# ALGORITHMIC AND CODING-THEORETIC METHODS FOR GROUP TESTING AND PRIVATE INFORMATION RETRIEVAL 

A Dissertation by<br>ESMAEIL KARIMI

# Submitted to the Graduate and Professional School of Texas A\&M University in partial fulfillment of the requirements for the degree of DOCTOR OF PHILOSOPHY 

Chair of Committee, Alex Sprintson<br>Committee Members, Krishna Narayanan<br>Srinivas Shakkottai<br>Anxiao Jiang<br>Head of Department, Miroslav M. Begovic

May 2022

Major Subject: Electrical Engineering


#### Abstract

In the first part of this dissertation, we consider the Group Testing (GT) problem and its two variants, the Quantitative GT (QGT) problem and the Coin Weighing (CW) problem. An instance of the GT problem includes a ground set of items that includes a small subset of defective items. The GT procedure consists of a number of tests, such that each test indicates whether or not a given subset of items includes one or more defective items. The goal of the GT procedure is to identify the subset of defective items with the minimum number of tests.

Motivated by practical scenarios where the outcome of the tests can be affected by noise, we focus on the noisy GT setting, in which the outcome of a test can be flipped with some probability. In the noisy GT setting, the goal is to identify the set of defective items with high probability. We investigate the performance of two variants of the Belief Propagation (BP) algorithm for decoding of noisy non-adaptive GT under the combinatorial model for defective items. Through extensive simulations, we show that the proposed algorithms achieve higher success probability and lower false-negative and false-positive rates when compared to the traditional BP algorithm. We also consider a variation of the probabilistic GT model in which the prior probability of each item to be defective is not uniform and in which there is a certain amount of side information on the distribution of the defective items available to the GT algorithm. This dissertation focuses on leveraging the side information for improving the performance of decoding algorithms for noisy GT. First, we propose a probabilistic model, referred to as an interaction model, that captures the side information about the probability distribution of the defective items. Next, we present a decoding scheme, based on BP, that leverages the interaction model to improve


the decoding accuracy. Our results indicate that the proposed algorithm achieves higher success probability and lower false-negative and false-positive rates when compared to the traditional BP, especially in the high noise regime.

In the QGT problem, the result of a test reveals the number of defective items in the tested group. This is in contrast to the standard GT where the result of each test is either 1 or 0 depending on whether the tested group contains any defective items or not. In this dissertation, we study the QGT problem for the combinatorial and probabilistic models of defective items. We propose non-adaptive QGT algorithms using sparse graph codes over bi-regular and irregular bipartite graphs, and binary $t$-error-correcting BCH codes. The proposed schemes provide exact recovery with a probabilistic guarantee, i.e. recover all the defective items with high probability. The proposed schemes outperform existing nonadaptive QGT schemes for the sub-linear regime in terms of the number of tests required to identify all defective items with high probability.

The CW problem lies at the intersection of GT and compressed sensing problems. Given a collection of coins and the total weight of the coins, where the weight of each coin is an unknown integer, the problem is to determine the weight of each coin by weighing subsets of coins on a spring scale. The goal is to minimize the average number of weighings over all possible weight configurations. Toward this goal, we propose and analyze a simple and effective adaptive weighing strategy. This is the first non-trivial achievable upper bound on the minimum expected required number of weighings.

In the second part of this dissertation, we focus on the private information retrieval problem. In many practical settings, the user needs to retrieve information messages from a server in a periodic manner, over multiple rounds of communication. The messages are retrieved one at a time and the identity of future requests is not known to the server. We study the private information retrieval protocols that ensure that the identities of all the messages retrieved from the server are protected. This scenario can occur in practical
settings such as periodic content download from text and multimedia repositories. We refer to this problem of minimizing the rate of data download as online private information retrieval problem. Following the previous line of work by Kadhe et al., we assume that the user knows a subset of messages in the database as side information. The identities of these messages are initially unknown to the server. Focusing on scalar-linear settings, we characterize the per-round capacity, i.e., the maximum achievable download rate at each round. The key idea of our achievability scheme is to combine the data downloaded during the current round and the previous rounds with the original side information messages and use the resulting data as side information for the subsequent rounds.

## DEDICATION

To my dear parents and my beloved wife, for their endless love, support, and encouragement.

## ACKNOWLEDGMENTS

This dissertation would not have been possible without the support and help of many people. First and foremost, I owe my deepest gratitude to my committee chair and advisor, Prof. Alex Sprintson, for mentoring and guiding me, and for the resources and connections he provided for me. I would also like to express my sincere thanks to my committee members, Prof. Krishna Narayanan, Prof. Srinivas Shakkottai, and Prof. Anxiao Jiang, for serving on my dissertation committee and for their time, guidance, and valuable comments and feedback.

I have been fortunate to have had the pleasure of working with many wonderful collaborators. My very special appreciation goes to Prof. Emina Soljanin. I was very lucky for having the opportunity to visit her lab at Rutgers University. Apart from her great hospitality, I benefited the most from her extraordinary mentorship. I also feel lucky to have had the chance to work with Prof. Krishna Narayanan on multilple projects. I am very thankful to him for his insight, his valuable comments, and for all the helpful discussions we had over lengthy meetings. Moreover, I would like to express my sincere gratitude to Anoosheh Heidarzadeh who influenced my PhD more than anyone else. He collaborated with me on multiple projects from the beginning of my PhD and was always ready to help.

I owe a huge amount of thanks to my parents, Fariba and Mohammadvali, and my sisters, Habibeh, Fahimeh, and Hoda. You have always loved me, supported me, and been there for me, and I feel very blessed and fortunate to have you in my life.

My final and special thanks go to my wife, Nadia, for bringing meaning, happiness, and joy to my life. She is the love of my life and the source of all my inspirations and motivations for pursuing my passion.

## CONTRIBUTORS AND FUNDING SOURCES

## Contributors

This work was supported by a dissertation committee consisting of Professor Alex Sprintson (advisor) and Professors Krishna Narayanan and Srinivas Shakkottai of the Department of Electrical and Computer Engineering and Professor Anxiao Jiang of the Department of Computer Science and Engineering at the Texas A\&M University. The work presented in Chapters 2, 3, 4, 5, 6, and 7 is carried out in collaboration with Dr. Anoosheh Heidarzadeh of the Department of Electrical and Computer Engineering at the Texas A\&M University. The work presented in Chapters 3, 4, 5, and 6, is carried out in collaboration with Professor Krishna Narayanan of the Department of Electrical and Computer Engineering at the Texas A\&M University. All other work conducted for the dissertation was completed by the student independently.

## Funding Sources

This work was supported in part by the National Science Foundation (NSF) under Grants No. CIF-1642983 and CIF-1718658.

## NOMENCLATURE

| GT | Group Testing |
| :--- | :--- |
| QGT | Quantitative Group Testing |
| CS | Compressed Sensing |
| CW | Coin Weighing |
| BP | Belief Propagation |
| RSBP | Random Scheduling Belief Propagation |
| NW-RBP | Node-Wise Residual Belief Propagation |
| BPIP | Belief Propagation using Initial Prior probabilities |
| BPUP | Belief Propagation using Updated Prior probabilities |
| BPCG | Belief Propagation on Combined Graph |
| BCH | Bose-Chaudhuri-Hocquenghem |
| MLGT | Multi-Level Group Testing |
| LDPC | Low-Density Parity-Check |
| LLR | Log-Likelihood Ratio |
| FNR | False Negative Rate |
| FPR | False Positive Rate |
| MAP | Maximum a Posteriori |
| $\left\{w_{i}\right\}$ | Weight configuration |
| $w(S)$ | Total weight of the subset $S$ of coins $k$ |
| $\binom{n}{k}$ |  |


| $\Psi$ | Adaptive weighing strategy |
| :---: | :---: |
| $\mathcal{O}(\cdot)$ | Big O notation |
| A | Measurement matrix |
| y | Test results vector |
| $\mathbb{R}$ | Set of real numbers |
| $\mathbb{Z}_{\geq 0}$ | Set of non-negative integers |
| $\mathbb{N}$ | Set of positive integers |
| $\mathbb{F}_{2}$ | Finite field of two elements |
| $\mathbb{F}_{2^{m}}$ | Extension field of $\mathbb{F}_{2}$ |
| $\mathbf{T}_{G}$ | Adjacency matrix of a bipartite graph $G$ |
| U | Signature matrix |
| [i] | Set of integers $\{1, \ldots, i\}$ |
| $L(\cdot)$ | Lef-node degree distribution |
| $\{\mathbf{S}\}_{\ell}$ | Set of all subsets of size $\ell$ for set $S$ |
| PIR | Private Information Retrieval |
| OPIR | Online Private Information Retrieval |
| MDS | Maximum Distance Separable |
| $\mathbb{F}_{q}$ | Finite field for a prime power $q$ |
| $\mathbb{F}_{q^{m}}$ | Extension field of $\mathbb{F}_{q}$ |
| $\mathbb{F}_{q}^{k \times n}$ | $k \times n$-dimensional matrix space over $\mathbb{F}_{q}$ |
| $H(\cdot)$ | (Shannon) Entropy |
| $H(\cdot \mid \cdot)$ | Conditional entropy |
| $\mathbb{P}(\cdot)$ | Probability |
| $\mathbb{P}(\cdot \mid \cdot)$ | Conditional probability |

## TABLE OF CONTENTS

## Page

ABSTRACT ..... ii
DEDICATION ..... v
ACKNOWLEDGMENTS ..... vi
CONTRIBUTORS AND FUNDING SOURCES ..... vii
NOMENCLATURE ..... viii
TABLE OF CONTENTS ..... x
LIST OF FIGURES ..... xiii
LIST OF TABLES ..... xv

1. INTRODUCTION ..... 1
1.1 Group Testing ..... 1
1.1.1 Quantitative Group Testing ..... 3
1.1.1.1 Connection with Compressed Sensing ..... 3
1.1.2 Coin Weighing ..... 4
1.2 Private Information Retrieval ..... 5
1.3 Our Contributions and Organization ..... 6
2. A SIMPLE AND EFFICIENT STRATEGY FOR THE COIN WEIGHING PROB- LEM WITH A SPRING SCALE ..... 10
2.1 Introduction ..... 10
2.1.1 Related Work and Applications ..... 11
2.1.2 Main Contributions ..... 12
2.2 Setup and Notations ..... 12
2.3 Proposed Weighing Strategy ..... 15
2.3.1 Procedure $\Pi_{0}$ ..... 15
2.3.2 Procedure $\Pi_{1}$ ..... 16
2.3.3 Procedure $\Pi_{2}$ ..... 16
2.4 Analysis of the Proposed Startegy ..... 17
2.4.1 Average-Case Setting. ..... 17
2.4.2 Worst-Case Setting ..... 23
2.5 Optimal Nested Weighing Strategy ..... 24
2.5.1 Average-Case Setting ..... 24
2.5.2 Worst-Case Setting ..... 36
2.6 Comparison Results ..... 36
3. NON-ADAPTIVE QUANTITATIVE GROUP TESTING USING BI-REGULAR SPARSE GRAPH CODES ..... 38
3.1 Introduction ..... 38
3.1.1 Related Work and Applications ..... 39
3.1.2 Connection with Compressed Sensing ..... 39
3.1.3 Main Contributions ..... 40
3.2 Problem Setup and Notation ..... 41
3.3 Proposed Algorithm ..... 41
3.3.1 Binary $t$-error-correcting codes and $t$-separable matrices ..... 41
3.3.2 Encoding algorithm ..... 43
3.3.3 Decoding algorithm ..... 46
3.4 Main Results ..... 49
3.5 Proofs of Main Theorems ..... 49
3.5.1 Proof of Theorem 6. ..... 49
3.5.2 Proof of Theorem 7 ..... 53
3.6 Evaluation of $c(t)$ ..... 54
3.7 Comparison Results ..... 59
4. NON-ADAPTIVE QUANTITATIVE GROUP TESTING USING IRREGULAR SPARSE GRAPH CODES ..... 61
4.1 Introduction ..... 61
4.1.1 Related Work and Applications ..... 62
4.1.2 Main Contributions ..... 63
4.2 Problem Setup and Notations ..... 63
4.3 Proposed Algorithm ..... 64
4.3.1 Testing algorithm ..... 64
4.3.2 Recovery Algorithm ..... 67
4.4 Main Results ..... 69
4.5 Proof of Main Theorems ..... 71
4.5.1 Proof of Theorem 8 ..... 71
4.5.2 Proof of Theorem 9 ..... 76
4.6 Comparison Results ..... 76
5. SCHEDULING IMPROVES THE PERFORMANCE OF BELIEF PROPAGA- TION FOR NOISY GROUP TESTING ..... 78
5.1 Introduction ..... 78
5.1.1 Main Contributions ..... 79
5.2 Problem Setup and Notations ..... 80
5.3 Decoding Algorithms ..... 81
5.3.1 Belief Propagation ..... 81
5.3.2 Random Scheduling Belief Propagation ..... 83
5.3.3 Node-wise Residual Belief Propagation ..... 85
5.4 Simulation Results ..... 86
6. NOISY GROUP TESTING WITH SIDE INFORMATION ..... 88
6.1 Introduction ..... 88
6.1.1 Contribution ..... 90
6.2 Probabilistic Model ..... 90
6.3 Proposed Decoding Algorithms ..... 93
6.3.1 Belief Propagation Using Initial Prior Probabilities ..... 93
6.3.2 Belief Propagation Using Updated Prior Probabilities. ..... 98
6.3.3 Belief Propagation on Combined Graphs ..... 98
6.4 Simulation Results ..... 106
7. SINGLE-SERVER SINGLE-MESSAGE ONLINE PRIVATE INFORMATION RETRIEVAL WITH SIDE INFORMATION ..... 110
7.1 Introduction ..... 110
7.1.1 Main Contributions ..... 111
7.1.2 Related Work ..... 112
7.2 Problem Formulation and Main Results ..... 113
7.3 Converse Proof ..... 116
7.4 Achievability Scheme ..... 121
7.5 Example of a protocol execution ..... 126
8. CONCLUSIONS AND FUTURE DIRECTIONS ..... 129
REFERENCES ..... 133

## LIST OF FIGURES

FIGURE Page
2.1 Recursive form of $T_{i, i}$. ..... 18
2.2 Recursive form of $T(n \mid \Delta)$ ..... 23
2.3 The average-case and worst-case results for the proposed strategy, the op- timal nested strategy, and the information-theoretic lower bound. ..... 36
3.1 Tree-like representation of neighborhood of the edge between a left node $v$ and a right node $c$ in the pruned graph. ..... 51
3.2 The number of required tests $(m)$ to identify all $K$ defective items with probability approaching one (for different values of $K$ ) among $N=2^{16}$ items for different values of $t$ obtained via analysis. ..... 58
3.3 The average fraction of unidentified defective items obtained via Monte Carlo simulations for $N=2^{16}$ items among which $K=100$ items are defective. ..... 59
4.1 A tree-like representation of the neighborhood of an edge $e$ between a left node $v$ of degree $i$ and a right node $c$ of degree $r$ in the right-regular bipartite graph. ..... 72
4.2 The number of required tests $(m)$ to identify all defective items (for different values of $K$ ) among $N=2^{32}$ items obtained via analysis. ..... 75
4.3 The probability of error obtained via Monte Carlo simulations for $N=2^{16}$ items among which $K=100$ items are defective. ..... 76
5.1 An example of a factor graph representing a group testing scheme ..... 81
6.1 An example of an interaction graph. Nodes $x_{i}^{(0)}$ and $x_{i}^{(1)}$ represent individual $i$ at times 0 and 1, respectively. An interaction node $I_{i}$ captures interactions between individual $i$ and other individuals from time 0 to time 1. 91
6.2 An example of a factor graph representing a group testing scheme. A variable node $x_{i}^{(1)}$ represents individual $i$ at time 1 . A factor node $y_{t}$ represents test $t$. Factor nodes represented by dotted squares correspond to the a priori probability distribution of the variable nodes.
6.3 An example of a combined graph. Nodes $x_{i}^{(0)}$ and $x_{i}^{(1)}$ represent individual $i$ at times 0 and 1, respectively. An interaction node $I_{i}$ captures interactions between individual $i$ and other individuals from time 0 to time 1 . A factor node $y_{t}$ represents test $t$. Factor nodes represented by dotted squares correspond to the a priori probability distribution of the status of individuals at time 0 .
6.4 Success probability as a function of the number of tests $M$ based on simulation results for $N=500$ individuals, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, and the interaction probability $\theta=0.008$, under the binary symmetric noise model with parameter $\rho \in\{0.01,0.05\}$.. 107
6.5 FNR vs. FPR based on simulation results for threshold $\tau \in[-10,10]$, $N=500$ individuals, $M=350$ tests, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, and the interaction probability $\theta=0.008$, under the binary symmetric noise model with parameter $\rho \in\{0.01,0.05\} . .108$

