Query-efficient computation in property testing and learning theory

García Soriano, D.

Citation for published version (APA):
Chapter 5

Group testing, non-adaptivity, and explicit lower bounds

We have been making a thorough study of the difficulty of testing function isomorphism over the previous chapters. Although we know that the complexity is $\tilde{\Omega}(n)$ in the worst case, we are yet to offer an explicit example of a function meeting the lower bound. We describe here a series of recent results that prove that $k$-parities are hard for testing isomorphism, and subsequently delve into the intimate connections between junta/isomorphism testing and the area of group testing. The content of this chapter is based on the manuscripts


5.1 Measuring the size of a parity

Parities are a natural candidate of functions for which testing isomorphism is hard. Under the additional promise that the input function is linear, testing isomorphism to a $k$-parity reduces to checking if a given linear function has size precisely $k$. (In the absence of such a promise, one can always self-correct $f$ a la BLR. This increases the complexity by a logarithmic factor, but as we shall see this is not needed.) Another way of looking at the problem is as determining, by making as few queries as possible to the Hadamard encoding of a word $x$, whether $|x| = k$ or not. Indeed, let $f = x^*$ be the parity of the bits in the support of $x$. For all $y \in \{0, 1\}^n$, we have $f(y) = \langle x, y \rangle$, which is the $y$th bit of the Hadamard encoding of $x$. So the task is essentially how to compute $|x|$ efficiently\footnote{Remarkably, this task can be accomplished in the quantum setup with just one query, by deploying the Bernstein-Vazirani algorithm [BV97] (in fact, $x$ itself can be found).} if we can query the XOR of arbitrary subsets of the bits of $x$. (Decision trees where the
queries are allowed to be XORs of subsets of the inputs have appeared in the literature \cite{ZS10}.

Deciding if the size of a parity is \( k \) is the same problem as deciding if it is \( n - k \) because of the observation leading to Lemma \ref{lem:parity}. For even \( n \), the case \( k = n/2 \) is particularly interesting because it enables us to verify the equality between the sizes of two unknown parities \( f, g \in \text{PAR}^n \). Indeed, define a parity on \( 2n \) variables by \( h(x_1, x_2) = f(1^n \oplus x_1) \oplus g(x_2) \), where \( x_1, x_2 \in \{0, 1\}^n \); then \( h \in \text{PAR}^n \) if and only if \( f \) and \( g \) are isomorphic.

A related problem is determining if a parity has size at most \( k \) (naturally, this is equivalent to the problem of deciding if the size is at least \( n - k \), or at most \( n - k - 1 \)). Upper bounds for this task imply upper bounds for testing isomorphism to \( k \)-parities (one can perform one test to verify the condition \( |x| \leq k \) and another one for \( |x| \leq k - 1 \)). Lower bounds here do not immediately imply lower bounds for testing isomorphism, but they do imply lower bounds for testing \( k \)-juntas (because one way of checking if \( f \in \text{PAR}_{\leq k} \) is testing that \( f \) is linear and also a \( k \)-junta).

The first step towards analyzing the hardness of these problems was taken by Goldreich.

5.1.1. **Theorem (Goldreich \cite[Theorem 4]{Gol10}).** Testing if a linear function \( f \in \text{PAR}^n \) (\( n \) even) is in \( \text{PAR}^n_{\leq n/2} \) requires \( \Omega(\sqrt{n}) \) queries.

5.1.2. **Theorem (Goldreich).** Testing if a linear function \( f \in \text{PAR}^n \) (\( n \) even) is in \( \text{PAR}^n_{n/2} \) requires \( \Omega(\sqrt{n}) \) queries.

The second theorem was not stated in \cite{Gol10}, but follows from the proofs therein.

He conjectured that the true bound should be \( \Theta(n) \), which will be confirmed in the next subsection. Remarkably, by using one of the other results in the very same paper, it is possible to strengthen the bound of Theorem 5.1.1 to \( \Omega(n) \), and to give a simpler proof of Theorem 5.1.2 up to polylogarithmic factors.

5.1.3. **Theorem (Goldreich).** \cite[Corollary 2.2]{Gol10} At least \( \Omega(n) \) queries are needed to distinguish a random parity on \( n \) variables from a random parity whose size is a multiple of three.

5.1.4. **Theorem (Chakraborty et al. \cite{CGM11}).** Testing if a linear function \( f \in \text{PAR}^n \) (\( n \) even) is in \( \text{PAR}^n_{n/2} \) requires \( \Omega(n) \) queries.

**Proof.** Fix \( n \) and suppose that for every \( k \leq n \) there is a tester \( A_k \) that can determine, under the assumption \( f \in \text{PAR}^n \), whether \( f \) is a parity of size \( \leq k \) using \( o(n/(\log n \log \log n)) \) queries. Then, by standard binary search and probability amplification, one could find the exact number of influential variables of \( f \) in \( o(n) \) steps. This contradicts Theorem 5.1.3 so for every \( n \) there must be some \( k = k(n) \)
such that deciding if $f \in \text{PAR}_k^n$ (for $f \in \text{PAR}^n$) needs $\Omega(n/(\log n \log \log n))$ queries. (In fact, the additional $\log \log n$ factor in the denominator can be avoided because of the results of Feige et al. [FRPU94] about noisy boolean decision trees.) If $k = n/2$ we are done. Since we can replace $k$ with $n - k - 1$, we can assume $k > n/2$.

Now we prove that determining whether some function $g \in \text{PAR}_k^n$ belongs to $\text{PAR}_k^k$ requires $\tilde{\Omega}(n/(\log n \log \log n))$ queries. We argue by contradiction. Take any tester for this property making $o(n/(\log n \log \log n))$ queries and, for each $f \in \text{PAR}^n$, let the tester’s input be the function $g_f: \{0, 1\}^{2k} \to \{0, 1\}$, where $g_f(x \uplus y) = f(x)$ for any $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^{2k-n}$. Clearly, the number of influential variables of $f$ and $g_f$ is the same and hence one would be able to determine for any $f \in \text{PAR}^n$ whether the size of $f$ is $\leq k$ or $> k$ using $o(n/(\log n \log \log n))$ queries. But this is not possible as shown in the previous paragraph.

