K_{acc} accumulates the decoded elements. */
while $\ell > \frac{3\delta l}{\alpha-1}$ **do** /* We decode $l/2$ elements in each iteration.*/
  for $i \leftarrow 1$ **to** $l/2$ **do**
    /* Look for query for which we can decode a new element. */
    foreach $S \in \mathcal{S}^{(\ell)}$ **do**
      if $\mathsf{Result}(S) < \alpha - 1$ **and** $|S \cap K_{acc}| = \mathsf{Result}(S) - 1$ **then**
        $K_{acc} \leftarrow K_{acc} \cup \{DecodeElement(S, K_{acc})\}$;
  $\ell \leftarrow \ell/2$;
while $\ell > 0$ **do** /* Iteratively decode all the remaining elements.*/
  foreach $T \in \mathcal{T}$ **do**
    if $\mathsf{Result}(T) < \alpha - 1$ **and** $|T \cap K_{acc}| = \mathsf{Result}(T) - 1$ **then**
      $K_{acc} \leftarrow K_{acc} \cup \{DecodeElement(T, K_{acc})\}$;
      $\ell \leftarrow \ell - 1$;
return K_{acc}

Procedure *DecodeElement(Q,K_{acc})*
  $v \leftarrow 0$;
  for $j \leftarrow 2\log_2 n$ **downto** 0 **do**
    /* Take the feedback from set $R_j(Q)$. Calculate the feedback from set $R_j(Q)$, if hidden set was exactly K_{acc}. The difference is j-th least significant bit of Balanced ID of new element v. */
    $v \leftarrow 2 \cdot v + \mathsf{Result}(R_j(Q)) - |R_j(Q) \cap K_{acc}|$
  return v

not learned. We set K_2 to be the set of learned elements and $K_1 = K \setminus K_2$. Clearly, $|K_2| \leq k$, and by the inductive assumption $|K_1| \leq \ell \leq k$. For such K_1 and K_2, by the definition of SuI, there are at most $\ell/2$ elements from K_1 that are not occurring in some round without other such elements or with at least $\alpha - 1$ of already learned elements from K_2. Consider a previously not learned element $v \in K_1$, for which there exists a *good query* in $\bar{\mathcal{S}}^{(\ell)}$, *i.e.,* a query $S \in \bar{\mathcal{S}}^{(\ell)}$ such that $S \cap K_1 = \{v\}$ and $|S \cap K_2| < \alpha - 1$. At this point of decoding of set K we know the Balanced IDs of all elements from set K_2. Hence we can calculate the $2\log_2 n$-bit feedback vector from sets $K_2 \cap R_1(S), K_2 \cap R_2(S), \ldots, K_2 \cap R_{2\log_2 n}(S)$. We compare this feedback vector with the output of the enhanced selector, which is the feedback vector for sets $K \cap R_1(S), K \cap R_2(S), \ldots, K \cap R_{2\log_2 n}(S)$. The difference between the latter and the former is exactly the Balanced ID of v. In case this difference does not form a Balanced ID of any element, *i.e.,* it has some value bigger than 1 or otherwise the number of 1's is different from $\log_2 n$, or in case $\mathcal{F}_\alpha(K \cap S) = \alpha$ (recall that S is also in the constructed selector) the feedback from this $\mathcal{R}(S)$ is ignored. This is done to avoid misinterpreting the feedback and false discovery of an element which is not in K. Indeed, first note that the fact $|K \cap S| \geq \alpha$ will automatically discard the part of the feedback from $K \cap R_1(S), K \cap R_2(S), \ldots, K \cap R_{2\log_2 n}(S)$, as it indicates that the intersection is too large to provide correct decoding of an element. Second, assuming $|K \cap S| < \alpha$, if there are no elements in $K_1 \cap S$ then the difference between feedbacks gives vector of zeros, and if there will be at least two elements in $K_1 \cap S$, the difference between feedbacks will contain a value of at least 2 or all 1's, as it will be a bitwise sum of at least two Balanced IDs of $\log_2 n$ ones each. By the definition of SuI we can find $l/2$ such elements v. This shows that during decoding of enhanced $\bar{\mathcal{S}}^{(\ell)}$ we learn the identities of $\ell/2$ new elements. This completes the inductive proof. Note here that the inductive step, being one of $O(\log n)$ steps, defines a polynomial time algorithm decoding some elements one-by-one – indeed, it computes two feedbacks of polynomial number of queries, computes the difference and deducts based on the structure of subsequent blocks of $O(\log n)$ size.