## LIST OF TABLES

TABLE
3.1 The function $c(t)$ and the optimal left degree $\ell^{\star}$. ..... 49
4.1 The constant $c(t, d)$ for $t=1$ and $d \in\{3,4, \cdots, 18\}$ ..... 70
4.2 The constant $c(t, d)$ for $t=2$ and $d \in\{2,3, \cdots, 17\}$ ..... 70
4.3 The constant $c(t, d)$ for $t=3$ and $d \in\{2,3, \cdots, 17\}$ ..... 71
5.1 The performance of different decoding algorithms for the combinatorial model when $N=100$. ..... 87
6.1 The conditional probability that individual 1 at time 1 is not infected, given the status of individuals 1,3 , and 5 at time 0 . ..... 104

## 1. INTRODUCTION

### 1.1 Group Testing

In the first part of this dissertation, we consider the problem of Group Testing (GT). The invention of GT dates back to World War II when the U.S. military needed to identify soldiers infected with syphilis. In order to identify an infected individual, one can take a blood sample and test it. However, testing the blood samples of a large number of soldiers individually costed lots of time and resources. Also, since the infected population was sparse, this testing procedure seemed inefficient. It is thus natural to ask whether it is possible to test $N$ individuals with less than $N$ tests. Dorfman, in his seminal work [1], introduced the concept of GT for the first time. He showed that by testing pools of blood samples, the required number of tests for identifying infected individuals could be dramatically reduced. In other words, the required number of tests can be reduced when the test is performed as follows. For each test, blood samples are taken from a group of soldiers and a pool is made by mixing the blood samples. Then, the test is performed on the pool. A negative test result indicates that all the soldiers involved in the pool are not infected with syphilis, whereas a positive test result reveals that at least one of the soldiers involved in the pool is infected with syphilis.

The GT problem is defined formally as follows. An instance of the GT problem includes a set S of $N$ items which includes a small subset of defective items. The GT procedure consists of a sequence of tests, such that each test indicates whether there are one or more defective items in a given subset of S . The goal of the GT procedure is to identify the subset of defective items through the minimum number of tests. Aside from the theoretical endeavors, the GT problem has also gained substantial attention from the practical perspective. In particular, the GT problem has been studied for a wide range of
applications from biology and medicine [2] to information and communication technology [3, 4], and computer science [5]. Very recently, group testing has also been used for COVID-19 detection [6-10].

There are two different models for the defective items in the literature: deterministic and randomized. In the deterministic model (a.k.a. the combinatorial model), the exact number of defective items is known, whereas in the randomized model (a.k.a. the probabilistic model), each item is defective with some probability, independent of the other items [11-15]. There are also two types of GT algorithms: non-adaptive, and adaptive. In an adaptive scheme, each test depends on the outcomes of the previous tests. On the other hand, in a non-adaptive scheme, all tests are planned in advance. In other words, the result of one test does not affect the design of another test. A GT algorithm consists of two parts: encoding and decoding. The encoding part is concerned with the test design, i.e., the question of which item should be included in which test. The decoding part is concerned with identifying the defective items given the test design and outcomes of the tests.

Let $D$ be the index set of the defective items and $\hat{D}$ be an estimation of $D$. Depending on the application at hand, there can be different requirements for the closeness of $\hat{D}$ to $D[16,17]$. The strongest condition for closeness is exact recovery when it is required that $\hat{D}=D$. Two weaker conditions are partial recovery without false detections when it is required that $\hat{D} \subseteq D$ and $|\hat{D}| \geq(1-\epsilon)|D|$, and partial recovery without missed detections when it is required that $D \subseteq \hat{D}$ and $|\hat{D}| \leq(1+\epsilon)|D|$. There are also different types of the recovery guarantees [17]. The strongest guarantee is perfect recovery guarantee when the exact or partial recovery needs to be achieved with probability 1 (over the space of all problem instances). A slightly weaker guarantee is probabilistic recovery guarantee when it suffices to achieve the exact or partial recovery with high probability only (and not necessarily with probability 1 ).

Motivated by practical scenarios where the outcome of the tests can be affected by noise, we consider both noiseless and noisy settings. Under the noiseless GT setting, We get a negative test result if all items in the test are not defective, and a positive result if at least one item in the test is defective. However, under the noisy GT setting, the outcome of a test can be flipped with some probability. In the noisy GT setting, the goal is to identify the set of defective items with high probability $(1-\varepsilon)$, for small values of $\varepsilon$.

There is a variant of the GT problem, the quantitative GT problem, and a generalization of the GT problem, the coin weighing problem. We define the quantitative GT problem and the coin weighing problem in Section1.1.1 and Section1.1.2, respectively.

### 1.1.1 Quantitative Group Testing

The Quantitative Group Testing (QGT) problem is to identify the defective items, where the result of a test reveals the number of defective items in the tested group. The key difference between the QGT problem and the standard GT problem is that, unlike the former, in the latter the result of each test is either 1 or 0 depending on whether the tested group contains any defective items or not. The QGT problem has been extensively studied for a wide range of applications, e.g., multi-access communication, spectrum sensing, and network tomography, see, e.g., [13], and references therein.

### 1.1.1.1 Connection with Compressed Sensing

A closely related problem to QGT is the problem of compressed sensing (CS) in which the goal is to recover a sparse signal from a set of (linear) measurements. Given an N dimensional sparse signal with a support size up to $K$, the objective is to identify the indices and the values of non-zero elements of the signal with minimum number of measurements. The main differences between the CS problem and the QGT problem are in the signal model and the constraints on the measurement matrix. Most of the existing works on the CS problem consider real-valued signals and measurement matrices. The QGT
problem, however, requires both the signal and the measurement matrix to be binary.
There are a number of CS algorithms in the literature that use binary measurement matrices, see, e.g. $[18,19]$ and references therein. However, these strategies either use techniques not applicable to binary signals, or provide different types of closeness and guarantee than those required in this work. There are also several CS algorithms for the support recovery where the objective is to determine the indices of the non-zero elements of the signal but not their values [20,21]. The support recovery problem is indeed equivalent to the QGT problem. Notwithstanding, the existing schemes for support recovery rely on non-binary measurement matrices, and hence are not suitable for the QGT problem.

Last but not least, to the best of our knowledge, the majority of works on the CS problem focus mainly on the order optimality of the number of measurements, whereas in this dissertation for the QGT problem we are also interested in minimizing the constant factor hidden in the order.

### 1.1.2 Coin Weighing

The Coin Weighing (CW) problem is defined as follows. Suppose that there is a collection of $n \geq 2$ coins of total weight $d$, where each coin has an unknown integer weight in the set $\{0,1 \ldots, k\}$, for some known integers $d \geq 1$ and $k \geq 1$. The problem is to determine the weight of each coin by weighing subsets of coins in a spring scale while minimizing (i) the maximum number of required weighings over all possible weight configurations (worst-case setting), or (ii) the average number of required weighings over all possible weight configurations (average-case setting).

The CW problem is a generalization of the group testing problem. In particular, for $k=1$ and $d \leq n$, the CW problem is equivalent to the combinatorial quantitative group testing problem, see, e.g., [22]. Consider a set of $n$ items among which a subset of $d$ items are defective. The problem is to identify the defective items by performing fewest
possible tests over subsets of the items, where the result of each test indicates the number of defective items in the tested subset. Also, for $d \ll n$ and $k \geq 1$, the CW problem is equivalent to the integral compressed sensing problem where both the signal and the sensing matrix are integer valued, see, e.g., [23]. Consider an $n$-dimensional signal with at most $d$ components of non-zero integer values and at least $n-d$ components of zero value. The problem is to construct smallest possible number of linear measurements of the signal with integer coefficients, from which the signal can be uniquely recovered.

### 1.2 Private Information Retrieval

In the second part of this dissertation, we focus on the Private Information Retrieval (PIR) problem. The goal of the PIR schemes [24] is to enable a user to download a message or a set of messages belonging to a database whose copies are stored on a single or multiple remote servers, without revealing which message it is requesting. In a single server scenario, the entire database needs to be downloaded to preserve the privacy of the requested message. However, when the user has some side information about the database [25-33], the information-theoretic privacy can be achieved more efficiently than downloading the whole database.

In the PIR with side information setting, the user has access to a random subset of the messages in the database as side information, which are unknown to the server. This side information could have been obtained from other trusted users or through previous interactions with the server. In this setting, the savings in the download cost depend on whether the user wants to protect only the privacy of the requested message, or the privacy of both the requested message and the messages in the side information.

To the best of our knowledge, all of the prior works on PIR focus on retrieval of a single or multiple messages at once. However, in many practical settings, the user needs to retrieve multiple messages periodically, over multiple rounds. For example, a user
might retrieve a book or a movie from an on-line repository on a daily basis. We refer to this setting as online PIR, inspired by the fact that the user does not know the identities of the future items that need to be retrieved from the server. The key requirement in such scenarios is to protect the identity of all the requested messages up to the current round. By leveraging previously downloaded messages, the user can significantly reduce the number of bits that need to be downloaded. Accordingly, we analyze both the fundamental limits as well as the achievability schemes for the online PIR schemes.

### 1.3 Our Contributions and Organization

In this section, we summarize the key contributions of this dissertation, and describe the organization of the chapters.

In Chapter 2, we study a generalized version of the CW problem with a spring scale that lies at the intersection of group testing and compressed sensing problems. We propose and analyze a simple and effective adaptive weighing strategy for $d=k=2$. The results of our theoretical analysis show that the proposed strategy requires $2 \log _{2} n-1$ number of weighings in worst case, and it requires about $1.365 \log n-0.5$ number of weighings on average. (The average-case result is obtained by a numerical evaluation of the exact recursive formulas, derived for the analysis of performance of the proposed strategy.) This is the first non-trivial achievable upper bound on the minimum expected required number of weighings for $d=k=2$. Additionally, for the average-case setting, we design and analyze an optimal strategy within the class of nested strategies, which are mostly being used in today's applications, that requires $\frac{2 n+1}{n+1} \log n-\frac{2(n-1)}{n+1}$ weighings on average. A simple analysis shows that as $n$ grows unbounded, the proposed strategy, when compared to the optimal nested strategy, requires about $31.75 \%$ less number of weighings on average; and when compared to the information-theoretic lower bound, the proposed strategy requires at most about $8.16 \%$ extra number of weighings on average.

In Chapter 3, we investigate the QGT problem for a scenario in which there are $K$ defective items among a population of $N$ items. We propose a non-adaptive QGT strategy for the sub-linear regime where $\frac{K}{N}$ vanishes as $K, N \rightarrow \infty$. We utilize sparse graph codes over bi-regular bipartite graphs and binary $t$-error-correcting BCH codes for the design of the proposed strategy. Leveraging powerful density evolution techniques for the analysis enables us not only to determine the exact value of constants in the number of tests needed but also to provide provable performance guarantees. We show that the proposed scheme provides exact recovery with probabilistic guarantee, i.e. recovers all the defective items with high probability. In particular, for the sub-linear regime, the proposed scheme requires at most $m \approx 1.19 \mathrm{~K} \log _{2}\left(4.74 \frac{N}{K}\right)$ tests to recover all the defective items with probability approaching one as $K, N \rightarrow \infty$. This bound can be achieved by $t=2$. Moreover, for any $t \leq 4$, the encoding and decoding algorithms of the proposed scheme have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

In Chapter 4, we consider a QGT problem with a probabilistic model for defective items, where in a population of $N$ items, each item is defective with probability $\frac{K}{N}$, independently from the other items. We propose a non-adaptive QGT scheme for the sub-linear regime where $\frac{K}{N}$ vanishes as $K, N \rightarrow \infty$. The encoding algorithm of the proposed scheme relies on sparse graph codes over irregular bipartite graphs with optimized left-degree profiles as well as binary $t$-error-correcting BCH codes. As part of the process of optimizing the left-degree profile of the graph, we take advantage of the density-evolution technique to analyze the probability of error of the proposed peeling-based recovery algorithm, i.e., the probability that a defective item remains unidentified over the iterations of the recovery algorithm. We provide provable guarantees on the performance of the proposed scheme in terms of the required number of tests. In particular, we show that in the sub-linear regime the proposed scheme requires $m=c(t, d) K\left(t \log \left(\frac{\ell N}{c(t, d) K}+1\right)+1\right)$ tests to identify all defective items with high probability, where $d$ and $\ell$ are the maximum and average
left degree, respectively, and $c(t, d)$ is constant with respect to $K$ and $N$, and depends only on $t$ and $d$. Moreover, we show that, for any $t \leq 4$, the testing and recovery algorithms of the proposed scheme have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

In Chapter 5, we focus on the noisy GT problem which is concerned with recovering all defective items in a given population of items. We consider a practical regime in which the number of items is in the order of hundreds, and investigate the performance of two variants of Belief Propagation (BP) algorithm for decoding of noisy non-adaptive GT under the combinatorial model for defective items. Through extensive simulations, we show that the proposed algorithms achieve higher success probability and lower false-negative and false-positive rates when compared to the traditional BP algorithm.

Motivated by practical scenarios, such as testing for viral diseases, we study a GT with side information problem in Chapter 6. We focus on the following GT settings: (i) the GT procedure is noisy, i.e., the outcome of the GT procedure can be flipped with a certain probability; (ii) there is a certain amount of side information on the distribution of the infected individuals available to the GT algorithm. First, we propose a probabilistic model, referred to as an interaction model, that captures the side information about the probability distribution of the infected individuals. Our model is motivated by the availability of contact tracing information which can be collected from surveys and mobile phone applications. Next, we present a decoding scheme, based on belief propagation, that leverages the interaction model to improve the decoding accuracy. Our results indicate that the proposed algorithm achieves higher success probability and lower false-negative and false-positive rates when compared to the traditional belief propagation especially in the high noise regime.

In Chapter 7, we turn to the PIR problem. We study the problem of single-server online PIR with side information. In this problem, there is a user who wishes to download
a sequence of messages $\mathcal{X}_{W}=\left\{X_{W_{1}}, X_{W_{2}}, \ldots, X_{W_{t}}\right\}$ from a database $\mathcal{X}$ of $K$ messages, stored on a single server. The communication is performed in rounds, such that at round $i$, the user wishes to retrieve a message $X_{W_{i}}$ for some $W_{i} \in[K]$. We assume that the user decides on which message $W_{i}$ to request at round $i$ at the beginning of that round and that the identity of the future messages $W_{j}, j>i$ are not known at that time. We also assume that at the beginning of the first round the user has access to $M$ messages which are selected uniformly at random from the database. The identity of these $M$ messages are not known to the server.