Summarizing, for every $n$ there exists some (unknown)

$$n' = \max(2k(n), 2(n - k(n) - 1)) = \Theta(n)$$

such that determining whether $g \in \text{PAR}^{n'}$ is a parity of $\leq n'/2$ bits needs

$\Omega(n/\log n \log \log n) = \Omega(\log n'/\log \log n')$

queries. Hence the lower bound holds for infinitely many $n$. \hfill \Box

In fact, since Theorem 5.1.3 is a statement about random parities, and a random parity has size circa $n/2 \pm O(\sqrt{n})$ with high probability, one can argue (proof omitted) that this also implies an $\tilde{\Omega}(\sqrt{n})$ lower bound for checking if the size of $f$ is $n/2$, which almost matches Theorem 5.1.1.

5.1.1 Relationship to communication complexity

In a recent paper, Blais, Brody and Matulef noticed a nice connection with some well-studied problems in communication complexity. In this setup, introduced by Yao [Yao79], two parties, traditionally called Alice and Bob, each have an input and they need to devise a protocol to determine some property of the joint input. Unlimited access to their respective inputs and arbitrary computations are allowed, and the measure for the protocol’s efficiency is provided by the amount of communication they need to transmit to each other. We consider the public-coin model, whereby Alice and Bob share a common source of randomness. (See the book by Kushilevitz and Nisan [KN97] for a comprehensive treatment.)

In the $k$-set disjointness problem, Alice and Bob receive two $k$-sets $x, y \in \binom{[n]}{k}$ and would like to determine if $x \cap y = \emptyset$ or not. Furthermore, they are guaranteed that either $x \cap y = \emptyset$ or $|x \cap y| = 1$. This problem is known to have communication complexity $\Theta(k)$. The upper bound is due to Håstad and Wigderson [HW07].
The lower bound was first established by Kalyanasundaram and Schnitger [KS92], and subsequent simplifications and generalizations of the proof were found by Razborov [Raz92] and Bar-Yossef et al. [BJKS04].

5.1.5. Theorem (Blais, Brody & Matulef [BBM11]). Testing isomorphism to $k$-parities requires $\Omega(k)$ queries.

Proof. Let $k$ be even (a similar argument works for odd $k$). To solve a $k/2$-set disjointness problem with $2q$ bits of communication (and shared randomness), Alice and Bob can use a $q$-query solution for testing isomorphism to $k$-parities as indicated next. Alice forms the function $f = x^c$ and Bob forms the function $g = y^c$. Consider the function $f = (x \oplus y)^c$. Since $|x \oplus y| = |x| + |y| - 2|x \cap y|$, the function $f$ is a $k$-parity if $x \cap y = \emptyset$, and a $(k - 2)$-parity if $|x \cap y| = 1$. For every query $z \in \{0, 1\}^n$ the testing algorithm makes, Alice and Bob can use two bits of communication to make sure they both know $h(z) = f(z) \oplus g(z)$. Then they both know which query the tester makes next, because it is determined by the shared randomness and the replies to previous queries. Hence they can simulate the tester of isomorphism. In particular they can find out whether $h$ is a $k$-parity or a $(k - 2)$-parity; equivalently, they can tell whether $x$ and $y$ intersect or not, under the assumptions made.

In the contrapositive form, what we have shown is that the $\Omega(k)$ lower bound for $k/2$-set disjointness also applies to telling the case where $h \in \text{PAR}_k$ apart from the case where $h \in \text{PAR}_{k-2}$, which is always possible by a tester of isomorphism to $(k - 2)$-parities.

5.1.6. Corollary. The query complexity of testing isomorphism to $k$-parities is $O(k \log k)$ and $\Omega(k)$.

Of interest to us in the rest of the chapter is what the situation looks like when we ask for non-adaptive algorithms. We will see that the query complexity becomes $\Theta(k \log k)$ in this case, and we conjecture that this bound remains valid in the adaptive case.

5.2 Background on group testing

The field of group testing seeks efficient procedures to identify a set of defective items by performing “batch” tests, whereby a collection of items are tested together to determine if the batch contains a defective item or not. The subject originated during the Second World War as a means of detecting diseases in soldiers’ blood samples [Dor43], without having to test each sample individually. The problem

---

2Despite its name, group testing is not concerned with property testing or group theory.
has since emerged time and again in a great many different contexts, theoretical as well as applied, including design theory, error-correcting codes, DNA screening, etc. A good textbook on the topic has been written by Du and Hwang \cite{DH00}; somewhat related to our work are the papers \cite{KS64,BGV05,PR08,INR10}.

We formalize the problem as follows. We are given black-box access to a function $f: \{0, 1\}^n \rightarrow \{0, 1\}$ of the form

$$f(x_1, \ldots, x_n) = \bigvee_{i \in R} x_i,$$

for some unknown set $R$ (the set of relevant variables of $f$). (These are the “defective items”.) We abbreviate this as $f = \text{OR}_R$ and also write

$$\text{OR}_k = \bigcup_{R \subseteq [n], |R| = k} \{\text{OR}_R\}$$

and

$$\text{OR}_{\leq k} = \bigcup_{0 \leq k' \leq k} \{\text{OR}_{k'}\}.$$

One can regard queries to $\text{OR}_R$ as questions of the form

“Is $|X \cap R| = 0$ or $|X \cap R| \geq 1$?"

for any $X \subseteq [n]$. An upper bound $k$ on the size of $R$ is given, and the goal is to determine the elements of $R$ while attempting to minimize the number of queries made.

Most prior work on group testing comes in two flavors, probabilistic and deterministic. The probabilistic view imposes a distribution on the location of the $k$ relevant variables (see, e.g., \cite{Mac98}), while in the deterministic setting this is arbitrary. We work in the latter setting. A lot of effort has been put into optimizing the constants involved, but we will focus on asymptotics as usual.