The above analysis implies, that before applying $(n, 3\delta k/(\alpha-1), k, \alpha-1)$-SSuI we have not discovered at most $3\delta k/(\alpha-1)$ elements. Thus, by definition, $(n, 3\delta k/(\alpha-1), k, \alpha-1)$-SSuI combined with Balanced IDs reveals all the remaining elements in the same way as the SuI's

above – the only difference in the argument is that instead of leaving at most $\ell/2$ undiscovered elements in the ℓ-th inductive step, due to the nature of SuI's, the SSuI guarantees that *every* undiscovered element will occur in a good query. The same argument as for SuI's proves that the decoding algorithm defined this way works in polynomial time. By Theorem 3, the length of $(n, \ell, 1/2, k, \alpha)$-SuI is $O(\min\{n, \ell d\delta \log^2 n\})$, which sums up to $O(kd\delta \log^2 n)$, and is multiplied by $\Theta(\log n)$ due to amplification by Balanced IDs. By Theorem 5, the length of $(n, 3\delta k/\alpha, k, \alpha)$-SSuI is $O((\frac{k}{\alpha})^2\delta^2 \log^2 n)$, and it is also increased by factor $\Theta(\log n)$ due to Balanced IDs. If we apply the above reasoning with respect to the number of queries containing an element, we get that every element occurs in $O(\frac{k}{\alpha}\delta \log^2 n + d\delta \log^3 n)$ queries. □

Implementing α-Round-Robin for large values of k/α. The question arises from the previous result if one could efficiently construct a shorter sequence of queries if $(k/\alpha)^2 > n/\alpha$? In the case of full feedback (*i.e.,* $\alpha = k$) the common way to deal with large values of k is via Round-Robin, which means that queries are singletons and consequently, the length of such query sequence is n. This also works for an arbitrary value of $\alpha \leq k$, however the lower bound in Theorem 2 suggest that in such case there could exist a shorter query system of length $O((k + (n/\alpha)) \operatorname{polylog} n)$. Indeed, if we modify our construction in such case, we could obtain such a goal. Namely, we concatenate:

- selectors $\bar{\mathcal{S}}^{(\ell)}$, for ℓ being decreasing powers of 2 from $\ell = k$ to $\ell = 6\delta k/(\alpha - 1)$;
- selectors $\bar{\mathcal{S}}|_{\alpha}^{(\ell)}$, for ℓ being decreasing powers of 2 from $\ell = 3\delta k/(\alpha - 1)$ to $\ell = 1$.

In the case, where $k > 6\delta k/(\alpha - 1)$, then the construction contains only selectors $\bar{\mathcal{S}}|_{\alpha}^{(\ell)}$. Then we enhance them based on Balanced IDs, as in the previous construction. Then, applying Theorem 3 for concatenated $\bar{\mathcal{S}}^{(\ell)}$ and Theorem 4 for concatenated $\bar{\mathcal{S}}|_{\alpha}^{(\ell)}$, instead of combination of Theorems 3 and 5 as it was in the proof of Lemma 1 with respect to $\bar{\mathcal{S}}^{(\ell)}$, we get the following result.

Lemma 2. *There is an explicit polynomial-time algorithm constructing non-adaptive queries $Q_1, \ldots, Q_m$ and decoding any hidden set K of size at most $k \leq n$, from the feedback vector in polynomial time, under feedback $\mathcal{F}_\alpha$ and for $m = O((k + n/\alpha)d\delta \log^3 n)$ queries. Moreover, each element occurs in $O(d\delta \log^3 n)$ queries.*

Proof. The proof is analogous to the proof of Lemma 1, except that we continue proving the invariant until $\ell = 6\delta\kappa/(\alpha - 1)$, using same properties guaranteed by Theorems 3, and continue the invariant until $\ell = 1$, using SuI's of slightly different length formula from Theorem 4. The correctness argument, as well as polynomial-time query construction and decoding of the elements, are the same as in the invariant proof in Lemma 1. Then, by Theorem 3 for concatenated $\bar{\mathcal{S}}^{(\ell)}$ and by Theorem 4 for concatenated $\bar{\mathcal{S}}|_{\alpha}^{(\ell)}$, we argue that the total length of the obtained sequence is $m = O((k + n/\alpha)d\delta \log^3 n)$. Indeed, the first part results from the telescoping sum for different ℓ and the second component is a logarithmic amplification of the original $O((nd\delta/\alpha) \log n)$ length of SuI's; all is amplified by $O(\log n)$ due to enhancement of the used SuI's by Balanced IDs. In all the components, every element belongs to $O(d\delta \log n)$ queries, by Theorems 3 and 4, in the final sequence it occurs in $O(d\delta \log^3 n)$ queries. □