We focus on the scenario where at round $i$, the user wishes to protect the identity of all the requested messages individually up to round $i,\left\{W_{1}, \ldots, W_{i}\right\}$ for $1 \leq i \leq t$. That is, after the user makes a request to the server at round $i$, the server cannot decide which of the $K$ messages is more likely to get requested at that round and at the previous rounds. Focusing on scalar-linear settings, we characterize the per-round capacity, i.e., the maximum achievable download rate at each round. We also present a scalar-linear coding scheme that achieves this capacity. The key idea of our scheme is to combine the data downloaded during the current round and the previous rounds, with the original side information (unknown to server) so as to construct new side information for the subsequent rounds. We show that for the setting with $K$ messages stored at the server and a random subset of $M$ messages available to the user at the first round, the per-round capacity of the scalar-linear scheme is $C_{1}=(M+1) / K$ for the first round and $C_{i}=\left(2^{i-1}(M+1)\right) / K M$ for round $i \geq 2$, provided that $K /(M+1)=2^{l}$ for some $l \geq 1$.

We conclude this dissertation in Chapter 8 by summarizing our results and presenting some interesting future directions.

## 2. A SIMPLE AND EFFICIENT STRATEGY FOR THE COIN WEIGHING PROBLEM WITH A SPRING SCALE*

### 2.1 Introduction

In this chapter, we consider a generalized version of the coin weighing (CW) problem with a spring scale [34]. Suppose that there is a collection of $n \geq 2$ coins of total weight $d$, where each coin has an unknown integer weight in the set $\{0,1 \ldots, k\}$, for some known integers $d \geq 1$ and $k \geq 1$. The goal is to determine the weight of each coin by weighing subsets of coins in a spring scale. The problem is to devise an adaptive weighing strategy, where each weighing can depend on the results of the previous weighings, that minimizes (i) the maximum number of required weighings over all possible weight configurations (worst-case setting), or (ii) the average number of required weighings over all possible weight configurations (average-case setting).

The CW problem is a generalization of the group testing problem. In particular, for $k=1$ and $d \leq n$, the CW problem is equivalent to the combinatorial quantitative group testing problem, see, e.g., [22]. Also, for $d \ll n$ and $k \geq 1$, the CW problem is equivalent to the integral compressed sensing problem where both the signal and the sensing matrix are integer valued, see, e.g., [23].

For $d=k=1$, a simple adaptive bisecting weighing strategy is optimal in both worstcase and average-case settings [13]. However, the simplest non-trivial case of the problem, i.e., $d=k=2$, is still open, and hence the focus of this work. For the worst-case setting, a simple information-theoretic argument yields a lower bound on the minimum required number of weighings by $\max \left\{\log _{2} n, \log _{3}\binom{n}{2}\right\}$ (see Theorem 1); and for the average-

[^0]case setting, a similar argument gives a lower bound of $\frac{2}{n+1} \log _{2} n+\frac{n-1}{n+1} \log _{3}\binom{n}{2}$ on the minimum expected required number of weighings (see Theorem 1). Notwithstanding, the question whether these lower bounds are achievable remains open. For the worst-case setting, $2 \log _{2} n-1$ weighings are known to be sufficient, and this bound is achievable by a simple nested strategy (see [13, Lemma 1]). This quantity also serves as an upper bound for the average-case setting, and no tighter achievable upper bound was previously reported.

### 2.1.1 Related Work and Applications

The worst-case setting of the CW problem was originally proposed in [35] for $k=1$ and unknown $d$, and was later studied for $k=1$ and known $d$, e.g., in [36-38]. Various order-optimal strategies were previously proposed for unknown $d$, see, e.g., [36, 39], and for known $d$, see, e.g., [34,40-42]. Recently, in [34], Bshouty proposed the first and only known order-optimal strategy for any $k>1$ and unknown $d$, and no such result exists for any $k>1$ and known $d$. Despite the rich literature on the worst-case setting, there was no result for the average-case setting of the CW problem prior to the present work, excluding the results that trivially carry over from worst case into average case.

The worst-case setting of the CW problem has also been extensively studied for a wide range of applications, e.g., multi-access communication, spectrum sensing, traffic monitoring, anomaly detection, and network tomography, to name a few (see, e.g., [13], and references therein). Moreover, most of these applications are being run repeatedly over time, and for such applications, the average-case performance is expected to be more relevant than the worst-case performance. This observation is the primary motivation for studying the average-case setting of the CW problem in this work.

### 2.1.2 Main Contributions

In this work, we propose and analyze a simple and effective adaptive weighing strategy for the case $d=k=2$. The results of our theoretical analysis show that the proposed strategy requires $2 \log _{2} n-1$ number of weighings in worst case, and it requires about $1.365 \log n-0.5$ number of weighings on average. (The average-case result is obtained by a numerical evaluation of the exact recursive formulas, derived for the analysis of performance of the proposed strategy.) This is the first non-trivial achievable upper bound on the minimum expected required number of weighings for $d=k=2$. Additionally, for the average-case setting, we design and analyze an optimal strategy within the class of nested strategies, which are mostly being used in today's applications, that requires $\frac{2 n+1}{n+1} \log n-\frac{2(n-1)}{n+1}$ weighings on average. A simple analysis shows that as $n$ grows unbounded, the proposed strategy, when compared to the optimal nested strategy, requires about $31.75 \%$ less number of weighings on average; and when compared to the information-theoretic lower bound, the proposed strategy requires at most about $8.16 \%$ extra number of weighings on average.

### 2.2 Setup and Notations

Fix an integer $l \geq 1$, and let $n=2^{l}$. Let $N=\{1, \ldots, n\}$. Consider a collection $N$ of $n$ coins, each coin $i \in N$ of an unknown integer weight $w_{i} \in\{0,1,2\}$. We refer to the set $\left\{w_{1}, \ldots, w_{n}\right\}$, simply denoted by $\left\{w_{i}\right\}$, as the weight configuration, or the configuration, for short. For any $S \subseteq N$, denote by $w(S)$ the total weight of the subset $S$ of coins, i.e., $w(S)=\sum_{i \in S} w_{i}$. We assume that the total weight of $N$, i.e., $w(N)$, is equal to 2 .

The problem is to determine the weight of all coins in $N$ by weighing subsets of $N$ in a spring scale. In the worst-case setting of the problem, the goal is to minimize the maximum number of required weighings over all possible configurations; and in the average-case setting of the problem, the goal is to minimize the expected number of required weigh-
ings over all possible configurations, where all possible configurations are assumed to be equally probable.

Since $w(N)=2$ and $w_{i} \in\{0,1,2\}$ for all $i \in N$, there are $n$ distinct configurations such that $w_{i}=2$ for some $i \in N$, and $w_{j}=0$ for all $j \in N \backslash\{i\}$, and there are $\binom{n}{2}$ distinct configurations such that $w_{i}=w_{j}=1$ for some $i, j \in N$ and $w_{k}=0$ for all $k \in N \backslash\{i, j\}$. We refer to the first group of configurations as Type-I, and refer to the second group as Type-II. For example, for $n=2$, the possible configurations $\left\{w_{1}, w_{2}\right\}$ are $\{2,0\},\{0,2\}$, and $\{1,1\}$, where the first two configurations are Type-I and the third one is Type-II. For the ease of exposition, we define a representative function $\Delta\left(\left\{w_{i}\right\}_{i \in S}\right)$ for any $S \subseteq N, w(S)=2$, as follows. For any Type-I (sub-) configuration $\left\{w_{i}\right\}_{i \in S}$, $\Delta\left(\left\{w_{i}\right\}_{i \in S}\right)=0$, and for any Type-II (sub-) configuration $\left\{w_{i}\right\}_{i \in S}, \Delta\left(\left\{w_{i}\right\}_{i \in S}\right)=|i-j|$, where $w_{i}=w_{j}=1$.

Any adaptive weighing strategy $\Psi$ can be defined as a sequence $\left\{S_{1}, S_{2}, \ldots\right\}$ of subsets of coins that are to be weighed following the prescribed order, where the choice of each subset $S_{i}$ can depend on $\left\{S_{j}\right\}_{j=1}^{i-1}$ and $\left\{w\left(S_{j}\right)\right\}_{j=1}^{i-1}$.

Consider an arbitrary strategy $\Psi$. Denote by $T_{\text {ave }}^{\Psi}(n)$ the expected number of weighings required by the strategy $\Psi$ to determine the weight of all coins in $N$, over all possible weight configurations. For any subset $S$ of coins, all with unknown weights, we denote by $T_{w}^{\Psi}(s)$ the expected number of weighings that the strategy $\Psi$ performs to determine the weight of all coins in $S$, where $s=|S|$ and $w=w(S)$. The expectation is taken over all possible (sub-) configurations $\left\{\tilde{w}_{i}\right\}_{i \in S}, \tilde{w}_{i} \in\{0,1,2\}$, such that $\sum_{i \in S} \tilde{w}_{i}=w$.

For any subset $S$ of coins, all with unknown weights, such that $w(S)=2$, denote by $T^{\Psi}(s \mid \Delta)$ the expected number of weighings that the strategy $\Psi$ performs to determine the weight of all coins in $S$, given that $\Delta\left(\left\{w_{i}\right\}_{i \in S}\right)=\Delta$, where $s=|S|$. Here, the expectation is taken over all possible (sub-) configurations $\left\{\tilde{w}_{i}\right\}_{i \in S}, \tilde{w}_{i} \in\{0,1,2\}$, such that $\sum_{i \in S} \tilde{w}_{i}=2$ and $\Delta\left(\left\{\tilde{w}_{i}\right\}_{i \in S}\right)=\Delta$.

For any disjoint subsets $A$ and $B$ of coins, all with unknown weights, such that $w(A)=$ 1 and $w(B)=1$, denote by $T^{\Psi}(a, b)$ the expected number of weighings required by the strategy $\Psi$ to determine the weight of all coins in $A$ and $B$, where $a=|A|$ and $b=|B|$. The expectation is here taken over all possible (sub-) configurations $\left\{\tilde{w}_{i}\right\}_{i \in A}$ and $\left\{\tilde{w}_{i}\right\}_{i \in B}$, $\tilde{w}_{i} \in\{0,1\}$, such that $\sum_{i \in A} \tilde{w}_{i}=1$ and $\sum_{i \in B} \tilde{w}_{i}=1$. For convenience, we adopt the convention $T^{\Psi}(1, s)=T^{\Psi}(s, 1)=T_{1}^{\Psi}(s)$.

From now on, whenever the strategy $\Psi$ is clear from the context, we omit the superscript $\Psi$, and denote $T_{\text {ave }}^{\Psi}(n), T_{w}^{\Psi}(s), T^{\Psi}(s \mid \Delta)$, and $T^{\Psi}(a, b)$ by $T_{\text {ave }}(n), T_{w}(s), T(s \mid \Delta)$, and $T(a, b)$, respectively. Moreover, we define $T_{\max }(n), T_{w}^{\star}(s), T^{\star}(s \mid \Delta)$, and $T^{\star}(a, b)$ similarly as $T_{\text {ave }}(n), T_{w}(s), T(s \mid \Delta)$, and $T(a, b)$, respectively, except for the maximum number of weighings, instead of the expected number of weighings, that the strategy $\Psi$ must perform.

Theorem 1. For any weighing strategy $\Psi$, we have

$$
T_{\max }^{\Psi}(n) \geq \max \left\{\log _{2} n, \log _{3}\binom{n}{2}\right\}
$$

and

$$
T_{\mathrm{ave}}^{\Psi}(n) \geq \frac{2}{n+1} \log _{2} n+\frac{n-1}{n+1} \log _{3}\binom{n}{2}
$$

Proof. Recall that there are two types of weight configurations: Type-I and Type-II. For any Type-I configuration, the result of weighing on any subset of coins is either zero or non-zero, and the number of distinct possible configurations of Type-I is $n$. Thus, at least $\log _{2} n$ weighings are needed to distinguish a particular configuration of this type. For any Type-II configuration, the result of weighing on any subset of coins can be 0 , or 1 , or 2 . Thus, there are $\binom{n}{2}$ distinct possible configurations of this type, and to distinguish a particular configuration within this class, one needs at least $\log _{3}\binom{n}{2}$ weighings. Accordingly, for
a configuration of an unknown type, at least $\max \left\{\log _{2} n, \log _{3}\binom{n}{2}\right\}$ weighings are required to identify the configuration. Since all configurations are equally probable, it can be easily verified that a randomly chosen configuration is of Type-I or of Type-II with probability $\frac{2}{n+1}$ or $\frac{n-1}{n+1}$, respectively. Consequently, on average, at least $\frac{2}{n+1} \log _{2} n+\frac{n-1}{n+1} \log _{3}\binom{n}{2}$ weighings are necessary to identify a particular configuration of an unknown type.

### 2.3 Proposed Weighing Strategy

In this section, we propose a weighing strategy that determines the weight of all coins, for the setup in Section 2.2.

For any set $S=\left\{i_{1}, \ldots, i_{|S|}\right\}$ such that $|S|$ is a power of 2 , we denote by $S_{1}$ and $S_{2}$ the two disjoint subsets $\left\{i_{1}, \ldots, i_{|S| / 2}\right\}$ and $\left\{i_{|S| / 2+1}, \ldots, i_{|S|}\right\}$, respectively.

The proposed strategy is based on three recursive procedures $\Pi_{0}, \Pi_{1}$, and $\Pi_{2}$, described shortly. At the beginning, the strategy initializes the weight of all coins by zero, i.e., $\hat{w}_{i}=0$ for all $i \in N$. Then, it starts with the procedure $\Pi_{0}$ over the set $N$. The weights of coins will be updated recursively according to the procedures $\Pi_{0}, \Pi_{1}$, and $\Pi_{2}$. This process is terminated once the sum of weights of all coins, $\sum_{i \in N} \hat{w}_{i}$, is equal to 2 , and the strategy returns $\left\{\hat{w}_{i}\right\}_{i \in N}$.

The inputs of the procedure $\Pi_{0}$ are a set $S$ and its weight $w(S)$. The procedure $\Pi_{1}$ takes as input two disjoint sets $A$ and $B$ such that $w(A)=w(B)=1$, and the procedure $\Pi_{2}$ takes as input two disjoint sets $A$ and $B$ such that $w(A)=w(B)=w\left(A_{1} \cup B_{1}\right)=1$. (Recall that $A_{1}=\left\{i_{1}, \ldots, i_{|A| / 2}\right\}$ and $B_{1}=\left\{j_{1}, \ldots, j_{|B| / 2}\right\}$ when $A=\left\{i_{1}, \ldots, i_{|A|}\right\}$ and $B=\left\{j_{1}, \ldots, j_{|B|}\right\}$.) We represent these procedures by $\Pi_{0}(S), \Pi_{1}(A, B)$, and $\Pi_{2}(A, B)$, respectively.

### 2.3.1 Procedure $\Pi_{0}$

For any $S=\{i\}$, the procedure $\Pi_{0}(S)$ updates $\hat{w}_{i}$ by $w(S)$; and for any $S,|S|>1$, the procedure $\Pi_{0}(S)$ begins with weighing $S_{1}$. If $w\left(S_{1}\right)=0$ or $w\left(S_{1}\right)=2$, the procedure
$\Pi_{0}(S)$ continues with $\Pi_{0}\left(S_{2}\right)$ or $\Pi_{0}\left(S_{1}\right)$, respectively. Otherwise, depending on $w(S)=1$ or $w(S)=2$, the procedure $\Pi_{0}(S)$ continues with $\Pi_{0}\left(S_{1}\right)$ or $\Pi_{1}\left(S_{1}, S_{2}\right)$, respectively. We note that for $w\left(S_{1}\right)=0$ or $w\left(S_{1}\right)=2$, the procedure $\Pi_{0}$ follows a simple bisecting strategy, and for $w\left(S_{1}\right)=1$, it follows a generalized bisecting strategy defined below.