In the adaptive case, tight bounds of $\Theta(k \log(n/k))$ can easily be obtained in various ways (we are assuming $k \leq n/2$, otherwise $\Theta(n)$ bounds are trivial). For example, for the upper bound one can query the sets $[1], [2], [4] \ldots$, until a relevant set has been found; then it is possible to perform a binary search for the index of the first relevant variable. The set $R$ can thus be determined after $k$ iterations of this process, and a somewhat careful analysis that we omit shows that this makes $O(k \log(n/k))$ queries. (Another approach would be to use the Winnow algorithm of Littlestone \cite{Lit88} with random samples.) On the other hand, the lower bound is immediate from an information-theoretical argument, as there are $\binom{n}{k} = 2^{\Omega(k \log(n/k))}$ possible $k$-subsets of $[n]$.

Therefore, our discussion of standard group testing in this section is limited to non-adaptive algorithms. Associated with a deterministic, non-adaptive algorithm is a binary query matrix $M \in \mathcal{M}_{q \times n}$ whose rows $r_1, \ldots, r_q \in \{0, 1\}^n$ are the
indicator vectors of each of the (non-adaptive) queries made. For two binary vectors \( a, b \in \{0, 1\}^n \), recall that \( a \lor b \) represents their bitwise disjunction (which can be thought of as the union of two sets), and \( a \land b \) represents their bitwise conjunction (the intersection of two sets). When \( f = \text{OR}_R \), the response vector

\[
M \cdot R \triangleq (f(r_1), \ldots, f(r_q))^T
\]

is then simply the disjunction of the columns of \( M \) indexed by \( R \), which can be interpreted as the boolean product of matrix \( M \) and a column vector \( R \) (the characteristic vector of the set of relevant variables), where +, \cdot correspond to \( \lor \), \( \land \), respectively. (Note that this departs from our convention in the rest of the thesis that \( a + b \) denotes addition over \( \mathbb{F}_2^n \).) Observe that being able to determine \( R \) from \( M \cdot R \) under the promise \( |R| \leq k \) is equivalent to the following property being satisfied:

- the OR (union) of each set of \( \leq k \) columns of \( M \) is unique, in the sense that \( M \cdot R \neq M \cdot R' \) for any \( R \neq R' \) with \( |R|, |R'| \leq k \).

Such a matrix is called \( k \)-separable. Its columns are sometimes said to form a uniquely decipherable code of order \( k \). This name reflects the connection, exploited by Kautz and Singleton [KS64], with certain kind of codes. Namely, the property above guarantees unique decodability of \( R \) from the response vector of the corresponding \( \leq k \)-OR, although the procedure may take as much as \( \Omega(n^k) \) time. Most of the constructions of \( k \)-separable matrices possess a slightly stronger property that allows for quick decoding:

- the OR (union) of any set of \( \leq k \) columns of \( M \) does not contain another column of \( M \),

or equivalently

- every subset of \( k + 1 \) columns of \( M \) contains a (permuted) \( (k + 1) \)-sized identity submatrix.

Such a matrix \( M \) is called \( k \)-strongly selective, or \( k \)-disjunct. The set of its columns is also called a \( k \)-cover-free family, \( k \)-union-free family, or a zero-false-drop code of order \( k \) (more variations include “disjunctive codes” and “superimposed codes”). It is readily seen that any \( k \)-disjunct matrix is \( k \)-separable, and that any \( k \)-separable matrix is \((k - 1)\)-disjunct: if \( M \) is not \((k - 1)\)-disjunct, then it has a subset \( S \) of \( k - 1 \) columns whose union includes the \( j \)th column for some \( j \notin S \); but then \( M \cdot S = M \cdot (S \cup \{j\}) \), so \( M \) is not \( k \)-separable.

Given a \( k \)-disjunct matrix, an efficient decoding algorithm runs as follows. Observe that every negative answer \( (f(r_i) = 0) \) to a query \( r_i \) implies that no element of \( r_i \) (viewed as a subset of \([n]\)) belongs to \( R \) as well. This allows us to label some columns as irrelevant, for each query with a negative answer. We can
safely discard these columns as no element of \( R \) will be removed in this way. The definition of \( k \)-disjunctness simply asserts that after going through the responses to all queries and discarding all such columns, we are left with precisely the columns associated with \( R \). In short, 
\[
R = [n] \setminus \bigcup_{i \in [n]} r_i \quad \text{OR}_{R(r_i) = 0}
\]
holds for any \( R \in \binom{n}{k} \) (recall that we are identifying elements of \( \{0, 1\}^n \), such as \( r_i \), with subsets of \([n]\)).

Constructions are known for \( k \)-disjunct matrices where the number of rows (which translates into the number of queries) is \( q = O(k^2 \log n) \) \cite{KS64,BV03}. The best known lower bounds are \( q = \Omega(k^2 \log n / \log k) \) \cite{Fur96,EFF82,Rus94}; the proof can also be found in \cite[p. 116]{Juk11}.

**5.3 Relaxed group testing: adaptive**

We study the following relaxation of the group testing problem. Instead of trying to find \( R \) itself, suppose we only want to distinguish between the cases \(|R| \leq k\) or \(|R| > k\). We assume \( k \leq n/2 \) because tight bounds are easy otherwise. This relaxation does not appear to help much if we restrict ourselves to deterministic algorithms (see below), but the situation looks vastly different if randomization is allowed. As usual we consider algorithms with success probability, say, 2/3.

A straightforward \( O(k \log k) \) adaptive upper bound for relaxed group testing follows by using the random partitions introduced in Definition 3.4.1: start from a random partition of \([n]\) into \( \sim k^2 \) buckets and then simulate the deterministic, adaptive group testing upper bound for \( n = O(k^2) \), where single variables have been replaced with buckets.

By now the astute reader will have noted a striking resemblance between group testing and junta testing. All the testers we discussed in Section 3.6 used independence tests as a building block, and this is the only way they accessed \( f \) (except for the random choices of \( x \) and \( y \) with \( f(x) \neq f(y) \)). Suppose \( f : \{0, 1\}^n \to \{0, 1\} \) is a junta with set \( R \) of relevant variables. With \( f \) we can associate the function \( f' = \text{OR}_R \) that takes the union of the bits in \( R \). An independence test of \( f \) on \( A \) is roughly the same as an \( \text{OR} \) query for \( f \) on \( A \), except that sometimes \( A \) may be incorrectly reported as non-influential by an independence test. So by replacing independence tests with calls to \( f' \), the one-sided junta testers immediately yield algorithms that accept if \( f \in \text{OR}_{\leq k} \). The rejection conditions being different, it would appear that there is no guarantee that they reject if \( f = \text{OR}_R \) where \(|R| > k\), but they actually do.