Combining Lemma 1 with Lemma 2 gives Theorem 1. Note that in both lemmas the decoding algorithm proceeds query-by-query, each time spending polylogarithmic time on each of them; additionally, for each decoded element, an update of the feedback of next queries needs to be done, which takes time proportional to the number of occurrences of the discovered element in the queries. Thus, it is asymptotically upper bounded by the length of the sequence plus k times the upper bound on the number of occurrences of an element in the queries (polylogarithmic).

5 Lower Bound

Proof of Theorem 2. We will first show the $\min\{n/\alpha, k^2/\alpha^2\}$ component. Assume that a sequence of queries $Q_1, Q_2, \ldots, Q_t$ of length t solves Group Testing. We want to show the lower bound that holds for any feedback function capped at α hence we assume that the feedback function $\mathcal{F}$ returns the whole set (i.e., the identifiers of all the elements). Recall that $\mathcal{F}$ works only for sets with at most α elements. We begin by proving the following:

Claim A: For any set K, with $|K| \leq k$ and any $x \in K$, there must exist $\tau \in \{1, 2, \ldots, t\}$, such that $x \in Q_\tau$ and $|K \cap Q_\tau| \leq \alpha + 1$.

The proof is by contradiction. Assume that such a set K^* and element x^* exist for which there is no such query. Consider feedback vectors for sets K^* and $K^* \setminus \{x^*\}$. For any query that does

not contain x^*, the feedback is clearly identical. For any query Q_τ, such that $x^* \in Q_\tau$, we have $|Q_\tau \cap K^*| \geq \alpha+2$ and $|Q_\tau \cap (K^* \setminus \{x^*\})| \geq \alpha+1$ and sets K^* and $K^* \setminus \{x^*\}$ are indistinguishable under any feedback capped at α hence the sequence of queries does not solve the problem. This completes the proof of Claim A. ■

Take all queries that have at most $\alpha + 1$ elements and all elements that belong to such queries. We have: $N_s = \bigcup_{\tau \in \{1,2,\ldots,t\} | Q_\tau| \leq \alpha+1} Q_\tau$ and remaining elements $N_l = N \setminus N_s$. Consider two cases:
Case 1: $|N_s| \geq n/2$.

Observe that: $t \geq |\{\tau \in \{1, 2, \ldots, t\} : |Q_\tau| \leq \alpha + 1\}| \geq \frac{N_s}{\alpha+1} \geq \frac{n}{2(\alpha+1)}$.

Case 2: $|N_l| \geq n/2$.

In this case, we take an arbitrary subset K_1 of $k/2$ elements from N_l. For every element $x \in K_1$, we consider a set of queries $Q(x) = \{Q_\tau \in \{Q_1, Q_2, \ldots, Q_t\} : x \in Q_\tau, |Q_\tau \cap K_1| \leq \alpha + 1\}$. We first show the following:

Claim B: For every $x \in K_1$, we have $|Q(x)| \geq \frac{k}{2(\alpha+2)}$. The proof is by contradiction. Assume that for some $x^* \in K_1$ we have $|Q(x^*)| < \frac{k}{2(\alpha+2)}$. Then, for every query $Q \in Q(x^*)$, we take $\alpha + 2 - |Q \cap K_1|$ elements from $Q \setminus K_1$. Such elements exist since $|Q| \geq \alpha + 2$. Choose such elements for each query in $Q(x^*)$ and gather them in set K_2. Note that since $|Q(x^*)| < \frac{k}{2(\alpha+2)}$, then $|K_2| \leq k/2$. Now observe that set $K_1 \cup K_2$ and element x^* violate Claim A. The obtained contradiction completes the proof of Claim B. ■

Now observe that each query belongs to at most $\alpha + 1$ sets $Q(x)$ for different values of $x \in K_1$. Thus: $t \geq \frac{\sum_{x \in K_1} |Q(x)|}{\alpha+1} \geq \frac{k^2}{4(\alpha+1)(\alpha+2)}$. To complete the proof observe that any algorithm must fall either into Case 1 or Case 2, hence any algorithm needs to use $\Omega\left(\min\{n/\alpha, k^2/\alpha^2\}\right)$ queries.