### 2.3.2 Procedure $\Pi_{1}$

For any $A=\{i\}$ and $B=\{j\}$, the procedure $\Pi_{1}(A, B)$ updates $\hat{w}_{i}$ and $\hat{w}_{j}$ by 1 ; For any $A$ and $B$ such that $|A|=1$ and $|B|>1$ or $|A|>1$ and $|B|=1$, the procedure $\Pi_{1}(A, B)$ continues with two procedures $\Pi_{0}(A)$ and $\Pi_{0}(B)$. For any $A$ and $B$ such that $|A|>1$ and $|B|>1$, the procedure $\Pi_{1}(A, B)$ weighs $A_{1} \cup B_{1}$. If $w\left(A_{1} \cup B_{1}\right)=0$ or $w\left(A_{1} \cup B_{1}\right)=2$, the procedure $\Pi_{1}(A, B)$ continues with $\Pi_{1}\left(A_{2}, B_{2}\right)$ or $\Pi_{1}\left(A_{1}, B_{1}\right)$, respectively; otherwise, it continues with $\Pi_{2}(A, B)$.

### 2.3.3 Procedure $\Pi_{2}$

For any $A=\left\{i_{1}, i_{2}\right\}$ and $B=\left\{j_{1}, j_{2}\right\}$, the procedure $\Pi_{2}(A, B)$ weighs $A_{1}=\left\{i_{1}\right\}$, and updates $\hat{w}_{i_{1}}, \hat{w}_{i_{2}}, \hat{w}_{j_{1}}$, and $\hat{w}_{j_{2}}$ by $w\left(A_{1}\right), 1-w\left(A_{1}\right), 1-w\left(A_{1}\right)$, and $w\left(A_{1}\right)$, respectively. For any $A$ and $B$ such that $\max (|A|,|B|)>2$ and $|A| \leq|B|$, the procedure $\Pi_{2}(A, B)$ weighs $A_{1} \cup\left(B_{2}\right)_{1}$. (Recall that $\left(B_{2}\right)_{1}=\left\{j_{|B| / 2+1}, \ldots, j_{3|B| / 4}\right\}$ when $B_{2}=\left\{j_{|B| / 2+1}, \ldots, j_{|B|}\right\}$.) If $w\left(A_{1} \cup\left(B_{2}\right)_{1}\right)$ is equal to 0,1 , or 2 , the procedure $\Pi_{2}(A, B)$ continues with $\Pi_{1}\left(A_{2}, B_{1}\right), \Pi_{1}\left(A_{1},\left(B_{2}\right)_{2}\right)$, or $\Pi_{1}\left(A_{1},\left(B_{2}\right)_{1}\right)$, respectively. For any $A$ and $B$ such that $\max (|A|,|B|)>2$ and $|B|<|A|$, the procedure is the same, except for $A$ and $B$ being interchanged.

Example 1. Consider $n=8$ coins of weights $w_{3}=w_{6}=1$ and $w_{i}=0$ for all $i \notin\{3,6\}$. Let $N=\{1, \ldots, 8\}$. Initialize $\hat{w}_{i}$ by 0 for all $i \in N$. Applying $\Pi_{0}(N)$, the set $\{1, \ldots, 4\}$ is weighed. Since $w(\{1, \ldots, 4\})=1$, the strategy proceeds with $\Pi_{1}(\{1,2,3,4\},\{5,6,7,8\})$. According to the strategy, the set $\{1,2\} \cup\{5,6\}$ is weighed. Since $w(\{1,2\} \cup\{5,6\})=1$ the strategy continues with $\Pi_{2}(\{1,2,3,4\},\{5,6,7,8\})$. Ac-
cording to the procedure $\Pi_{2}$, weighing is performed on $\{1,2\} \cup\{7\}$. Since $w(\{1,2\} \cup$ $\{7\})=0$, the strategy proceeds with $\Pi_{1}(\{3,4\},\{5,6\})$, and weighs $\{3\} \cup\{5\}$. Since $w(\{3\} \cup\{5\})=1$, the strategy continues with $\Pi_{2}(\{3,4\},\{5,6\})$. According to the procedure $\Pi_{2}$, the weighing is performed on $\{3\}$. Since $w(\{3\})=1$, the strategy updates $\hat{w}_{3}=1, \hat{w}_{4}=0, \hat{w}_{5}=0$, and $\hat{w}_{6}=1$. Since $\sum_{i \in N} \hat{w}_{i}=2$, the process is terminated.

### 2.4 Analysis of the Proposed Startegy

In this section, we analyze the average-case and worst-case performance of the strategy proposed in Section 2.3. For simplifying the notation, for all $0 \leq i, j \leq l$, we denote $T\left(2^{i}, 2^{j}\right)$ and $T^{\star}\left(2^{i}, 2^{j}\right)$ by $T_{i, j}$ and $T_{i, j}^{\star}$, respectively.

### 2.4.1 Average-Case Setting

The following two lemmas are useful for computing $T_{i, j}$ recursively for different values of $i$ and $j$.

Lemma 1. $T_{0,0}=0, T_{1,1}=\frac{3}{2}$, and for all $1<i<l$,

$$
T_{i, i}=\frac{3}{4} T_{i-1, i-1}+\frac{1}{4} T_{i-2, i-1}+\frac{3}{2} .
$$

Proof. It is easy to see that $T_{0,0}=T(1,1)=0$, since we have two coins and the weight of each coin is 1 , so no weighing is required. To obtain $T_{1,1}=T(2,2)$, we know that we have four coins and the total weight of two coins (set $A$ ) is 1 and the total weight of the other two (set $B$ ) is also 1 . Thus, one coin in $A$ and one coin in $B$ are weighed together. The weighing outcome is either (i) 0 or 2 with probability $\frac{1}{2}$, or (ii) 1 with probability $\frac{1}{2}$. In the case (i), with just one weighing the weights of all coins are determined. In the case (ii), one more weighing is needed to be performed on one coin in $A$ or one coin in $B$, in order to find the weights of all coins. Thus, $T_{1,1}=\frac{1}{2}(1)+\frac{1}{2}(2)=\frac{3}{2}$.

For any $1<i<l, T_{i, i}$ can be computed based on a similar reasoning as follows.


Figure 2.1: Recursive form of $T_{i, i}$.

By performing one weighing on the union set of half of $A$ (say $A_{1}$ ) and half of $B$ (say $B_{1}$ ), with probability $\frac{1}{2}$ the weighing outcome is 0 or 2 , and the expected number of extra required weighings is $T_{i-1, i-1}$. Otherwise, with probability $\frac{1}{2}$, the weighing outcome is 1 . In this case, one more weighing is needed to be performed on the union set of $A_{1}$ and half of $B_{2}$ (say $\left(B_{2}\right)_{1}$ ). Followed by two weighings, with probability $\frac{1}{2}$, the expected number of extra required weighings is equal to $T_{i-1, i-1}$; and with probability $\frac{1}{2}$, this quantity is equal to $T_{i-2, i-1}$ (see Fig. 2.1). Thus, we have

$$
T_{i, i}=\frac{1}{2}\left(T_{i-1, i-1}+1\right)+\frac{1}{4}\left(T_{i-1, i-1}+2\right)+\frac{1}{4}\left(T_{i-2, i-1}+2\right),
$$

or equivalently,

$$
T_{i, i}=\frac{3}{4} T_{i-1, i-1}+\frac{1}{4} T_{i-2, i-1}+\frac{3}{2} .
$$

This completes the proof.

Lemma 2. For all $1 \leq j<l, T_{0, j}=j$; for all $1<j<l, T_{1, j}=j+\frac{1}{4}$; and for all $1<i \leq l-1$ and $1-i<j \leq l-i$,

$$
T_{i, i+j}=\frac{3}{4} T_{i-1, i+j-1}+\frac{1}{8} T_{i-2, i+j-1}+\frac{1}{8} T_{i+j-2, i+j-2}+\frac{3}{2} .
$$

Proof. For any $T_{i, j}$, we consider two disjoint sets $A$ and $B$ of size $2^{i}$ and $2^{j}$, respectively, each set of total weight 1 . By the definition, $T_{0, j}=T\left(1,2^{j}\right)=T_{1}\left(2^{j}\right)$. In this case, the
proposed strategy applies the procedure $\Pi_{0}(B)$, which requires $j$ weighings, on average, to determine the weights of all coins. Thus, $T_{0, j}=j$.

Now, consider $T_{1, j}=T\left(2,2^{j}\right)$. In this case, with one weighing (on the set $A_{1} \cup B_{1}$ ) with probability $\frac{1}{2}$, the outcome is 0 or 2 . For the outcome 0 (or 2 ), we find that one coin with weight 1 is in $A_{2}$ ( or $A_{1}$ ) and the other coin of weight 1 is in $B_{2}$ (or $B_{1}$ ). Thus, $T\left(1,2^{j-1}\right)=j-1$ more weighings, on average, are needed to determine the weights of all coins. Also, with probability $\frac{1}{2}$, the weighing outcome is 1 and the weight of no coin is discovered by this particular weighing. In this case, with one more weighing (on the set $\left.A_{1} \cup\left(B_{2}\right)_{1}\right)$, with probability $\frac{1}{2}$ the weighing outcome is 0 , and $T\left(1,2^{j-1}\right)=j-1$ more weighings, on average, are needed to determine the weights of all coins. Otherwise, with probability $\frac{1}{2}$, the weighing outcome is 1 or 2 , and thus, $T\left(1,2^{j-2}\right)=j-2$ more weighings are needed on average to find the weights of all coins. As a result, we have

$$
T_{1, j}=\frac{1}{2}(j-1+1)+\frac{1}{4}(j-1+2)+\frac{1}{4}(j-2+2)=j+\frac{1}{4} .
$$

Lastly, consider $T_{i, i+j}=T\left(2^{i}, 2^{i+j}\right)$. Performing one weighing (on the set $A_{1} \cup B_{1}$ ), there are two cases: (i) the outcome is 0 or 2 (with probability $\frac{1}{2}$ ), and (ii) the outcome is 1 (with probability $\frac{1}{2}$ ). First, consider the case (i). By a similar argument as before, for the outcome 0 (or 2), we find the weight of all coins in $A_{1}$ (or $A_{2}$ ) and that of all coins in $B_{1}$ (or $B_{2}$ ), respectively. Thus, $T\left(2^{i-1}, 2^{i+j-1}\right)$ more weighings are needed, on average, to determine the weights of all coins. Next, consider the case (ii). There are two subcases: (ii-1) with probability $\frac{1}{4}$, the coin with weight 1 belongs to the larger set, and (ii-2) with probability $\frac{1}{4}$, the coin with weight 1 belongs to the smaller set. In the case (ii-1), the weight of no coin can be determined by this particular weighing. Thus, with one more weighing (on the set $\left(A_{2}\right)_{1} \cup B_{1}$ or $\left.A_{1} \cup\left(B_{2}\right)_{1}\right)$, with probability $\frac{1}{2}$ the weighing outcome is 0 , and consequently, $T\left(2^{i-1}, 2^{i+j-1}\right)$ more weighings are needed, on average, to determine
the weights of all coins; otherwise, with probability $\frac{1}{2}$, the weighing outcome is 1 or 2 , and $T\left(2^{i-2}, 2^{i+j-1}\right)$ more weighings are needed, on average, to find the weights of all coins. In the case (ii-2), again, the weight of no coin is determined by this specific weighing. Thus, with one more weighing (on set $\left(A_{2}\right)_{1} \cup B_{1}$ or $\left.A_{1} \cup\left(B_{2}\right)_{1}\right)$, with probability $\frac{1}{2}$, the weighing outcome is 0 , and $T\left(2^{i-1}, 2^{i+j-1}\right)$ more weighings are needed, on average, to determine the weights of all coins. Otherwise, with probability $\frac{1}{2}$, the weighing outcome is 1 or 2 , and $T\left(2^{i+j-2}, 2^{i+j-2}\right)$ more weighings, on average, are needed to find the weights of all coins. Thus, we have

$$
\begin{aligned}
T_{i, i+j} & =\frac{1}{2}\left(T_{i-1, i+j-1}+1\right)+\frac{1}{8}\left(T_{i-1, i+j-1}+2\right)+\frac{1}{8}\left(T_{i-2, i+j-1}+2\right) \\
& +\frac{1}{8}\left(T_{i-1, i+j-1}+2\right)+\frac{1}{8}\left(T_{i+j-2, i+j-2}+2\right),
\end{aligned}
$$

or in turn,

$$
T_{i, i+j}=\frac{3}{4} T_{i-1, i+j-1}+\frac{1}{8} T_{i-2, i+j-1}+\frac{1}{8} T_{i+j-2, i+j-2}+\frac{3}{2} .
$$

This completes the proof.
For any $0 \leq \Delta \leq n-1$, define $\Delta_{n}=\frac{n}{2}-\left|\Delta-\frac{n}{2}\right|$. For simplifying the notation, let

$$
q_{\Delta, i} \triangleq \begin{cases}\frac{2^{i-1} \Delta_{n}}{2^{l-1}-2^{i-1} \Delta_{n}}, & \Delta_{n}<2^{l-(i+1)}, 2^{l-(i+1)} \geq 1 \\ 1, & \Delta_{n} \geq 2^{l-(i+1)}, 2^{l-(i+1)} \geq 1 \\ 0, & \text { otherwise }\end{cases}
$$

for all $1 \leq i<l$, and

$$
q_{\Delta, 0} \triangleq \begin{cases}\frac{\Delta_{n}}{2^{l-1}}, & \Delta_{n}<2^{l-1} \\ 1, & \text { otherwise }\end{cases}
$$

for $i=0$. Also, let $q_{\Delta, l} \triangleq 1$. Moreover, let

$$
p_{\Delta, j} \triangleq \begin{cases}\frac{2^{l-1}-2^{j} \Delta_{n}}{2^{l-1}-2^{j-1} \Delta_{n}}, & \Delta_{n}<2^{l-1}, 2^{l-1} \geq 1 \\ 0, & \text { otherwise }\end{cases}
$$

for all $1 \leq j<l$, and

$$
p_{\Delta, 0} \triangleq \begin{cases}\frac{2^{l-1}-\Delta_{n}}{2^{l-1}} & \Delta_{n}<2^{l-1}, 2^{l-1} \geq 1 \\ 0 & \text { otherwise }\end{cases}
$$

The following lemma is useful for computing $T(n \mid \Delta)$ based on the values of $T_{i, j}$.
Lemma 3. For any $0 \leq \Delta \leq n-1$, we have

$$
T(n \mid \Delta)=\sum_{i=0}^{l-1} m_{\Delta, i}\left(T_{l-i-1, l-i-1}+i+1\right)+m_{\Delta, l} l
$$

where $m_{\Delta, 0}=q_{\Delta, 0}$, and

$$
m_{\Delta, i}=q_{\Delta, i} \prod_{j=0}^{i-1} p_{\Delta, j}
$$

for all $1 \leq i \leq l$.
Proof. Fix an arbitrary $0 \leq \Delta \leq n-1$. Consider the application of the proposed strategy on an arbitrary configuration $\left\{w_{i}\right\}$ such that $\Delta\left(\left\{w_{i}\right\}\right)=\Delta$. Let $p_{\Delta, 0}$ (or respectively, $\left.q_{\Delta, 0}\right)$ be the probability that the outcome of the first weighing is 0 or 2 (or respectively, 1 ). Thus, with probability $p_{\Delta, 0}$, followed by performing one weighing, the expected number of extra required weighings is $T\left(\left.\frac{n}{2} \right\rvert\, \Delta\right)$. Similarly, with probability $p_{\Delta, 0} p_{\Delta, 1}$, after performing two weighings, $T\left(\left.\frac{n}{4} \right\rvert\, \Delta\right)$ more weighings are needed on average, and so forth (see the straight line in Fig. 2.2). Thus, with probability $\prod_{i=0}^{l-1} p_{\Delta, i}, l$ weighings are needed. On the other hand, with probability $q_{\Delta, 0}$, followed by performing one weighing, $T_{l-1, l-1}$ more
weighings are needed on average. Similarly, with probability $p_{\Delta, 0} q_{\Delta, 1}$, after performing two weighings, $T_{l-2, l-2}$ extra number of weighings are needed on average, and so forth (see the diagonal lines in Fig. 2.2). Putting everything together, we can write

$$
T(n \mid \Delta)=q_{\Delta, 0}\left(T_{l-1, l-1}+1\right)+p_{\Delta, 0} q_{\Delta, 1}\left(T_{l-2, l-2}+2\right)+\cdots+p_{\Delta, 0} p_{\Delta, 1} \cdots p_{\Delta, l-1}(l)
$$

or equivalently,

$$
T(n \mid \Delta)=m_{\Delta, 0}\left(T_{l-1, l-1}+1\right)+m_{\Delta, 1}\left(T_{l-2, l-2}+2\right)+\cdots+m_{\Delta, l-1}\left(T_{0,0}+l\right)+m_{\Delta, l}(l)
$$

This completes the proof.