For example, take the adaptive junta tester from Section 3.6. As long as \( k \) or fewer relevant blocks have been detected, it conducts an influence test on the rest,
and if the rest is found to be relevant, it binary searches for one more additional block. If this process is repeated $k + 1$ times, then it will find $k + 1$ relevant blocks for any OR of more than $k$ variables, provided the initial random partition separates at least $k + 1$ of the elements of $R$, which occurs with high probability. Thus we arrive at a second $O(k \log k)$ adaptive solution for relaxed group testing; in fact, the algorithm finds a $k$-isolating partition for $R$ (see Definition 3.4.1).

An $\Omega(k)$ adaptive lower bound can be proven for relaxed group testing in exactly the same way as for parities (Section 5.1.1). To see this, observe that to solve a set-disjointness problem, Alice can form the function $f = OR_x$ and Bob the function $g = OR_y$, and then simulate queries to $h \triangleq f \lor g$. (Of course, the same kind of argument works for ANDs just as well as it does for XORs and ORs.) All in all we have the following.

5.3.1. Theorem. The adaptive query complexity of the relaxed group testing problem with parameter $k \leq n/2$ is $\Omega(k)$ and $O(k \log k)$.

5.3.2. Remark. If we demand deterministic algorithms (which are always correct), then the adaptive complexity of the problem becomes $\Theta(\log\binom{n}{k})$. The lower bound uses the same technique, since the reduction above then gives a deterministic protocol for the $k/2$-set disjointness problem, the communication complexity of which is $\lceil \log(\binom{n}{k/2}) \rceil = \Theta(\log\binom{n}{k})$ because of the rank bound [KN97, Example 2.12].

5.3.1 Interlude: a 3-way variant

Interestingly, were we endowed with the power to perform a somewhat stronger test of relevance, the problem would admit faster adaptive solutions. Namely, suppose that querying for a set $A$ allowed us to distinguish with certainty from among the three cases $|A \cap R| = 0$, $|A \cap R| = 1$ and $|A \cap R| \geq 2$, instead of only the first from the other two. Then we claim that the problem could be solved with $O(k)$ queries, which gets rid of the $\log k$ factor. One notable consequence of this fact is that many lower bound techniques with an information-theoretic flavor cannot go beyond $\Omega(k)$ for this problem, as there is a randomized ternary decision tree with $O(k)$ depth that solves it.

To see this, observe that there is nothing to do if the whole set $[n]$ is not relevant, so assume it is. We keep a partition $\mathcal{P}$ of $[n]$ such that each element of $\mathcal{P}$ intersects $R$. Initially, $\mathcal{P}$ is the singleton $\{[n]\}$. At each step we take an arbitrary $A \in \mathcal{P}$ such that $|A \cap R| \geq 2$ (if there is one), and take a random partition of $A$ into two sets $A_1, A_2$. With probability no less than $1/2$, at least one relevant index in $A$ lands into each of $A_1, A_2$; otherwise we draw another random pair (this condition can be tested with two queries to the function). When we succeed in splitting $A$ into two relevant sets $A_1, A_2$, we replace $A$ with $A_1$ and $A_2$ in the partition to form $\mathcal{P}_{\text{next}} = \mathcal{P}_{\text{prev}} - \{A\} \cup \{A_1, A_2\}$, and go back to picking another element of $\mathcal{P} = \mathcal{P}_{\text{next}}$ if possible.
5.4 Relaxed group testing: non-adaptive

Intuitively, the reason why this works is that if we know for sure that $|A \cap R| \geq 2$, then we know that it pays to persist in attempting to split $A$: we will succeed promptly. Notice that if we have a test for $|A \cap R| \geq 2$ that works only with high constant probability, then $O(\log k)$ iterations decrease the error probability of each such test to $O(1/k)$ (this is necessary to handle those $A$ with $|A \cap R| = 1$). This leads to a third constant-success probability solution to the relaxed group testing problem. Again, it has complexity $O(k \log k)$.

An unexpected application of these ideas, modulo some tailoring, will be exhibited in Chapter 9 on cycle finding.

5.4 Relaxed group testing: non-adaptive

For non-adaptive testers, these algorithmic techniques do not seem to work. Nevertheless, we show in the following section that, somewhat surprisingly, the task can still be accomplished non-adaptively with $O(k \log k)$ queries (Theorem 5.4.7). (Note that when $n = O(k^2)$, the best $k$-disjunct matrices have size between $O(k^2 \log k)$ and $\Omega(k^2)$, so applying the non-adaptive group testers to a random partition results in much worse query complexities.) Additionally, the algorithm can handle random errors in the response vector, in the form of false negatives. This sets the stage for applications to property testing (Section 5.5).

We can show that $\Omega(k \log k)$ non-adaptive queries are necessary (Theorem 5.4.1 and its corollary). This applies even to the noiseless setting, thus showing that our non-adaptive algorithm is optimal. It is also easy to derive from it an $\Omega(k \log k)$ lower bound for testing $k$-juntas non-adaptively. We should also point out that our upper bound for relaxed group testing actually finds $k$ blocks each containing at most one relevant variable; it is also possible to argue that each of these blocks has size roughly $n/k$. Once we know a block has one relevant variable, we can use standard group testing with $k = 1$ to find the variable itself with $O(1 \log(n/1))$ non-adaptive queries. Hence our result also gives a 2-stage randomized solution to the standard group testing problem with complexity $O(k \log k + k \log(n/k)) = O(k \log n)$. However, this is already known, even deterministically (see [BGV05, BV06, MT11], which are based on [CGR02]).

5.4.1 Lower bound

The first observation we need for the non-adaptive lower bound is the following. In Section 5.3 we showed how to use a $q$-query solution to relaxed group testing to yield a solution to set disjointness with $2q$ bits of communication. But note that if the algorithm is non-adaptive, then we can obtain a solution to set disjointness with only $q + 1$ bits of communication, all of which directed from Alice to Bob. This is because there is no need for both parties to know $h(z)$, since the whole set of queries is determined already by the common source of randomness and
each of them knows what comes next. Bob can perform all the computations and figure out the answer. Hence it is enough to prove lower bounds for the one-way communication complexity of the set disjointness problem.