To see that any algorithm in $\mathcal{F}_\alpha$ feedback model at least $k\frac{\log \frac{n}{k}}{\log \alpha}$ queries, observe that the feedback vector must be unique for each set K with at most k elements. Hence we need at least $\binom{n}{k}$ different feedback vectors for different sets. Feedback has at most α values hence we get $\alpha^t \geq \binom{n}{k}$ and $t \in \Omega(k\frac{\log \frac{n}{k}}{\log \alpha})$. This completes the proof of Theorem 2. ■

6 Extensions, Applications and Open Directions

Fault-tolerance. Algorithms 1 and 2 are fault-tolerant with respect to $f \leq c \cdot \frac{k}{\alpha}$ *adversarially* jammed results, for some sufficiently small constant $c > 0$ – here by a jammed result we understand receiving a null as a result of a query. It also tolerates *stochastic* failures, where a result is jammed with probability $p \leq c$. Indeed, in adversarial failures we observe that, by dispersion, $c \cdot \frac{k}{\alpha}$ could be made negligible (by right selection of constant c) with respect to the number of selected elements in SuI, and it is also smaller than the number of single occurrences (i.e., without interference from other $k - 1$ elements) of an element in SSuI. Thus the analysis of the algorithms hold (the original analysis have margins to accommodate those additional negligible jamming, as they were done with only asymptotic upper bounds). The tolerance of stochastic failures follows from the fact that a random p-fraction of results of queries coming from SuI's and SSuI are again negligible from perspective of the fraction of correctly encoded elements and can be accommodated by our asymptotic analysis.

Theorem 6. *Algorithms 1 and 2 tolerate $f \leq c \cdot \frac{k}{\alpha}$ adversarial jammings and stochastic jammings with probability $p \leq c$, for some sufficiently small constant $c > 0$.*

Other applications. Applications to efficient GT and dynamic maintenance of multi-sets, graphs, as well as to Private Parallel Information Retrieval and coding, are detailed in Appendix B.

Open directions. Considering only polynomially-constructible query systems leaves some interesting open directions. One such open direction is whether optimal-length query sequence can be constructed in polynomial time or perhaps it is possible to show some reduction that constructing a nearly-shortest query sequence is computationally hard (even if we know that it exists). Shrinking polylogarithmic gaps between lower and (existential) upper bounds is another challenging direction, improving constants, as well as considering other interesting classes of GT models with an α-capped results, e.g., parity. We also believe that with some adjustment, GT codes could be applied to efficiently solve many open problems in online streaming, communication and graph learning fields.

References

[1] N. Augenblick, J. T. Kolstad, Z. Obermeyer, and A. Wang. Group testing in a pandemic: The role of frequent testing, correlated risk, and machine learning. Technical report, National Bureau of Economic Research, 2020.

[2] W. H. Bay, E. Price, and J. Scarlett. Optimal non-adaptive probabilistic group testing in general sparsity regimes. *arXiv preprint arXiv:2006.01325*, 2020.

[3] N. H. Bshouty. Optimal algorithms for the coin weighing problem with a spring scale. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009.

[4] C.-C. Cao, C. Li, and X. Sun. Quantitative group testing-based overlapping pool sequencing to identify rare variant carriers. *BMC bioinformatics*, 15(1):1–14, 2014.

[5] J. Capetanakis. Generalized tdma: The multi-accessing tree protocol. *IEEE Transactions on Communications*, 27(10):1476–1484, 1979.

[6] J. Capetanakis. Tree algorithms for packet broadcast channels. *IEEE transactions on information theory*, 25(5):505–515, 1979.

[7] K. Censor-Hillel, B. Haeupler, N. A. Lynch, and M. Médard. Bounded-contention coding for the additive network model. *Distributed Computing*, 28(5):297–308, 2015.

[8] M. Cheraghchi and J. Ribeiro. Simple codes and sparse recovery with fast decoding. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 156–160. IEEE, 2019.

[9] B. S. Chlebus and D. R. Kowalski. Almost optimal explicit selectors. In M. Liskiewicz and R. Reischuk, editors, *Fundamentals of Computation Theory, 15th International Symposium, FCT 2005, Lübeck, Germany, August 17-20, 2005, Proceedings*, volume 3623 of *Lecture Notes in Computer Science*, pages 270–280. Springer, 2005.