By combining the results of Lemmas $1-3$, we can compute $T_{\text {ave }}(n)$ for the proposed strategy as follows.

Theorem 2. For the proposed strategy, $T_{\mathrm{ave}}(n)$ can be computed as $T_{\mathrm{ave}}(n)=P M I$, where $P=\left[P_{0}, \ldots, P_{n-1}\right]$ is a row vector of length $n$, where $P_{\Delta}=\frac{2(n-\Delta)}{n(n+1)}$ for all $0 \leq$ $\Delta \leq n-1$; and $I=\left[I_{1}, \ldots, I_{l}, l\right]^{\top}$ is a column vector of length $l+1$, where $I_{i}=T_{l-i, l-i}+i$ for all $1 \leq i \leq l$; and $M=\left(m_{\Delta, i}\right)_{0 \leq \Delta \leq n-1,0 \leq i \leq l}$ is an $n \times(l+1)$ matrix, where $\left\{m_{\Delta, i}\right\}$ are defined in Lemma 3.

Proof. Fix an arbitrary $0 \leq \Delta \leq n-1$. It is easy to verify that there exist $n-\Delta$ distinct configurations $\left\{w_{i}\right\}$ such that $\Delta\left(\left\{w_{i}\right\}\right)=\Delta$. Also, the total number of possible configurations are $n+\binom{n}{2}=\frac{n(n+1)}{2}$. Thus, for a randomly chosen configuration $\left\{w_{i}\right\}$, the probability that $\Delta\left(\left\{w_{i}\right\}\right)=\Delta$ is equal to $P_{\Delta}=\frac{2(n-\Delta)}{n(n+1)}$. Then, it is easy to see that $T_{\text {ave }}(n)=\sum_{\Delta=0}^{n-1} P_{\Delta} T(n \mid \Delta)$. Re-writing this equation in matrix form by using the result of Lemma 3, the result of the theorem follows immediately.


Figure 2.2: Recursive form of $T(n \mid \Delta)$.

### 2.4.2 Worst-Case Setting

Theorem 3. For the proposed strategy, we have $T_{\max }(n)=2 \log _{2} n-1$.

Proof. First, we prove that $T_{\max }(n)=T_{2}^{\star}(n)=T_{2}^{\star}\left(\frac{n}{2}\right)+2$. It is easy to verify that $T^{\star}(n \mid \Delta)$ and $T_{i, i}^{\star}$ can be computed recursively similar to $T(n \mid \Delta)$ and $T_{i, i}$, respectively, as shown in Fig. 2.2 and Fig. 2.1, by replacing $T$ with $T^{\star}$ everywhere. As can be seen in Fig. 2.2, the straight lines correspond to the cases in which one weighing resolves the weights of half of the coins; whereas, the diagonal lines correspond to the cases in which the weight of none of the coins is determined. That is, the diagonal lines correspond to the cases that require more number of weighings. Moreover, from $T^{\star}(n \mid \Delta)$ to $T^{\star}\left(\frac{n}{4}, \frac{n}{4}\right)$, there are two ways (see Fig. 2.1); one way is through $T^{\star}\left(\left.\frac{n}{2} \right\rvert\, \Delta\right)$ which requires two weighings, and the other way is through $T^{\star}\left(\frac{n}{2}, \frac{n}{2}\right)$ which requires, in worst case, three weighings, noting that $T^{\star}\left(\frac{n}{2}, \frac{n}{2}\right)=T^{\star}\left(\frac{n}{4}, \frac{n}{4}\right)+2$. Thus, among the diagonal lines, the first one, reaching to $T^{\star}\left(\frac{n}{2}, \frac{n}{2}\right)$, yields the maximum number of required weighings. By these arguments, $T_{2}^{\star}(n)=T^{\star}\left(\frac{n}{2}, \frac{n}{2}\right)+1$. Similarly, it can be shown that $T_{2}^{\star}\left(\frac{n}{2}\right)=T^{\star}\left(\frac{n}{4}, \frac{n}{4}\right)+1$. Thus, $T_{2}^{\star}(n)=T_{2}^{\star}\left(\frac{n}{2}\right)+2$. More generally, we can write the recursive formula $T_{2}^{\star}\left(2^{i}\right)=T_{2}^{\star}\left(2^{i-1}\right)+2$ for all $1<i \leq l$. Noting that $T_{2}^{\star}(2)=1$, by solving the above recursion, we have $T_{2}^{\star}(n)=2 \log _{2} n-1$.

### 2.5 Optimal Nested Weighing Strategy

In a nested strategy, followed by weighing a subset $S$ of coins, if the weight of some coin(s) in $S$ remains undetermined, the next weighing must be performed on a proper subset of $S$. Moreover, if there are multiple such subsets $S$, this procedure must be performed separately for each $S$.

### 2.5.1 Average-Case Setting

For any collection $S$ of coins, denote by $d(S)$ the number of coins in $S$ with non-zero weight. For any $1 \leq s \leq n, w \in\{1,2\}$, and $d \in\{1,2\}$, denote by $\Psi_{d}(s, w)$ an optimal nested strategy for all collections $S$ of coins, each with an unknown weight in the set $\{0,1,2\}$, such that $|S|=s, w(S)=w$, and $d(S)=d$. That is, the expected number of weighings required by the strategy $\Psi_{d}(s, w)$ over all such $S$ (for any given $s, w$, and $d$ ) is minimum, among all possible nested strategies. Similarly, define $\Psi(s, w)$ as $\Psi_{d}(s, w)$, except when the expectation is taken over all $S$ such that $|S|=s$ and $w(S)=w$, and define the strategy $\Psi$ as $\{\Psi(s, w)\}_{1 \leq s \leq n, 1 \leq w \leq 2}$. We wish to design the strategy $\Psi$ and analyze $T_{\text {ave }}^{\Psi}(n)$.

Take an arbitrary collection $S$ of coins such that $|S|=s, w(S)=w$, and $d(S)=d$. Consider the application of a nested strategy, represented by $\Psi_{d}^{m}(s, w)$, on $S$ as follows. The strategy $\Psi_{d}^{m}(s, w)$ begins with weighing an arbitrary subset $R$ of coins in $S$ of size $1 \leq m \leq|S|-1$. If $w(R)=0$ or $w(R)=2$, the strategy $\Psi_{d}^{m}(s, w)$ proceeds with applying the strategy $\Psi_{d}(s-m, w)$ on $S \backslash R$, or the strategy $\Psi_{d}(m, w)$ on $R$, respectively. Otherwise, the strategy $\Psi_{d}^{m}(s, w)$ applies the strategies $\Psi_{d}(m, 1)$ and $\Psi_{d}(s-m, 1)$ on $R$ and $S \backslash R$, respectively. Denote by $T_{w, d}^{m}(s)$ the expected number of weighings required by the strategy $\Psi_{d}^{m}(s, w)$ over all such $S$, and let $T_{w, d}^{\text {opt }}(s) \triangleq \min _{1 \leq m \leq s-1} T_{w, d}^{m}(s)$. Similarly, define the strategy $\Psi^{m}(s, w)$ the same as $\Psi_{d}^{m}(s, w)$, except when $\Psi_{d}$ is replaced by $\Psi$ everywhere. Denote by $T_{w}^{m}(s)$ the expected number of weighings required by the strategy $\Psi^{m}(s, w)$
over all $S$ such that $|S|=s$ and $w(S)=w$, and let $T_{w}^{\text {opt }}(s) \triangleq \min _{1 \leq m \leq s-1} T_{w}^{m}(s)$. A simple recursive argument yields that for the strategy $\Psi$ defined earlier, we have $T_{\text {ave }}^{\Psi}(n)=$ $T_{2}^{\mathrm{opt}}(n)$.

For the ease of notation, for any $2 \leq s \leq n$ and $1 \leq m \leq s-1$, we define $\alpha_{i, j}(s, m) \triangleq$ $\binom{s-i}{m-j} /\binom{s}{m}$ for all $i, j$ such that $0 \leq m-j \leq s-i$, and define $\alpha_{i, j}(s, m) \triangleq 0$, otherwise. For brevity, we simply refer to $\alpha_{i, j}(s, m)$ by $\alpha_{i, j}$ whenever $s$ and $m$ are clear from the context. Based on the above definitions, the following results can be shown.

Lemma 4. For any $2 \leq s \leq n$ and $1 \leq m \leq s-1$, we have

$$
T_{1}^{m}(s)=\alpha_{1,0}\left(T_{1}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{1,1}\left(T_{1}^{\mathrm{opt}}(m)+1\right)
$$

where $T_{1}^{\text {opt }}(1)=0$. Moreover, for any $3 \leq s \leq n$ and $1 \leq m \leq s-1$, we have

$$
\begin{gathered}
T_{2}^{m}(s)=\frac{2}{s+1} T_{2,1}^{m}(s)+\frac{s-1}{s+1} T_{2,2}^{m}(s) \\
T_{2,1}^{m}(s)=\alpha_{1,0}\left(T_{2,1}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{1,1}\left(T_{2,1}^{\mathrm{opt}}(m)+1\right)
\end{gathered}
$$

and
$T_{2,2}^{m}(s)=\alpha_{2,0}\left(T_{2,2}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{2,2}\left(T_{2,2}^{\mathrm{opt}}(m)+1\right)+2 \alpha_{2,1}\left(T_{1}^{\mathrm{opt}}(m)+T_{1}^{\mathrm{opt}}(s-m)+1\right)$,
where $T_{2,1}^{\mathrm{opt}}(1)=T_{2,2}^{\mathrm{opt}}(1)=0$, and $T_{2,1}^{\mathrm{opt}}(2)=T_{2,2}^{\mathrm{opt}}(2)=1$.

Proof. Consider an arbitrary collection of $s$ coins of total weight $w$. There are two cases: (i) $w=1$, and (ii) $w=2$. In the case (i), a randomly chosen subset of $m$ coins weighs 0 with probability $\alpha_{1,0}$, and it weighs 1 with probability $\alpha_{1,1}$. In these two sub-cases the expected number of extra required weighings is $T_{1,1}^{\mathrm{opt}}(s-m)$ and $T_{1,1}^{\mathrm{opt}}(m)$, respectively.

Note that $T_{1,1}^{m}(t)=T_{1}^{m}(t)$ for all $t$, and so, $T_{1,1}^{\text {opt }}(t)=T_{1}^{\text {opt }}(t)$ for all $t$. Thus,

$$
T_{1}^{m}(s)=\alpha_{1,0}\left(T_{1}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{1,1}\left(T_{1}^{\mathrm{opt}}(m)+1\right)
$$

In the case (ii), there are two sub-cases: (ii-1) there is one coin of weight 2 (there exist $s$ distinct sub-configurations with this characteristic), and (ii-2) there are two coins, each of weight 1 (there exist $\binom{s}{2}$ distinct such sub-configurations). Thus,

$$
T_{2}^{m}(s)=\frac{2}{s+1} T_{2,1}^{m}(s)+\frac{s-1}{s+1} T_{2,2}^{m}(s) .
$$

In the case (ii-1), a randomly chosen subset of $m$ coins weighs 0 or 2 with probability $\alpha_{1,0}$ or $\alpha_{1,1}$, respectively, and in these two sub-cases the expected number of extra required weighings is $T_{2,1}^{\mathrm{opt}}(s-m)$ and $T_{2,1}^{\mathrm{opt}}(m)$, respectively. Thus,

$$
T_{2,1}^{m}(s)=\alpha_{1,0}\left(T_{2,1}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{1,1}\left(T_{2,1}^{\mathrm{opt}}(m)+1\right)
$$

Similarly, in the case (ii-2), a randomly chosen subset of $m$ coins weighs 0 , or 1 , or 2 with probability $\alpha_{2,0}$, or $2 \alpha_{2,1}$, or $\alpha_{2,2}$, respectively. In these three sub-cases, the expected number of extra required weighings is $T_{2,2}^{\mathrm{opt}}(s-m), T_{1}^{\mathrm{opt}}(m)+T_{1}^{\mathrm{opt}}(s-m)$, and $T_{2,2}^{\mathrm{opt}}(m)$, respectively. Thus,
$T_{2,2}^{m}(s)=\alpha_{2,0}\left(T_{2,2}^{\mathrm{opt}}(s-m)+1\right)+2 \alpha_{2,1}\left(T_{1}^{\mathrm{opt}}(m)+T_{1}^{\mathrm{opt}}(s-m)+1\right)+\alpha_{2,2}\left(T_{2,2}^{\mathrm{opt}}(m)+1\right)$.

This completes the proof.

Lemma 5. For any $2 \leq s \leq n$, we have $\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in \operatorname{argmin}_{1 \leq m \leq s-1} T_{1}^{m}(s)$; for any $3 \leq s \leq n$ and $d \in\{1,2\}$, we have $\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in \operatorname{argmin}_{1 \leq m \leq s-1} T_{2, d}^{m}(s)$; and for any $3 \leq s \leq n$, we have $\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in \operatorname{argmin}_{1 \leq m \leq s-1} T_{2}^{m}(s)$.

Proof. The proof techniques are the same for $T_{1}^{m}(s), T_{2, d}^{m}(s)$, and $T_{2}^{m}(s)$, and we only state the proof for $T_{1}^{m}(s)$ to avoid repetition. In particular, we shall show that for any $2 \leq s \leq n$, we have $\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in \operatorname{argmin}_{1 \leq m \leq s-1} T_{1}^{m}(s)$. The proof is by induction on $s$. It is easy to see that for $s=2$, we have $\left\lfloor\frac{2}{2}\right\rfloor,\left\lceil\frac{2}{2}\right\rceil=1=\operatorname{argmin}_{1 \leq m \leq 2-1} T_{1}^{m}(2)$. The induction hypothesis is that for $s \leq l-1$,

$$
\begin{equation*}
\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in \underset{1 \leq m \leq s-1}{\operatorname{argmin}} T_{1}^{m}(s) \tag{2.1}
\end{equation*}
$$

holds. To complete the proof, it is enough to show that for $s=l$, we have $\left\lfloor\frac{l}{2}\right\rfloor,\left\lceil\frac{l}{2}\right\rceil \in$ $\operatorname{argmin}_{1 \leq m \leq l-1} T_{1}^{m}(l)$. Based on the formula for $T_{1}^{m}(s)$ in Lemma 4, it can be readily confirmed that $T_{1}^{m}(s)$ is symmetric around midrange point, i.e., $T_{1}^{\left\lceil\frac{s}{2}\right\rceil}(s)=T_{1}^{\left\lfloor\frac{s}{2}\right\rfloor}(s)$. Thus, showing the proof for $\left\lfloor\frac{s}{2}\right\rfloor$ suffices. More specifically, we need to show the following:
for all $1 \leq m \leq l-1$. Due to the symmetry, it suffices to show that (2.2) holds for all $1 \leq m \leq\left\lfloor\frac{l}{2}\right\rfloor$. We consider two cases: (i) $l=2 k$, and (ii) $l=2 k+1$, for some $k \geq 1$.