5.4.1. Theorem (de Wolf [Wol06]). The one-way communication complexity of the $k$-set disjointness problem is $\Theta(k \log k)$ for $k = O(\sqrt{n})$, and $\Theta(k \log(n/k))$ for $\Omega(\sqrt{n}) \leq k \leq n/2$.

(It is $\Theta(n)$ for $k \geq n/2$.)

Proof. The upper bound follows from the relaxed group testing upper bounds, so we only need to prove the first part. We assume for simplicity that $k$ is a power of two and divides $n$.

First consider the case $k \leq \sqrt{n}/2$. Let $x$ be Alice’s $n$-bit input. For Alice we restrict our attention to inputs of a particular structure. Namely, consider partition of $[n]$ into $k$ consecutive sets of size $n/k \geq 2k$. The inputs we allow contain precisely a bit set to one inside each block of the partition, and moreover the offset of the unique index set to one within the $i$th block is a number between 0 and $2k - 1$, inclusive. In this case, $x$ describes a message $M$ of $k$ integers $m_1, \ldots, m_k$ in the interval $\{0, \ldots, 2k - 1\}$. This is an $m$-bit long message, where $m = k \log(2k)$. We can write Alice’s input as $x = u(m_1) \ldots u(m_k)$, where $u(m_i) \in \{0, 1\}^{n/k}$ is the unary expression of the number $m_i$ using $n/k$ bits (where the rightmost $n/k - k$ bits of each $u(m_i)$ are always zero). For instance, the picture below illustrates the case where $n = 40$, $k = 4$, and $M = (1, 7, 0, 5)$:

\[
\begin{array}{cccc}
\text{block } & \text{offset} & \text{bit set to one} \\
1 & 1 & 0100000000 \\
2 & 2 & 0000001000 \\
3 & 3 & 1000000000 \\
4 & 3 & 0000010000 \\
\end{array}
\]

Let $\rho_x$ be the $q$-bit message that Alice sends on this input. Below we show that the message is a random-access code for $M$, i.e., it allows a user to recover each bit of $M$ with probability at least $1 - \delta$ (though not necessarily all bits of $M$ simultaneously). Then our lower bound will follow from Nayak’s random-access code lower bound [Nay99]. This says that

\[ q \geq (1 - H(\delta))m, \]

where $\delta$ is the error probability of the protocol and $H(\delta)$ is its entropy.

Suppose Bob is given $\rho_x$ and wants to recover some bit of $x$. Say this bit is the $\ell$th bit of the binary expansion of $m_i$. Then Bob completes the protocol using the following $y$: $y$ is 0 everywhere except on the $k$ bits in the $i$th block of size $n/k$ whose offsets $j$ (measured from the start of the block) satisfy the following: $0 \leq j < 2k$ and the $\ell$th bit of the binary expansion of $j$ is 1.

Recall that Alice has a 1 in block $i$ only at position $m_i$. Hence $x$ and $y$ will intersect iff the $\ell$th bit of the binary expansion of $m_i$ is 1, and moreover, the
size of the intersection is either 0 or 1. Also, the Hamming weight of \( y \) is \( k \) by definition. Running the \( k \)-set disjointness protocol with confidence \( 1 - \delta \) will now give Bob the sought-for bit of \( M \) with probability at least \( 1 - \delta \), which shows that \( \rho_x \) is a random-access code for \( M \).

If \( k > \sqrt{n/2} \) then we can do basically the same proof, except that the integers \( m_i \) are now in the interval \( \{0, \ldots, n/k - 1\} \), \( m = k \log(n/k) \), and Bob puts only \( n/2k < k \) ones in the \( i \)th block of \( y \) (he can put his remaining \( k - n/2k \) indices somewhere at the end of the block, at an agreed place where Alice won’t put 1s). This gives a lower bound of \( \Omega(k \log(n/k)) = \Omega(\log(n/k)) \).

5.4.2. Remark. The lower bound holds even for quantum one-way communication complexity. The proof remains intact except that if we wish to allow Alice and Bob to share entanglement, the random-access code bound needs to be replaced with Klauck’s [Kla00] version, which is weaker by a factor of two.

5.4.3. Corollary. Let \( k \leq \sqrt{n} \). At least \( \Omega(k \log k) \) queries are needed for non-adaptive solutions to the following problems: relaxed group testing with parameter \( k \), testing \( k \)-juntas and testing isomorphism to \( k \)-parities.

5.4.2 Upper bound

Here we present an algorithm to solve the relaxed group testing problem in the presence of one-sided noise of some constant rate \( \eta > 0 \). Let \( x \in \{0, 1\}^n \) and \( f = \text{OR}_x \). Suppose that, instead of querying \( f \), we can only query a “noisy version” \( \tilde{f} \) of \( f \). Let \( \tilde{f} \) denote a function on \( \{0, 1\}^n \) whose image on any input is a \( 0 - 1 \) random variable, and the variables \( \{f(x)\}_{x \in \{0,1\}^n} \) are independent. Then \( \tilde{f} \) is called a \( \eta \)-noisy oracle for \( f \) if the following two properties are satisfied:

1. \( f(x) = 0 \) implies \( \tilde{f}(x) = 0 \);
2. \( f(x) = 1 \) implies \( \Pr[\tilde{f}(x) = 1] \geq 1 - \eta \).

Different calls to \( \tilde{f}(x) \) for the same \( x \) return independent copies of the same random variable, may yield different values.

Recall that we denote the boolean product of a \( q \times n \) matrix \( M \) and \( x \in \{0, 1\}^n \) by \( M \cdot x \). When \( f = \text{OR}_x \), the response vector \( a \in \{0, 1\}^q \) under \( f \) will be \( a = M \cdot x \). One equivalent way of modelling the response vector \( \tilde{a} \) under a \( \eta \)-noisy oracle \( \tilde{f} \) for \( f \) as defined above is as follows. Let \( e \in \{0, 1\}^q \) be a sequence \( q \) of independent \( 0 - 1 \) variables such that \( \Pr[e_i = 1] \geq 1 - \eta \) and let \( \tilde{a} = a \land e \). Our goal is to design \( M \) so as to allow, with high probability, determining if \( |x| = k \) given \( \tilde{a} \). We call this the noisy relaxed group testing problem.