[10] S.-S. Choi and J. H. Kim. Optimal query complexity bounds for finding graphs. *Artificial Intelligence*, 174(9-10):551–569, 2010.

[11] A. E. F. Clementi, A. Monti, and R. Silvestri. Selective families, superimposed codes, and broadcasting on unknown radio networks. In *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms, January 7-9, 2001, Washington, DC, USA*, pages 709–718. ACM/SIAM, 2001.

[12] R. Clifford, K. Efremenko, E. Porat, and A. Rothschild. Pattern matching with don't cares and few errors. *Journal of Computer and System Sciences*, 76(2):115–124, 2010.

[13] A. Coja-Oghlan, O. Gebhard, M. Hahn-Klimroth, and P. Loick. Optimal group testing. In J. D. Abernethy and S. Agarwal, editors, *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pages 1374–1388. PMLR, 2020.

[14] G. Cormode, M. Garofalakis, S. Muthukrishnan, and R. Rastogi. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 25–36, 2005.

[15] G. Cormode and M. Hadjieleftheriou. Finding frequent items in data streams. *Proceedings of the VLDB Endowment*, 1(2):1530–1541, 2008.

[16] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava. Finding hierarchical heavy hitters in data streams. In *Proceedings 2003 VLDB Conference*, pages 464–475. Elsevier, 2003.

[17] G. Cormode and S. Muthukrishnan. What's hot and what's not: tracking most frequent items dynamically. *ACM Transactions on Database Systems (TODS)*, 30(1):249–278, 2005.

[18] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. In *International colloquium on structural information and communication complexity*, pages 280–294. Springer, 2006.

[19] P. Damaschke. Threshold group testing. *Electronic Notes in Discrete Mathematics*, 21:265 – 271, 2005. General Theory of Information Transfer and Combinatorics.

[20] A. De Bonis, L. Gasieniec, and U. Vaccaro. Generalized framework for selectors with applications in optimal group testing. In *Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 - July 4, 2003. Proceedings*, volume 2719 of *Lecture Notes in Computer Science*, pages 81–96. Springer, 2003.

[21] G. De Marco, T. Jurdzinski, and D. R. Kowalski. Optimal channel utilization with limited feedback. In L. A. Gasieniec, J. Jansson, and C. Levcopoulos, editors, *Fundamentals of Computation Theory - 22nd International Symposium, FCT 2019, Copenhagen, Denmark, August 12-14, 2019, Proceedings*, volume 11651 of *Lecture Notes in Computer Science*, pages 140–152. Springer, 2019.

[22] G. De Marco and D. R. Kowalski. Searching for a subset of counterfeit coins: Randomization vs determinism and adaptiveness vs non-adaptiveness. *Random Structures & Algorithms*, 42(1):97–109, 2013.

[23] A. Djackov. On a search model of false coins. In *Topics in Information Theory (Colloquia Mathematica Societatis Janos Bolyai 16). Budapest, Hungary: Hungarian Acad. Sci*, pages 163–170, 1975.

[24] R. Dorfman. The detection of defective members of large populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943.

[25] D. Du, F. K. Hwang, and F. Hwang. *Combinatorial group testing and its applications*, volume 12. World Scientific, 2000.

[26] J. Engels, B. Coleman, and A. Shrivastava. Practical near neighbor search via group testing. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 9950–9962. Curran Associates, Inc., 2021.

[27] U. Feige and A. Lellouche. Quantitative group testing and the rank of random matrices. *CoRR*, abs/2006.09074, 2020.

[28] L. Galanis, Y. Wang, S. R. Jeffery, and D. J. DeWitt. Locating data sources in large distributed systems. In J. C. Freytag, P. C. Lockemann, S. Abiteboul, M. J. Carey, P. G. Selinger, and A. Heuer, editors, *Proceedings of 29th International Conference on Very Large Data Bases, VLDB 2003, Berlin, Germany, September 9-12, 2003*, pages 874–885. Morgan Kaufmann, 2003.

[29] R. Gallager. A perspective on multiaccess channels. *IEEE Transactions on information Theory*, 31(2):124–142, 1985.

[30] O. Gebhard, M. Hahn-Klimroth, D. Kaaser, and P. Loick. Quantitative group testing in the sublinear regime. *CoRR*, abs/1905.01458, 2019.