## Proof for Case (i)

Noting that $l=2 k$ and $\left\lfloor\frac{l}{2}\right\rfloor=k$, (2.2) can be written as

$$
\frac{\binom{2 k-1}{k}}{\binom{2 k}{k}} T_{1}^{\mathrm{opt}}(k)+\frac{\binom{2 k-1}{k-1}}{\binom{2 k}{k}} T_{1}^{\mathrm{opt}}(k) \leq \frac{\binom{2 k-1}{m}}{\binom{2 k}{m}} T_{1}^{\mathrm{opt}}(2 k-m)+\frac{\binom{2 k-1}{m-1}}{\binom{2 k}{m}} T_{1}^{\mathrm{opt}}(m),
$$

or equivalently,

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(k) \leq\left(\frac{2 k-m}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-m)+\left(\frac{m}{2 k}\right) T_{1}^{\mathrm{opt}}(m) \tag{2.3}
\end{equation*}
$$

for all $1 \leq m \leq k$. An inductive argument (on $m$ ) is used to prove that (2.3) holds. It is easy to see that (2.3) holds for $m=k$, i.e.,

$$
T_{1}^{\mathrm{opt}}(k)=\left(\frac{2 k-k}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-k)+\left(\frac{k}{2 k}\right) T_{1}^{\mathrm{opt}}(k) .
$$

We assume that (2.3) holds for $m>t$. We need to show that for $m=t$, the following holds:

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(k) \leq\left(\frac{2 k-t}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-t)+\left(\frac{t}{2 k}\right) T_{1}^{\mathrm{opt}}(t) \tag{2.4}
\end{equation*}
$$

The proof is by contradiction. Suppose that (2.4) does not hold, i.e.,

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(k)>\left(\frac{2 k-t}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-t)+\left(\frac{t}{2 k}\right) T_{1}^{\mathrm{opt}}(t) \tag{2.5}
\end{equation*}
$$

Since (2.3) holds for $m>t$ (by assumption), for $m=t+1$ we have

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(k) \leq\left(\frac{2 k-t-1}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-t-1)+\left(\frac{t+1}{2 k}\right) T_{1}^{\mathrm{opt}}(t+1) . \tag{2.6}
\end{equation*}
$$

Combining (2.5) and (2.6), we get

$$
\begin{aligned}
\left(\frac{2 k-t}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-t) & +\left(\frac{t}{2 k}\right) T_{1}^{\mathrm{opt}}(t) \\
& <\left(\frac{2 k-t-1}{2 k}\right) T_{1}^{\mathrm{opt}}(2 k-t-1)+\left(\frac{t+1}{2 k}\right) T_{1}^{\mathrm{opt}}(t+1)
\end{aligned}
$$

which equivalently can be written as

$$
\begin{equation*}
(2 k-t) T_{1}^{\mathrm{opt}}(2 k-t)-(2 k-t-1) T_{1}^{\mathrm{opt}}(2 k-t-1)<(t+1) T_{1}^{\mathrm{opt}}(t+1)-(t) T_{1}^{\mathrm{opt}}(t) \tag{2.7}
\end{equation*}
$$

We need to disprove (2.7). Before moving further with disproving (2.7), we shall show the following formula which will be used in the rest of the proof:

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(q)-T_{1}^{\mathrm{opt}}(q-1)=\frac{2^{\lfloor\log (q-1)\rfloor+1}}{q(q-1)} \tag{2.8}
\end{equation*}
$$

for all $2 \leq q \leq l-1$.
The proof of (2.8) is based on an inductive argument. It is easy to see that for $q=2$, we have $T_{1}^{\mathrm{opt}}(2)-T_{1}^{\mathrm{opt}}(2-1)=1-0=\frac{2^{\lfloor\log (2-1)\rfloor+1}}{2(2-1)}=1$. We assume that for $q \leq h-1$, (2.8) holds. It suffices to show that for $q=h$ we have

$$
T_{1}^{\mathrm{opt}}(h)-T_{1}^{\mathrm{opt}}(h-1)=\frac{2^{\lfloor\log (h-1)\rfloor+1}}{h(h-1)}
$$

We consider two cases: (i-1) $h=2 d$, and (i-2) $h=2 d+1$, for some $d \geq 1$. First, consider the case (i-1). Since $h \leq l-1$ and (2.1) holds for $s \leq l-1$, we have $\left\lfloor\frac{h}{2}\right\rfloor \in$ $\operatorname{argmin}_{1 \leq m \leq h-1} T_{1}^{m}(h)$ and $\left\lfloor\frac{h-1}{2}\right\rfloor \in \operatorname{argmin}_{1 \leq m \leq h-2} T_{1}^{m}(h-1)$. Using the formula in Lemma 4 and noting that $h=2 d,\left\lfloor\frac{h}{2}\right\rfloor=d$, and $\left\lfloor\frac{h-1}{2}\right\rfloor=d-1, T_{1}^{\text {opt }}(h)$ and $T_{1}^{\text {opt }}(h-1)$ can be written as

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(h)=T_{1}^{\mathrm{opt}}(d)+1, \tag{2.9}
\end{equation*}
$$

and

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(h-1)=\left(\frac{d}{2 d-1}\right) T_{1}^{\mathrm{opt}}(d)+\left(\frac{d-1}{2 d-1}\right) T_{1}^{\mathrm{opt}}(d-1)+1 . \tag{2.10}
\end{equation*}
$$

Subtracting (2.10) from (2.9) results in

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(h)-T_{1}^{\mathrm{opt}}(h-1)=\left(\frac{d-1}{2 d-1}\right)\left(T_{1}^{\mathrm{opt}}(d)-T_{1}^{\mathrm{opt}}(d-1)\right) \tag{2.11}
\end{equation*}
$$

Since (2.8) holds for $q \leq h-1$ (by assumption), we have

$$
\left(T_{1}^{\text {opt }}(d)-T_{1}^{\text {opt }}(d-1)\right)=\frac{2^{\lfloor\log (d-1)\rfloor+1}}{d(d-1)}
$$

Substituting $\left(T_{1}^{\text {opt }}(d)-T_{1}^{\text {opt }}(d-1)\right)$ by $\frac{2^{\lfloor\log (d-1)\rfloor+1}}{d(d-1)}$ in (2.11), we have

$$
T_{1}^{\mathrm{opt}}(h)-T_{1}^{\mathrm{opt}}(h-1)=\frac{2^{\lfloor\log (d-1)\rfloor+1}}{d(2 d-1)}=\frac{2^{\lfloor\log (2 d-2)\rfloor+1}}{2 d(2 d-1)}=\frac{2^{\lfloor\log (2 d-1)\rfloor+1}}{2 d(2 d-1)}=\frac{2^{\lfloor\log (h-1)\rfloor+1}}{h(h-1)}
$$

This completes the proof of (2.8) for the case (i-1). Now, consider the case (i-2). Noting that $h=2 d+1$ and using similar arguments as above, it can be shown that

$$
T_{1}^{\mathrm{opt}}(h)-T_{1}^{\mathrm{opt}}(h-1)=\left(\frac{d+1}{2 d+1}\right)\left(T_{1}^{\mathrm{opt}}(d+1)-T_{1}^{\mathrm{opt}}(d)\right)
$$

and

$$
T_{1}^{\mathrm{opt}}(d+1)-T_{1}^{\mathrm{opt}}(d)=\frac{2^{\lfloor\log (d)\rfloor+1}}{d(d+1)}
$$

Subsequently, we have

$$
T_{1}^{\mathrm{opt}}(h)-T_{1}^{\mathrm{opt}}(h-1)=\frac{2^{\lfloor\log (d)\rfloor+1}}{d(2 d+1)}=\frac{2^{\lfloor\log (2 d)\rfloor+1}}{2 d(2 d+1)}=\frac{2^{\lfloor\log (h-1)\rfloor+1}}{h(h-1)} .
$$

This completes the proof of (2.8) for the case (i-2).
Now, we proceed with disproving (2.7). By using (2.8) and substituting $T_{1}^{\mathrm{opt}}(2 k-t)$ by $T_{1}^{\mathrm{opt}}(2 k-t-1)+\frac{2^{\lfloor\log 2 k-t-1\rfloor+1}}{(2 k-t-1)(2 k-t)}$ and $T_{1}^{\mathrm{opt}}(t+1)$ by $T_{1}^{\mathrm{opt}}(t)+\frac{2^{\lfloor\log t\rfloor+1}}{t(t+1)}$ in $(2.7)$, we have

$$
T_{1}^{\mathrm{opt}}(2 k-t-1)+\frac{2^{\lfloor\log 2 k-t-1\rfloor+1}}{2 k-t-1}<T_{1}^{\mathrm{opt}}(t)+\frac{2^{\lfloor\log t\rfloor+1}}{t}
$$

which equivalently can be written as

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(2 k-t-1)-T_{1}^{\mathrm{opt}}(t)<\frac{2^{\lfloor\log t\rfloor+1}}{t}-\frac{2^{\lfloor\log 2 k-t-1\rfloor+1}}{2 k-t-1} \tag{2.12}
\end{equation*}
$$

To complete disproving (2.7), we need to show that (2.12) does not hold. To this end, we need to prove the following:

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(a+i)-T_{1}^{\mathrm{opt}}(a) \geq \frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+i\rfloor+1}}{a+i} \tag{2.13}
\end{equation*}
$$

for all $1 \leq a \leq l-2$ and $1 \leq i \leq l-1-a$. The proof of (2.13) is based on an inductive argument (on $i$ ). For $i=1$, we need to show that

$$
T_{1}^{\mathrm{opt}}(a+1)-T_{1}^{\mathrm{opt}}(a) \geq \frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+1\rfloor+1}}{a+1}
$$

Using (2.8), we have

$$
T_{1}^{\mathrm{opt}}(a+1)-T_{1}^{\mathrm{opt}}(a)=\frac{2^{\lfloor\log a\rfloor+1}}{a(a+1)}>\frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+1\rfloor+1}}{a+1}
$$

and subsequently, (2.13) holds for $i=1$. Next, we assume that for $i=b$, we have

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(a+b)-T_{1}^{\mathrm{opt}}(a) \geq \frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+b\rfloor+1}}{a+b} \tag{2.14}
\end{equation*}
$$

It is enough to show that for $i=b+1$, we have

$$
T_{1}^{\mathrm{opt}}(a+b+1)-T_{1}^{\mathrm{opt}}(a) \geq \frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+b+1\rfloor+1}}{a+b+1}
$$

We use proof by contradiction. Assume that

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(a+b+1)-T_{1}^{\mathrm{opt}}(a)<\frac{2^{\lfloor\log a\rfloor+1}}{a}-\frac{2^{\lfloor\log a+b+1\rfloor+1}}{a+b+1} \tag{2.15}
\end{equation*}
$$

Combining (2.14) and (2.15), we have

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(a+b+1)-T_{1}^{\mathrm{opt}}(a+b)<\frac{2^{\lfloor\log a+b\rfloor+1}}{a+b}-\frac{2^{\lfloor\log a+b+1\rfloor+1}}{a+b+1} \tag{2.16}
\end{equation*}
$$

Using (2.8), we have

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(a+b+1)-T_{1}^{\mathrm{opt}}(a+b)=\frac{2^{\lfloor\log (a+b)(a+b+1)\rfloor+1}}{a+b} \geq \frac{2^{\lfloor\log a+b\rfloor+1}}{a+b}-\frac{2^{\lfloor\log a+b+1\rfloor+1}}{a+b+1} \tag{2.17}
\end{equation*}
$$

Putting (2.16) and (2.17) together, we arrive at a contradiction, and consequently, (2.13) holds.

Now, we can disprove (2.12). Taking $a=t$ and $i=2 k-2 t-1$ in (2.13), we have

$$
T_{1}^{\mathrm{opt}}(2 k-t-1)-T_{1}^{\mathrm{opt}}(t) \geq \frac{2^{\lfloor\log t\rfloor+1}}{t}-\frac{2^{\lfloor\log 2 k-t-1\rfloor+1}}{2 k-t-1}
$$

which readily contradicts (2.12). This completes the disproof of (2.7), and consequently, the proof for the case (i).

Proof for Case (ii)
Noting that $l=2 k+1$ and $\left\lfloor\frac{l}{2}\right\rfloor=k$, (2.2) can be written as

$$
\frac{\binom{2 k}{k}}{\binom{2 k+1}{k}} T_{1}^{\mathrm{opt}}(k+1)+\frac{\binom{2 k}{k-1}}{\binom{2 k+1}{k}} T_{1}^{\mathrm{opt}}(k) \leq \frac{\binom{2 k}{m}}{\binom{2 k+1}{m}} T_{1}^{\mathrm{opt}}(2 k-m+1)+\frac{\binom{2 k}{m-1}}{\binom{k+1}{m}} T_{1}^{\mathrm{opt}}(m)
$$

or equivalently,

$$
\begin{align*}
\left(\frac{k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k & +1)+\left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) \\
& \leq\left(\frac{2 k-m+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-m+1)+\left(\frac{m}{2 k+1}\right) T_{1}^{\mathrm{opt}}(m) \tag{2.18}
\end{align*}
$$

for all $1 \leq m \leq k$. We use an inductive argument to prove that (2.18) holds. It is easy to see that (2.18) holds for $m=k$, i.e.,

$$
\begin{aligned}
\left(\frac{k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k+1)+ & \left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) \\
& =\left(\frac{2 k-k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-k+1)+\left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) .
\end{aligned}
$$

We assume that (2.18) holds for $m>t$. We need to show that for $m=t$, the following holds:

$$
\begin{align*}
\left(\frac{k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k+1) & +\left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) \\
& \leq\left(\frac{2 k-t+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-t+1)+\left(\frac{t}{2 k+1}\right) T_{1}^{\mathrm{opt}}(t) \tag{2.19}
\end{align*}
$$

The proof is by contradiction. Suppose that (2.19) does not hold, i.e.,

$$
\begin{align*}
\left(\frac{k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k+1) & +\left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) \\
& >\left(\frac{2 k-t+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-t+1)+\left(\frac{t}{2 k+1}\right) T_{1}^{\mathrm{opt}}(t) \tag{2.20}
\end{align*}
$$

We need to disprove (2.20). Since (2.18) holds for $m>t$ (by assumption), for $m=t+1$
we have

$$
\begin{align*}
\left(\frac{k+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k+1)+ & \left(\frac{k}{2 k+1}\right) T_{1}^{\mathrm{opt}}(k) \\
& \leq\left(\frac{2 k-t}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-t)+\left(\frac{t+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(t+1) \tag{2.21}
\end{align*}
$$

Combining (2.20) and (2.21) yields

$$
\begin{aligned}
\left(\frac{2 k-t+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-t+1) & +\left(\frac{t}{2 k+1}\right) T_{1}^{\mathrm{opt}}(t) \\
& <\left(\frac{2 k-t}{2 k+1}\right) T_{1}^{\mathrm{opt}}(2 k-t)+\left(\frac{t+1}{2 k+1}\right) T_{1}^{\mathrm{opt}}(t+1)
\end{aligned}
$$

which equivalently can be written as

$$
\begin{equation*}
(2 k-t+1) T_{1}^{\mathrm{opt}}(2 k-t+1)-(2 k-t) T_{1}^{\mathrm{opt}}(2 k-t)<(t+1) T_{1}^{\mathrm{opt}}(t+1)-(t) T_{1}^{\mathrm{opt}}(t) \tag{2.22}
\end{equation*}
$$

Using (2.8) and substituting $T_{1}^{\mathrm{opt}}(2 k-t+1)$ by $T_{1}^{\mathrm{opt}}(2 k-t)+\frac{2^{\lfloor\log (2 k-t)\rfloor+1}}{(2 k-t+1)(2 k-t)}$ and $T_{1}^{\mathrm{opt}}(t+1)$ by $T_{1}^{\text {opt }}(t)+\frac{2^{[\log t\rfloor+1}}{t(t+1)}$ in (2.22), we have

$$
T_{1}^{\mathrm{opt}}(2 k-t)+\frac{2^{\lfloor\log (2 k-t)\rfloor+1}}{2 k-t}<T_{1}^{\mathrm{opt}}(t)+\frac{2^{\lfloor\log (t)\rfloor+1}}{t}
$$

which equivalently can be written as

$$
\begin{equation*}
T_{1}^{\mathrm{opt}}(2 k-t)-T_{1}^{\mathrm{opt}}(t)<\frac{2^{\lfloor\log t\rfloor+1}}{t}-\frac{2^{\lfloor\log (2 k-t)\rfloor+1}}{2 k-t} \tag{2.23}
\end{equation*}
$$

Taking $a=t$ and $i=2 k-2 t$ in (2.13), we have

$$
T_{1}^{\mathrm{opt}}(2 k-t)-T_{1}^{\mathrm{opt}}(t) \geq \frac{2^{\lfloor\log t\rfloor+1}}{t}-\frac{2^{\lfloor\log 2 k-t\rfloor+1}}{2 k-t}
$$

which readily contradicts (2.23). This completes the disproof of (2.20), and consequently, the proof for the case (ii). So far, we proved that for any $2 \leq s \leq n$, we have $\left\lfloor\frac{s}{2}\right\rfloor,\left\lceil\frac{s}{2}\right\rceil \in$ $\operatorname{argmin}_{1 \leq m \leq s-1} T_{1}^{m}(s)$. The same procedure can be taken to prove the other two claims.