We introduce the following variation of the notion of \( k \)-disjunct matrix.
5.4.4. DEFINITION. A \( q \times n \) binary matrix \( M \) is \((\delta, \zeta)\)-approximately \( k \)-disjunct if, with probability at least \( 1 - \delta \), the union of a random \( k \)-subset of the columns of \( M \) and a vector in \( \{0, 1\}^q \) of weight at most \( \zeta q \) \( \{0, 1\}^q \) does not contain any other column of \( M \).

The reader may verify that \((\delta, \zeta)\)-approximate \( k \)-disjunctness implies \((\delta, \zeta)\)-approximate \((k - 1)\)-disjunctness as well. The property can be rewritten as

\[
(M \cdot x) \lor y \not\supseteq M \cdot z
\]

with probability \( 1 - \delta \), for random \( x \in \{0, 1\}^n \) with \( |x| = k \) (or \( |x| \leq k \)) and any \( y, z \in \{0, 1\}^q \) with \( |y| \leq \zeta q \) and \( z \subseteq [n] \setminus x, |z| = 1 \).

We need the following lemma.

5.4.5. LEMMA. Let \( n = O(k^2) \) and \( R \) be a \( k \)-subset of \([n]\). Let \( V \) be an \((60 \log k) \times n\) random matrix with entries in \([10]\).

For every column index \( j \in [n] \setminus R \), let \( X_j \) be the number of row indices \( i \in [60 \log k] \) for which there exists another column \( j' \in R \) with \( V_{ij} = V_{ij'} \). Then

\[
\Pr_{V} \left[ \forall j \in [n] \setminus R : X_j < 15 \log k \right] = 1 - o(1).
\]

**Proof.** Let \( j \in [n] \setminus R \). For each \( i \in [60 \log k] \),

\[
\Pr \left[ \forall j' \in R : V_{ij} \neq V_{ij'} \right] = (1 - 1/(10k))^k \geq 9/10.
\]

Hence \( X_j \) is the sum of \( 60 \log k \) independent binary random variables with expectation \( \leq 1/10 \), and by the Chernoff bounds

\[
\Pr [X_j \geq 15 \log k] \leq \exp(-(15/6 - 1)^2(1/10)60 \log k/3) = o(k^{-2}).
\]

The claim follows from the union bound over all \( n - k = O(k^2) \) elements of \([n] \setminus R \).

5.4.6. COROLLARY. Let \( n = O(k^2) \). There is a \((1 - o(1), 3/32)\)-approximately \( k \)-disjunct matrix \( M \in \mathcal{M}(600k \log k)^{n} \). Furthermore, each of the columns of \( M \) has Hamming weight \( 60 \log k \).

**Proof.** We use the notation of Lemma 5.4.5. Let \( V \) be a matrix such that

\[
\Pr_{R} \left[ \forall j \in [n] \setminus R : X_j < 15 \log k \right] = 1 - o(1);
\]

such \( V \) can be shown to exist by the said lemma and a straightforward averaging argument. Let \( V \) represent a code of \( n \) words of length \( 60 \log k \) over an \( 10k \)-sized alphabet. Concatenate it with the “trivial” code mapping each symbol \( x \in [10k] \) to the string \( 0^{x-1}10^{10k-x} \in \{0, 1\}^{10k} \) and let \( M \) be an \((600k \log k) \times n\) matrix.
whose columns are the words of the concatenated code. The claim follows by observing that

\[ X_j = \left| \{ i \in [600 \log k] \mid M_{ij} = 1 \land (\exists j' \in R \mid M_{ij} = 1) \} \right|. \]

\[ \tag*{\Box} \]

5.4.7. Theorem. There is a non-adaptive algorithm making \( O(k \log k) \) queries that solves the noisy relaxed testing problem with noise rate \( \eta = 1/2 \). It succeeds with high constant probability and has two-sided error.

**Proof.** Let \( \ell = 5(k + 1)^2 \). First we show how to reduce the general case to the case \( n < \ell \). When \( n \geq \ell \) the algorithm proceeds as follows:

1. Partition \([n]\) into \( \ell \) buckets \( I_1, \ldots, I_\ell \) at random.
2. Define the function \( \text{replicate} : \{0, 1\}^\ell \rightarrow \{0, 1\}^n \) by mapping \( x \in \{0, 1\}^\ell \) to the string \( y \in \{0, 1\}^n \) that is constant inside each bucket and takes value \( x_i \) on each element of \( I_i \).
3. Solve the relaxed noisy group testing problem for the composition of \( f \) and \( \text{replicate} \).

Let \( f = \text{OR}_x \) for some \( x \subseteq [n] \). Then \( f \circ \text{replicate} \) will be of the form \( \text{OR}_{x'} \) for some subset \( x' \subseteq [\ell] \). The size of \( x' \) will always be at most that of \( x \); in particular it will be bounded by \( k \) whenever \( |x| \leq k \). When \( |x| \geq k + 1 \), standard arguments based on the birthday paradox yield \( |x'| \geq k + 1 \) with probability at least 9/10. So the reduction does indeed work with high constant probability.

Let us deal first with the case \( n = \ell \) (note we can always add irrelevant variables if \( n < \ell \)). We do the following.

1. Obtain a \((600k \log k) \times \ell \) matrix \( M \) as in Corollary 5.4.6.
2. Randomly permute the columns of \( M \) to obtain \( M^\pi \).
3. For each row \( r_i \) of \( M^\pi \), query \( f \) on \( r_i \); let \( \tilde{a}_i \) be the answer.
4. Compute \( X_j \triangleq \left| \{ i \in [600 \log k] \mid M_{ij}^\pi = 1 \land \tilde{a}_i = 1 \} \right| \) for each column index \( j \in [\ell] \).
5. Let \( \tilde{x} \triangleq \{ j \in [\ell] \mid X_j \geq 15 \log k \} \).
6. Accept iff \( |\tilde{x}| \leq k \) (and if so, return the nonzero indices of \( \tilde{x} \)).
The fact that a random permutation is being applied to each of the rows of $M$ enables us to analyze $x$ as a uniformly random \(|x|\)-set, while leaving $M$ unpermuted. Because of the one-sided nature of the error, $\tilde{a}_i = 1$ implies $r_i \cap x \neq \emptyset$. It is then immediate from the definition of approximate $k$-disjunct matrices that, with probability $1 - o(1)$ over $x$ of size $k$, for any $j \in [\ell] \setminus x$, it holds that $X'_j < 15 \log k$ and hence $j \notin \tilde{x}$. This shows $k$-ORs are accepted with high probability.