[31] P. B. Gibbons, Y. Matias, and V. Poosala. Fast incremental maintenance of approximate histograms. *ACM Transactions on Database Systems (TODS)*, 27(3):261–298, 2002.

[32] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss. Fast, small-space algorithms for approximate histogram maintenance. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 389–398, 2002.

[33] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 454–465. Elsevier, 2002.

[34] V. Grebinski and G. Kucherov. Optimal reconstruction of graphs under the additive model. *Algorithmica*, 28(1):104–124, 2000.

[35] A. G. Greenberg, P. Flajolet, and R. E. Ladner. Estimating the multiplicities of conflicts to speed their resolution in multiple access channels. *Journal of the ACM (JACM)*, 34(2):289–325, 1987.

[36] A. G. Greenberg and S. Winograd. A lower bound on the time needed in the worst case to resolve conflicts deterministically in multiple access channels. *Journal of the ACM (JACM)*, 32(3):589–596, 1985.

[37] M. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. *ACM SIGMOD Record*, 30(2):58–66, 2001.

[38] I. F. Ilyas, W. G. Aref, and A. K. Elmagarmid. Supporting top-k join queries in relational databases. *VLDB J.*, 13(3):207–221, 2004.

[39] P. Indyk. Deterministic superimposed coding with applications to pattern matching. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 127–136. IEEE Computer Society, 1997.

[40] P. Indyk. Explicit constructions of selectors and related combinatorial structures, with applications. In D. Eppstein, editor, *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA*, pages 697–704. ACM/SIAM, 2002.

[41] P. Indyk, H. Q. Ngo, and A. Rudra. Efficiently decodable non-adaptive group testing. In M. Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1126–1142. SIAM, 2010.

[42] W. Kautz and R. Singleton. Nonrandom binary superimposed codes. *IEEE Transactions on Information Theory*, 10(4):363–377, 1964.

[43] J. Komlós and A. G. Greenberg. An asymptotically fast nonadaptive algorithm for conflict resolution in multiple-access channels. *IEEE Trans. Inf. Theory*, 31(2):302–306, 1985.

[44] K. Lee, K. Chandrasekher, R. Pedarsani, and K. Ramchandran. Saffron: A fast, efficient, and robust framework for group testing based on sparse-graph codes. *IEEE Transactions on Signal Processing*, 67(17):4649–4664, 2019.

[45] W. Liang and J. Zou. Neural group testing to accelerate deep learning. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 958–963. IEEE, 2021.

[46] S. Mallapaty et al. The mathematical strategy that could transform coronavirus testing. *Nature*, 583(7817):504–505, 2020.

[47] J. L. Massey. Collision-resolution algorithms and random-access communications. In *Multi-user communication systems*, pages 73–137. Springer, 1981.

[48] E. Porat and A. Rothschild. Explicit nonadaptive combinatorial group testing schemes. *IEEE Trans. Inf. Theory*, 57(12):7982–7989, 2011.

[49] A. Sebő. On two random search problems. *Journal of Statistical Planning and Inference*, 11(1):23–31, 1985.

[50] N. Sinnott-Armstrong, D. L. Klein, and B. Hickey. Evaluation of group testing for sars-cov-2 rna. *MedRxiv*, 2020.

[51] A. Ta-Shma, C. Umans, and D. Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In J. S. Vitter, P. G. Spirakis, and M. Yannakakis, editors, *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 143–152. ACM, 2001.

[52] S. Ubaru, S. Dash, A. Mazumdar, and O. Gunluk. Multilabel classification by hierarchical partitioning and data-dependent grouping. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22542–22553. Curran Associates, Inc., 2020.

[53] J. X. Yu, Z. Chong, H. Lu, and A. Zhou. False positive or false negative: Mining frequent itemsets from high speed transactional data streams. In *VLDB*, volume 4, pages 204–215, 2004.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Our abstract and introduction states the theoretical results that are later proven in our paper.
   (b) Did you describe the limitations of your work? [Yes] Our assumed model (including the result function) is described in Section 3.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Our result is theoretical hence does not have any direct negative impact. Also none of the existing solutions to the Group Testing (which is a field with almost 80 years of history) were harmful to the society.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] The assumptions are given in a statement of each theoretical result.
   (b) Did you include complete proofs of all theoretical results? [Yes] Some combinatorial constructions (used in proofs of Theorems 3,4,5), are included Appendix C.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]