Lemma 6. For any $0 \leq i \leq l$, we have $T_{1}^{\mathrm{opt}}\left(2^{i}\right)=i$, and $T_{2}^{\mathrm{opt}}\left(2^{i}\right)=\frac{(i-1) 2^{i+1}+i+2}{2^{i}+1}$.

Proof. By the results of Lemmas 4 and 5, the following recursive formulas can be shown:

$$
T_{1}^{\mathrm{opt}}(s)=\frac{\left\lceil\frac{s}{2}\right\rceil}{s} T_{1}^{\mathrm{opt}}\left(\left\lceil\frac{s}{2}\right\rceil\right)+\frac{\left\lfloor\frac{s}{2}\right\rfloor}{s} T_{1}^{\mathrm{opt}}\left(\left\lfloor\frac{s}{2}\right\rfloor\right)+1,
$$

for all $2 \leq s \leq n$, and

$$
T_{2,1}^{\mathrm{opt}}(s)=\frac{\left\lceil\frac{s}{2}\right\rceil}{s} T_{2,1}^{\mathrm{opt}}\left(\left\lceil\frac{s}{2}\right\rceil\right)+\frac{\left\lfloor\frac{s}{2}\right\rfloor}{s} T_{2,1}^{\mathrm{opt}}\left(\left\lfloor\frac{s}{2}\right\rfloor\right)+1,
$$

and

$$
\begin{aligned}
T_{2,2}^{\mathrm{opt}}(s) & =\frac{\left\lceil\frac{s}{2}\right\rceil\left(\left\lceil\frac{s}{2}\right\rceil-1\right)}{s(s-1)} T_{2,2}^{\mathrm{opt}}\left(\left\lceil\frac{s}{2}\right\rceil\right)+\frac{\left\lfloor\frac{s}{2}\right\rfloor\left(\left\lfloor\frac{s}{2}\right\rfloor-1\right)}{s(s-1)} T_{2,2}^{\mathrm{opt}}\left(\left\lfloor\frac{s}{2}\right\rfloor\right) \\
& +\frac{2\left\lfloor\frac{s}{2}\right\rfloor\left\lceil\frac{s}{2}\right\rceil}{s(s-1)}\left(T_{1}^{\mathrm{opt}}\left(\left\lceil\frac{s}{2}\right\rceil\right)+T_{1}^{\mathrm{opt}}\left(\left\lfloor\frac{s}{2}\right\rfloor\right)\right)+1,
\end{aligned}
$$

for all $3 \leq s \leq n$. Solving these recursions, it follows that $T_{1}^{\mathrm{opt}}\left(2^{i}\right)=i$ for all $1 \leq$ $i \leq l$, and $T_{2,1}^{\mathrm{opt}}\left(2^{i}\right)=i$ and $T_{2,2}^{\mathrm{opt}}\left(2^{i}\right)=\left(1+(i-1)\left(2^{i+1}-1\right)\right) /\left(2^{i}-1\right)$ for all $2 \leq i \leq l$. Noting that $T_{1}^{\mathrm{opt}}(1)=0$, it is easy to verify that $T_{1}^{\mathrm{opt}}\left(2^{i}\right)=i$ for all $0 \leq i \leq l$. Similarly, it follows that $T_{2}^{\text {opt }}(s)=\frac{2}{s+1} T_{2,1}^{\text {opt }}(s)+\frac{s-1}{s+1} T_{2,2}^{\text {opt }}(s)$ for all $3 \leq s \leq n$, and subsequently, $T_{2}^{\text {opt }}\left(2^{i}\right)=\left((i-1) 2^{i+1}+i+2\right) /\left(2^{i}+1\right)$ for all $2 \leq i \leq l$. Noting that $T_{2}^{\text {opt }}(1)=0$ and $T_{2}^{\text {opt }}(2)=1$, it is easy to see that $T_{2}^{\text {opt }}\left(2^{i}\right)=\left((i-1) 2^{i+1}+i+2\right) /\left(2^{i}+1\right)$ for all $0 \leq i \leq l$. This completes the proof.


Figure 2.3: The average-case and worst-case results for the proposed strategy, the optimal nested strategy, and the information-theoretic lower bound.

Recall that for the optimal nested strategy $\Psi$ defined earlier, we have $T_{\text {ave }}^{\Psi}(n)=$ $T_{2}^{\text {opt }}(n)$. Thus the following result is immediate by the result of Lemma 6.

Theorem 4. For the optimal nested strategy $\Psi$, we have $T_{\mathrm{ave}}(n)=\frac{2 n+1}{n+1} \log _{2} n-\frac{2(n-1)}{n+1}$.

### 2.5.2 Worst-Case Setting

Consider an optimal nested strategy $\Psi^{\star}$ for the worst-case setting, defined similarly as the strategy $\Psi$ for the average-case setting, except when considering the maximum number of required weighings (instead of the expected number of required weighings). Then, the following result holds [13].

Theorem 5. [13] For the optimal nested strategy $\Psi^{\star}$, we have $T_{\max }(n)=2 \log _{2} n-1$.

### 2.6 Comparison Results

In this section, we present our numerical results for the performance of the proposed strategy in both the average-case and worst-case settings. For each setting, the perfor-
mance of the proposed strategy is compared with the performance of the optimal nested strategy (defined in Section 2.5) and the information-theoretic lower bound (Theorem 1).

Fig. 2.3 illustrates that the proposed strategy, in the average-case setting, significantly outperforms the optimal nested strategy. Also, in the worst-case setting, the proposed strategy achieves the same performance as the nested strategy. Our numerical evaluations suggest that the expected number of weighings required by the proposed strategy, which is computable using the recursive formulas in Section 2.4, can be also approximated by $1.365 \log _{2} n-0.5$ as $n$ grows unbounded (see Fig. 2.3). In this asymptotic regime, the optimal nested strategy requires $2 \log _{2} n-2$ weighings on average, and the informationtheoretic lower bound is $2 \log _{3} n \approx 1.262 \log _{2} n$. Thus, a simple calculation shows that the proposed strategy, when compared to the optimal nested strategy, requires about $31.75 \%$ less number of weighings on average. Additionally, when compared to the informationtheoretic lower bound, the proposed strategy requires at most about $8.16 \%$ extra number of weighings on average.

## 3. NON-ADAPTIVE QUANTITATIVE GROUP TESTING USING BI-REGULAR SPARSE GRAPH CODES*

### 3.1 Introduction

In this chapter, we consider the problem of Quantitative Group Testing (QGT). Consider a set of $N$ items among which $K$ items are defective. The QGT problem is to identify (all or a sufficiently large fraction of) the defective items, where the result of a test reveals the number of defective items in the tested group. The key difference between the QGT problem and the original group testing problem is that, unlike the former, in the latter the result of each test is either 1 or 0 depending on whether the tested group contains any defective items or not. The objective of QGT is to design a test plan with minimum number of tests that identifies (all or a sufficiently large fraction of) the defective items.

There are two general categories of test strategies: non-adaptive and adaptive. In an adaptive scheme, each test depends on the outcomes of the previous tests. On the other hand, in a non-adaptive scheme, all tests are planned in advance. In other words, the result of one test does not affect the design of another test.

Let $S$ be the index set of the defective items and $\hat{S}$ be an estimation of $S$. Depending on the application at hand, there can be different requirements for the closeness of $\hat{S}$ to $S[16,17]$. The strongest condition for closeness is exact recovery when it is required that $\hat{S}=S$. Two weaker conditions are partial recovery without false detections when it is required that $\hat{S} \subseteq S$ and $|\hat{S}| \geq(1-\epsilon)|S|$, and partial recovery without missed detections when it is required that $S \subseteq \hat{S}$ and $|\hat{S}| \leq(1+\epsilon)|S|$. There are also different types of the recovery guarantees [17]. The strongest guarantee is perfect recovery guarantee when

[^1]the exact or partial recovery needs to be achieved with probability 1 (over the space of all problem instances). A slightly weaker guarantee is probabilistic recovery guarantee when it suffices to achieve the exact or partial recovery with high probability only (and not necessarily with probability 1 ). In this work, we are interested in the exact recovery of all defective items with the probabilistic recovery guarantee.

### 3.1.1 Related Work and Applications

The QGT problem has been extensively studied for a wide range of applications, e.g., multi-access communication, spectrum sensing, and network tomography, see, e.g., [13], and references therein. This problem was first introduced by Shapiro in [35]. Several non-adaptive and adaptive QGT strategies have been previously proposed, see, e.g., [13, 14, 34]. It was shown in [36] that any non-adaptive algorithm must perform at least $\left(2 K \log _{2}(N / K)\right) / \log _{2} K$ tests. Various order optimal or near-optimal non-adaptive strategies were previously proposed, see, e.g., $[14,34,36]$. The best known polynomial-time non-adaptive algorithms require $K \log N$ tests [36, 44]. Recently, a semi-quantitative group testing scheme based on sparse graph codes was proposed in [45], where the result of each test is an integer in the set $\{0,1,2, \ldots, L\}$. This strategy identifies a $(1-\epsilon)$ fraction of defective items using $c(\epsilon, L) K \log _{2} N$ tests with high probability, where $c(\epsilon, L)$ depends only on $\epsilon$ and $L$.

### 3.1.2 Connection with Compressed Sensing

A closely related problem to QGT is the problem of compressed sensing (CS) in which the goal is to recover a sparse signal from a set of (linear) measurements. Given an N dimensional sparse signal with a support size up to $K$, the objective is to identify the indices and the values of non-zero elements of the signal with minimum number of measurements. The main differences between the CS problem and the QGT problem are in the signal model and the constraints on the measurement matrix. Most of the existing
works on the CS problem consider real-valued signals and measurement matrices. The QGT problem, however, deals with binary signals and requires the measurement matrix to be binary-valued. There are a number of CS algorithms in the literature that use binaryvalued measurement matrices, see, e.g. $[18,19]$ and references therein. However, these strategies either use techniques which are not applicable to binary signals, or provide different types of closeness and guarantee than those required in this work. There are also several CS algorithms for the support recovery where the objective is to determine the indices of the non-zero elements of the signal but not their values [20,21,46]. The support recovery problem is indeed equivalent to the QGT problem. Notwithstanding, the existing schemes for support recovery rely on non-binary measurement matrices, and hence are not suitable for the QGT problem. Last but not least, to the best of our knowledge, the majority of works on the CS problem focus mainly on the order optimality of the number of measurements, whereas in this work for the QGT problem we are also interested in minimizing the constant factor hidden in the order.

### 3.1.3 Main Contributions

In this work, we propose a non-adaptive quantitative group testing strategy for the sublinear regime where $\frac{K}{N}$ vanishes as $K, N \rightarrow \infty$. We utilize sparse graph codes over biregular bipartite graphs with left-degree $\ell$ and right-degree $r$ and binary $t$-error-correcting BCH codes for the design of the proposed strategy. Leveraging powerful density evolution techniques for the analysis enables us not only to determine the exact value of constants in the number of tests needed but also to provide provable performance guarantees. We show that the proposed scheme provides exact recovery with probabilistic guarantee, i.e. recovers all the defective items with high probability. In particular, for the sub-linear regime, the proposed algorithm requires at most $m=c(t) K\left(t \log _{2}\left(\frac{\ell N}{c(t) K}+1\right)+1\right)+1$ tests to recover all defective items with probability approaching one as $K, N \rightarrow \infty$, where
$c(t)$ depends only on $t$. The results of our theoretical analysis reveal that the minimum number of required tests for the proposed algorithm is achieved by $t=2$. Moreover, for any $t \leq 4$, the encoding and decoding of the proposed algorithm have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

### 3.2 Problem Setup and Notation

Let the vector $\mathbf{x} \in\{0,1\}^{N}$ represent the set of $N$ items in which the coordinates with value 1 correspond to the defective items. A non-adaptive group testing problem consisting of $m$ tests can be represented by a measurement matrix $\mathbf{A} \in\{0,1\}^{m \times N}$, where the $i$-th row of the matrix corresponds to the $i$-th test. That is, the coordinates with value 1 in the $i$-th row correspond to the items in the $i$-th test. The results of the $m$ tests are expressed in the test vector $\mathbf{y} \in\{0,1, \ldots\}^{m}$, i.e.,

$$
\begin{equation*}
\mathbf{y}=\left[y_{1}, \cdots, y_{m}\right]^{T}=\mathbf{A} \mathbf{x} . \tag{3.1}
\end{equation*}
$$

The goal is to design a testing matrix $\mathbf{A}$ that has a small number of rows (tests), $m$, and can identify with high probability all the defective items given the test vector $\mathbf{y}$.

### 3.3 Proposed Algorithm

### 3.3.1 Binary $t$-error-correcting codes and $t$-separable matrices

Definition 1. (t-separable matrix) A binary matrix $\mathbf{D} \in\{0,1\}^{m \times n}$ (for $n>t$ ) is $t$ separable over field $\mathbb{F}$ if the sum (over field $\mathbb{F}$ ) of any set of $t$ columns is distinct.

Example 2. Consider the following matrix,

$$
\mathbf{D}=\left[\begin{array}{llll}
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1
\end{array}\right]
$$

The matrix $\mathbf{D}$ is 2 -separable over real field $\mathbb{R}$, but it is not 2 -separable over $\mathbb{F}_{2}$ since, for instance, the sum of the first and second columns over $\mathbb{F}_{2}$ is the same as the sum of the third and fourth columns over $\mathbb{F}_{2}$.

$$
\left[\begin{array}{l}
0 \\
0 \\
0
\end{array}\right] \oplus\left[\begin{array}{l}
1 \\
1 \\
0
\end{array}\right]=\left[\begin{array}{l}
0 \\
1 \\
1
\end{array}\right] \oplus\left[\begin{array}{l}
1 \\
0 \\
1
\end{array}\right]=\left[\begin{array}{l}
1 \\
1 \\
0
\end{array}\right] .
$$

From the definition, it can be easily seen that if a matrix $\mathbf{D}$ (with $n$ columns) is $t$ separable over a field $\mathbb{F}$, then $\mathbf{D}$ is also $s$-separable over $\mathbb{F}$ for any $1 \leq s<t<n$. The vector of test results, $\mathbf{y}$, is the sum of the columns in the testing matrix corresponding to the coordinates of the defective items. When a $t$-separable matrix over $\mathbb{R}$ is used as the testing matrix, the vector $\mathbf{y}$ will be distinct for any set of $t$ defective items. Thus, a $t$-separable matrix over $\mathbb{R}$ can be used as the testing matrix for identifying $t$ defective items. However, the construction of $t$-separable matrices for arbitrary $t$ with minimum number of rows is an open problem. Instead, we can leverage the idea that the parity-check matrix of any binary $t$-error-correcting code is a $t$-separable matrix over $\mathbb{F}_{2}$. Note that $t$-separability over $\mathbb{F}_{2}$ results in $t$-separability over $\mathbb{R}$. Hence, a possible choice for designing a $t$-separable matrix over $\mathbb{R}$ is utilizing the parity-check matrix of a binary $t$-error-correcting code.