We claim now that, for any $x \in [\ell]$, and with probability $1 - o(1)$, $x \subseteq \tilde{x}$ holds; in particular the algorithm rejects $k'$-ORs when $k' > k$. Indeed, each column of $M$ has $60 \log k$ nonzero entries. For every $j \in x$ and row $i$ containing a nonzero entry (that is, $M_{ij} = 1$), $\Pr[\tilde{a}_i = 1] \geq 1 - \eta \geq 1/2$. By the Chernoff bound, the probability that the number of $i$ with $M_{ij} = 1$ and $\tilde{a}_i = 1$ is smaller than $15 \log k$ is bounded by $\exp((-1/2)^230 \log k/2) = o(k^2)$. It follows by the union bound over all $j \in x$ that $x \subseteq \tilde{x}$, proving our claim.

This procedure also can be used to distinguish $k$-ORs from any $k'$-OR with $k' \neq k$, whether $k' > k$ or $k' < k$.

### 5.5 Strong $k$-juntas

Recall that an independence test for $S \subseteq [n]$ is performed as follows. Pick $\ln(1/\eta)/\delta$ random pairs $x, y \in \{0, 1\}^n$ conditioned on $x|_{[n]\setminus S} = y|_{[n]\setminus S}$ and return $\tilde{f}'(S) = 1$ if $f(x) \neq f(y)$ for some pair $(x, y)$, or else return $\tilde{f}'(S) = 0$. Independence tests have one-sided error, so all errors are in the form of false negatives, i.e., the reported value $\tilde{f}'(x)$ is 0 but the true value of $f'(x)$ is 1. In fact it is clear that

1. $\text{Inf}_f(S) = 0$ implies $\tilde{f}'(S) = 0$;
2. $\text{Inf}_f(S) \geq \delta$ implies $\Pr[\tilde{f}'(S) = 1] \geq 1 - \eta$.

We call such $\tilde{f}'$ a $\eta$-noisy relevance oracle for $f$. As we saw, all known testers for $k$-juntas can be viewed as making queries to such a relevance oracle.

Observe that the second proviso is met whenever $S$ contains a variable of influence $\geq \delta$, because influence is monotone. In fact, if $f$ has a subset $S$ of least $k$ variables of influence $\geq \delta$ each, then $\tilde{f}$ is a $\eta$-noisy oracle for $\text{OR}_S$ as defined in the previous section. (The oracle makes $O(\log(1/\eta)/\delta)$ queries to the function $f$.) This motivates the following definition:

#### 5.5.1. DEFINITION. A $\delta$-strong $k$-junta is a function $f: \{0, 1\}^n \to \{0, 1\}$ with $k$ variables of influence $\geq \delta$ and $n - k$ variables of influence 0.

Up to a factor of at most 2 in $\delta$, this is essentially the same as a $k$-junta which is $\delta$-far from all juntas on fewer than $k$ variables (Lemma 3.6.1). It may be worthwhile to note that any $k$-junta with Fourier degree $d$ is a $2^{-(d+1)}$-strong $k$-junta by a result of Nisan and Szegedy [NS92].)
5.5. Strong $k$-juntas

As usual, when we speak of a $\delta$-strong $\leq k$-junta, we mean a $\delta$-strong $k'$-junta for some $k' \leq k$.

As a consequence of the discussion above we obtain the following result: there is an algorithm making $O\left(\frac{k}{\delta} \log k\right)$ non-adaptive queries that, with high probability, accepts $\delta$-strong $\leq k$-juntas and rejects functions with at least $k + 1$ variables of influence $\geq \delta$. We can do better however:

5.5.2. THEOREM. There is a non-adaptive $O\left(\frac{k \log k}{\min(\delta, \varepsilon)}\right)$-query tester that, with high probability, accepts $\delta$-strong $\leq k$-juntas and rejects functions that are $\varepsilon$-far from all $k$-juntas.

Roughly speaking, the key feature of strong $k$-juntas that allows simpler testing algorithms is that every relevant variable can be easily identified, eliminating the need to tackle sets of variables that account for a noticeable fraction of the function’s overall influence despite each variable having individually low influence. This allows us to strengthen (and simplify) the acceptance conditions.

**Proof.** Suppose for the moment that $n = O(k^2)$ (so no partitioning is necessary). As we saw, it is possible to simulate a query to a $(1/2)$-noisy relevance oracle $f'$ for $f$ with $O(1/\delta)$ queries to a $\delta$-strong $k$-junta $f$. Then we apply the test of Theorem 5.4.7 to solve the noisy group testing problem with $O(k \log k)$ queries to $f'$ and reject if it rejects. This makes $O(k \log k/\delta)$ queries to $f$. Let $S \subseteq [\ell]$, $|S| \leq k$ be the set of relevant variables returned. If $f$ is a $\delta$-strong $k$-junta on $R$, then with high probability $R = S$, $f' = \text{OR}_S$, and this first step of our algorithm accepts.

The second part is designed to make the algorithm reject when $f$ is $\varepsilon$-far from a $k$-junta, in which case we have $\text{Inf}_f([n] \setminus S) \geq \varepsilon$. (Of course, we also need to ensure that $\delta$-strong $k$-juntas are accepted, but this will follow trivially from the fact that $\text{Inf}_f([n] \setminus S) = 0$ then.) Let $A = [n] \setminus S$. We construct a (multi)set $T \subseteq \{0,1\}^n$ of $100k \log k/\varepsilon$ inputs with density $1/k$, i.e., each of the queries $B$ is drawn from $\subseteq_{1/k} [n]$ as in Lemma 3.6.4. For each $B \in T$, we query two random strings $x^B, y^B \in \{0,1\}^n$ conditioned on $x^B |_{[n] \setminus B} = y^B |_{[n] \setminus B}$ and reject if there is some $B$ disjoint with $S$ for which $f(x^B) \neq f(y^B)$.

We expect $100k \log k/\varepsilon$ of the elements of $T$ to be all zero inside $S$, and by Chernoff bounds, with probability $1 - o(1)$ there are at least $96k \log k/\varepsilon$ like that. (We assume $k = \omega(1)$ or it is at least large enough; otherwise the theorem is easy.) Let $Q = \{B \in T \mid B |_{S} = 0\}$ be this multiset of size $\geq 96k \log k/\varepsilon$. Note that the distribution of $B \subseteq_{\rho} [n]$ conditioned on $B |_{S} = 0$ is exactly the same as the distribution of each element of $Q$ follows exactly $\subseteq_{1/k} A$.

When taking $B, x, y$ conditioned on $B |_{S} = 0$ and $x^B |_{[n] \setminus B} = y^B |_{[n] \setminus B}$, note that the probability that $f(x^B) \neq f(y^B)$ is exactly equal to $\mathbb{E}_{B \subseteq_{1/k} [n] \setminus S} \text{Inf}_f(B)$. 


Chapter 5. Group testing, non-adaptivity, and explicit lower bounds

Therefore, by Lemma 3.6.5, if \( f \) is \( \varepsilon \)-far from being a junta on \( S \), then each independence test associated with an element of \( B \) rejects with probability at least \( \varepsilon/k \). Since \( |B| = \Omega(k \log k/\varepsilon) \), in this case we reject with probability \( 1 - o(1) \), and we are done.

For larger \( n \), the argument is the same except that we start with a random partition of \( n \) into \( \ell = O(k^2) \) blocks and make blockwise-constant queries of density \( 1/k \). From then on we consider functions defined on \( \ell \)-bit inputs, as in Chapter 3; there we used \( \ell = O(k^{9/\varepsilon}) \), but a recent result of Blais et al. [BWY11] shows that we can in fact take \( \ell = O(k^2) \).

One consequence of the proof is that if \( f \) is a \( \delta \)-strong \( k' \)-junta (with \( k' \leq k \)), we return \( k' \) blocks that isolate the relevant variables of \( f \). Hence by using the techniques of Chapter 3 we can test isomorphism to to \( \Omega(1) \)-strong \( k \)-juntas with \( O(k \log k/\varepsilon) \) non-adaptive queries.

However, it seems that the following recent result may yield non-adaptive testers of isomorphism to any function that make \( O(k \log k/\varepsilon) \) queries.

5.5.3. Theorem (Ron & Tsur [RT11, Theorem 3.1]). There is a non-adaptive algorithm that makes \( O(k/\varepsilon) \) queries, accepts \( k \)-juntas and rejects functions far from \( 2k \)-juntas (with high probability).

The reason is that to be able to apply the techniques of Chapter 3, we do not need to reject functions that are far from \( k \)-juntas, as the junta testers do. We can get by rejecting functions that are far from \( 2k \)-juntas, which is possible by Theorem 5.5.3. One can then relate in a similar way the distance from \( f \) to being a \( 2k \)-junta to the distance to the “blockwise” version of \( f \) to being a \( 2k \)-junta. However, the resulting testers would have two-sided error.

5.6 Parities and SMP complexity

There is a recent paper of Leung, Li, and Zhang [LLZ11] whose main result follows immediately from our \( O(k \log k) \) non-adaptive upper bound for parities, and in fact is improved by it.

The problem is about communication complexity: Alice has a string \( x \in \{0, 1\}^n \), Bob has \( y \in \{0, 1\}^n \), and they want to compute some function \( f \) that depends only on \( |x \oplus y| \), i.e., \( f(x \sqcup y) = D(|x \oplus y|) \), for some known \( D: [n] \to \{0, 1\} \). They say such an \( f \) is a symmetric XOR function. The model is the Simultaneous Message Passing (SMP) model, which means Alice and Bob can’t talk to each other; rather, they send messages to a referee, who computes the answer at the end.

Suppose first we are promised that \( |x \oplus y| \leq k \). They proceed to give an \( O(k \log k / \log \log k) \) upper bound for the problem of finding out \( |x \oplus y| \). It is not hard to see, however, that our non-adaptive algorithm for finding the size
of a XOR gives an $O(k \log k)$ solution. Take $Q = \{p_1, p_2, \ldots, p_q\}$ to be the set of $q = O(k \log k)$ random queries used to find the size of a parity, provided it is at most $k$. Then Alice sends the bits $\langle x, p_1 \rangle, \langle x, p_2 \rangle, \ldots, \langle x, p_q \rangle$ (where the inner products are taken modulo 2), and Bob sends $\langle y, p_1 \rangle, \langle y, p_2 \rangle, \ldots, \langle y, p_q \rangle$. The referee then computes from this $\langle x \oplus y, p_1 \rangle, \langle x \oplus y, p_2 \rangle, \ldots, \langle x \oplus y, p_q \rangle$, and this is enough to compute the size of $x \oplus y$, if it is $\leq k$.

They state the main result, when $k$ is not known, in terms of a different parameter $r$, but in the paper it follows from a reduction to the previous problem with $r \leq k$. Using this reduction we obtain the following.

**5.6.1. Theorem.** Define $r_0$ and $r_1$ to be the minimum integers such that $r_0, r_1 \leq n/2$ and $D(k) = D(k + 2)$ for all $k \in [r_0, n - r_1]$; set $r = \max(r_0, r_1)$ and $f(x \sqcup y) = D(|x \oplus y|)$ by a symmetric XOR function $f$.

Then the randomized public-coin SMP complexity of computing $f$ is $O(r \log r)$.

**5.7 Summary**

We have seen that several related problems, such as testing isomorphism to $k$-parities and some variations of group testing, have complexity $O(k \log k)$ and $\Omega(k)$. We also showed that the query complexity becomes $\Theta(k \log k)$ if non-adaptivity is required.