We use binary BCH codes for this purpose. The key feature of the BCH codes which make them suitable for designing $t$-separable matrices is that it is possible to design binary BCH codes, capable of correcting any combination of $t$ or fewer errors.

Definition 2. [47] (Binary BCH codes) For any positive integers $m \geq 3$ and $t<2^{m-1}$, there exists a binary t-error-correcting BCH code with the following parameters:

$$
\begin{cases}n=2^{m}-1 & \text { block length } \\ n-k \leq m t & \text { number of parity-check digits } \\ d_{\min } \geq 2 t+1 & \text { minimum Hamming distance }\end{cases}
$$

The parity-check matrix of such a code is given by $\mathbf{H}_{t}=\left(\alpha^{(2 i-1)(j-1)}\right)_{i \in\{1, \cdots, t\}, j \in\{1, \cdots, n\}}$ where $\alpha$ is a primitive element in $\mathbb{F}_{2^{m}}$.

Since each entry of $\mathbf{H}_{t}$ is an element in $\mathbb{F}_{2^{m}}$, it can be represented by an $m$-tuple over $\mathbb{F}_{2}$. Thus, the number of rows in the binary representation of $\mathbf{H}_{t}$ is given by $R=t m=$ $t \log _{2}(n+1)$.

### 3.3.2 Encoding algorithm

The design of the measurement matrix $\mathbf{A}$ in our scheme is based on an architectural philosophy that was proposed in [17] and [48]. The key idea is to design A using a sparse bi-regular bipartite graph and to apply a peeling-based iterative algorithm for recovering the defective items given $\mathbf{y}$.

Let $G_{\ell, r}(N, M)$ be a randomly generated bipartite graph where each of the $N$ left nodes is connected to $\ell$ right nodes uniformly at random, and each of the $M$ right nodes is connected to $r$ left nodes uniformly at random. Note that there are $N \ell$ edge connections from the left side and $M r$ edge connections from the right side,

$$
\begin{equation*}
N \ell=M r \tag{3.2}
\end{equation*}
$$

Let $\mathbf{T}_{G} \in\{0,1\}^{M \times N}$ be the adjacency matrix of $G_{\ell, r}(N, M)$, where each column in $\mathbf{T}_{\mathcal{G}}$ corresponds to a left node and has exactly $\ell$ ones, and each row corresponds to a right node and has exactly $r$ ones. Let $\mathbf{t}_{i} \in\{0,1\}^{N}$ denote the $i$-th row of $\mathbf{T}_{G}$, i.e., $\mathbf{T}_{G}=\left[\mathbf{t}_{1}^{T}, \mathbf{t}_{2}^{T}, \cdots, \mathbf{t}_{M}^{T}\right]^{T}$. We assign $s$ tests to each right node based on a signature matrix
$\mathbf{U} \in\{0,1\}^{s \times r}$. The matrix $\mathbf{U}$ is chosen as $\mathbf{U}=\left[\mathbf{1}_{1 \times r}^{T}, \mathbf{H}_{t}^{T}\right]^{T}$, where $\mathbf{1}_{1 \times r}$ is an all-ones row of length $r$, and $\mathbf{H}_{t} \in\{0,1\}^{t \log _{2}(r+1) \times r}$ is the parity-check matrix of a binary $t$-errorcorrecting BCH code of length $r$. We then have $s=R+1=t \log (r+1)+1$.

The measurement matrix is given by $\mathbf{A}=\left[\mathbf{A}_{1}^{T}, \cdots, \mathbf{A}_{M}^{T}\right]^{T}$ where $\mathbf{A}_{i} \in\{0,1\}^{s \times N}$ is a matrix that defines the $s$ tests at the $i$-th right node. There are exactly $r$ ones in each row $\mathbf{t}_{i}$ of $\mathbf{T}_{G}$, and the signature matrix $\mathbf{U}=\left[\mathbf{u}_{1}, \mathbf{u}_{2}, \cdots, \mathbf{u}_{r}\right]$ has $r$ columns. Note that $\mathbf{u}_{i}=\left[1, \mathbf{h}_{i}^{T}\right]^{T}$ is the $i$-th column of $\mathbf{U}$, where $\mathbf{h}_{i}$ is the $i$-th column of $\mathbf{H}_{t} . \mathbf{A}_{i}$ is obtained by placing the $r$ columns of $\mathbf{U}$ at the coordinates of the $r$ ones of the row vector $\mathbf{t}_{i}$, and replacing zeros by all-zero columns, $\mathbf{A}_{i}=\left[\mathbf{0}, \ldots, \mathbf{0}, \mathbf{u}_{1}, \mathbf{0}, \ldots, \mathbf{u}_{2}, \mathbf{0}, \ldots, \mathbf{u}_{r}\right]$ where $\mathbf{t}_{i}=[0, \ldots, 0,1,0, \ldots, 1,0, \ldots, 1]$. The number of rows in the matrix $\mathbf{A}, m=M \times s$ where $s=t \log _{2}(r+1)+1$, represents the total number of tests in the proposed scheme.

Example 3. Let $N=14$ be the total number of items. Let $G$ be a randomly generated bi-regular graph with $N$ left nodes of degree $\ell=2$ and $M=4$ right nodes of degree $r=7$. For this example, suppose that the adjacency matrix $\mathbf{T}_{G}$ of the graph $G$ is given by

$$
\mathbf{T}_{\mathcal{G}}=\left[\begin{array}{llllllllllllll}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0
\end{array}\right]
$$

The parity-check matrix of a binary 1-error-correcting BCH code of length 7 given by

$$
\mathbf{H}_{1}=\left[\begin{array}{llll}
1 & \alpha & \cdots & \alpha^{6}
\end{array}\right]=\left[\begin{array}{lllllll}
0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}\right]
$$

where $\alpha \in \mathbb{F}_{2^{3}}$ is a root of the primitive polynomial $\alpha^{3}+\alpha+1=0$. The signature matrix
$\mathbf{U}=\left[\mathbf{1}_{1 \times 7}^{T}, \mathbf{H}_{1}^{T}\right]^{T}$ is then given by

$$
\mathbf{U}=\left[\begin{array}{lllllll}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}\right]
$$

Following the construction procedure explained earlier, the testing matrix $\mathbf{A}$ is then given by

$$
\mathbf{A}=\left[\begin{array}{llllllllllllll}
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
\hline 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}\right] .
$$

### 3.3.3 Decoding algorithm

Let the observation vector corresponding to the $i$-th right node be defined as

$$
\begin{equation*}
\mathbf{z}_{i}=\left[z_{i, 1}, z_{i, 2}, \cdots, z_{i, s}\right]^{T}=\mathbf{A}_{i} \mathbf{x}, \forall i \in\{1, \cdots, M\} \tag{3.3}
\end{equation*}
$$

Note that $\mathbf{z}_{i}=\left[y_{(i-1) s+1}, \cdots, y_{i s}\right]^{T}$.

Definition 3. ( $t$-resolvable right node) A right node is called $t$-resolvable if it is connected to tor fewer defective items.

The following lemma is useful for resolving the right nodes.

Lemma 7. The proposed algorithm detects and resolves all the t-resolvable right nodes.
Proof. Let us divide $\mathbf{z}_{i}$ into two blocks, $\mathbf{z}_{i}=\left[\mathbf{z}_{i}^{(1)^{T}}, \mathbf{z}_{i}^{(2)^{T}}\right]^{T}$, where $\mathbf{z}_{i}^{(1)}=z_{i, 1}$ and $\mathbf{z}_{i}^{(2)}=\left[z_{i, 2}, \cdots, z_{i, s}\right]^{T}$. We can rewrite (3.3) by placing $\left[1, \mathbf{h}_{i}^{T}\right]^{T}$ at the coordinates of $\mathbf{u}_{i}$ 's,

$$
\left[\begin{array}{l}
\mathbf{z}_{i}^{(1)} \\
\mathbf{z}_{i}^{(2)}
\end{array}\right]=\left[\begin{array}{llllllllll}
0 & \ldots & 0 & 1 & 0 & \ldots & 1 & 0 & \ldots & 1 \\
\mathbf{0} & \ldots & \mathbf{0} & \mathbf{h}_{1} & \mathbf{0} & \ldots & \mathbf{h}_{2} & \mathbf{0} & \ldots & \mathbf{h}_{r}
\end{array}\right] \mathbf{x} .
$$

Assume that $j \leq t$ defective items are connected to the $i$-th right node. The first block, $\mathbf{z}_{i}^{(1)}$, which is the first element of $\mathbf{z}_{i}$, shows the number of defective items connected to the $i$-th right node. Recall that the first row of the signature matrix is an all-ones vector. It means that there are $r$ ones in the first row of every $\mathbf{A}_{i}, i \in\{1,2, \cdots, M\}$. Thus, all $r$ items connected to the $i$-th right node are included in the test corresponding to the first row of $\mathbf{A}_{i}$. The second block, $\mathbf{z}_{i}^{(2)}$, is equal to the sum of $\mathbf{h}_{i}$ 's corresponding to the defective items connected to the $i$-th right node. Let $S_{i}$ be the set of indices of items (left nodes) that are connected to the $i$-th right node, and let $\mathbf{x}_{S_{i}}$ be the vector $\mathbf{x}$ restricted to the items indexed by $S_{i}$. Note that $\mathbf{x}_{S_{i}}$ can be viewed as an error vector for a $t$-error-correcting BCH code
with parity-check matrix $\mathbf{H}_{t}$, and the block vector $\mathbf{z}_{i}^{(2)}$ under modulo 2 can be interpreted as the syndrome corresponding to the error vector $\mathbf{x}_{S_{i}}$. The Hamming weight of the error vector $X_{S_{i}}$, i.e., the number of ones in $\mathbf{x}_{S_{i}}$, is equal to $j$. When $j \leq t$, the error vector $\mathbf{x}_{S_{i}}$ can be decoded from the corresponding syndrome by decoding the underlying BCH code, and hence all $j$ defective items connected to the $i$-th right node can be identified.

The decoding algorithm performs in rounds as follows. In each round, the decoding algorithm first iterates through all the right node observation vectors $\left\{\mathbf{z}_{i}\right\}_{i=1}^{M}$, and resolves all $t$-resolvable right nodes (by BCH decoding, as discussed in the proof of Lemma 7). Then, given the identities of the recovered left nodes, the edges connected to these defective items are peeled off the graph. That is, the contributions of the recovered defective items will be removed from the unresolved right nodes so that new right nodes may become $t$-resolvable for the next round. The decoding algorithm terminates when there is no more $t$-resolvable right nodes.

Example 4. Consider the group testing problem in the Example 3. Let the number of defective items be $K=3$ and let

$$
\mathbf{x}=[1,0,0,1,0,0,0,0,0,1,0,0,0,0]^{T},
$$

i.e., item 1, item 4, and item 10 are defective items. We show how the proposed scheme can identify the defective items. The result of the tests can be expressed as follows,

$$
\mathbf{y}=\left[\begin{array}{l}
\mathbf{z}_{1} \\
\mathbf{z}_{2} \\
\mathbf{z}_{3} \\
\mathbf{z}_{4}
\end{array}\right]=\mathbf{A x}=\left[\begin{array}{c}
\mathbf{u}_{1} \\
\mathbf{u}_{5} \\
\mathbf{u}_{2}+\mathbf{u}_{5} \\
\mathbf{u}_{1}+\mathbf{u}_{2}
\end{array}\right]
$$

Then, the right-node observation vectors are given by

$$
\begin{gathered}
\mathbf{z}_{1}=\mathbf{u}_{1}=[1,0,0,1]^{T} \\
\mathbf{z}_{2}=\mathbf{u}_{5}=[1,1,1,0]^{T} \\
\mathbf{z}_{3}=\mathbf{u}_{2}+\mathbf{u}_{5}=[2,1,2,0]^{T} \\
\mathbf{z}_{4}=\mathbf{u}_{1}+\mathbf{u}_{2}=[2,0,1,1]^{T}
\end{gathered}
$$

Because the signature matrix is built using a 1 -separable matrix, each right node can be resolved if it is connected to at most one defective item.

Iteration 1: we first find the 1-resolvable right nodes. The first and second right nodes are 1-resolvable because $z_{1,1}=z_{2,1}=1$. Using a BCH decoding algorithm, one can find that the defective items connected to the first and second right nodes are item 1 and item 10 , respectively. Next, we remove the contributions of the items 1 and 10 from the unresolved right nodes. The new observation vectors will be as follows,

$$
\begin{aligned}
& \mathbf{z}_{3}=\mathbf{u}_{2}=[1,0,1,0]^{T} \\
& \mathbf{z}_{4}=\mathbf{u}_{2}=[1,0,1,0]^{T}
\end{aligned}
$$

Iteration 2: it can be easily observed that the third and forth right nodes are 1-resolvable since $z_{3,1}=z_{4,1}=1$. Using a BCH decoding algorithm, it follows that the item 4 is the defective item connected to both right nodes 3 and 4 . Since all the $K=3$ defective items are identified, the decoding algorithm terminates.

### 3.4 Main Results

In this section, we present our main results. Theorem 6 characterizes the required number of tests that guarantees the identification of all defective items with probability approaching one as $K, N \rightarrow \infty$. Theorem 7 presents the computational complexity of the proposed algorithm. The proofs of Theorems 6 and 7 are given in Section 3.5.

Theorem 6. For the sub-linear regime, the proposed scheme recovers all defective items with probability approaching one (as $K, N \rightarrow \infty$ ) with at most

$$
m=c(t) K\left(t \log _{2}\left(\frac{\ell N}{c(t) K}+1\right)+1\right)+1
$$

tests, where $c(t)$ depends only on $t$. Table 3.1 shows the values of $c(t)$ for $t \leq 8$.

| $t$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $c(t)$ | 1.222 | 0.597 | 0.388 | 0.294 | 0.239 | 0.202 | 0.176 | 0.156 |
| $\ell^{\star}$ | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |

Table 3.1: The function $c(t)$ and the optimal left degree $\ell^{\star}$.

Theorem 7. The encoding and decoding of the proposed algorithm for any $t \leq 4$ have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

### 3.5 Proofs of Main Theorems

### 3.5.1 Proof of Theorem 6

Let $N$ be the total number of items, out of which $K$ items are defective. Note that in the QGT problem, performing one initial test (on all items) would suffice to obtain the number of defective items. As mentioned in Section 3.3.3, our scheme employs an
iterative decoding algorithm. In each iteration, the algorithm finds and resolves all the $t$ resolvable right nodes. At the end of each iteration, the decoder subtracts the contribution of the identified defective items from the unresolved right nodes. This process is repeated until there is no t-resolvable right nodes left in the graph. The fraction of defective items that remain unidentified when the decoding algorithm terminates can be analyzed using density evolution as follows.

Assuming that the exact number of the defective items, $K$, is known and the values assigned to the defective and non-defective items are one and zero, respectively, the left-and-right-regular bipartite graph can be pruned. All the zero left nodes and their respective edges are removed from the graph. The number of left nodes in the pruned graph is $K$, but the degree of these nodes remains unchanged. On the other hand, the number of right nodes remains unchanged, but the resulting graph is not right-regular any longer.

Let $\lambda$ be the average right degree, i.e., $\lambda=\frac{K \ell}{M}$. Let $\rho(x) \triangleq \sum_{i=1}^{\min (K, r)} \rho_{i} x^{i-1}$ be the right edge degree distribution, where $\rho_{i}$ is the probability that a randomly picked edge in the pruned graph is connected to a right node of degree $i$, and $\min (K, r)$ is the maximum degree of a right node. As shown in [48], as $K, N \rightarrow \infty$, we have $\rho_{i}=e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}$. The following lemma is useful for computing the fraction of unidentified defective items at each iteration $j$ of the decoding algorithm.

Lemma 8. Let $p_{j}$ be the probability that a randomly chosen defective item is not recovered at iteration $j$ of the decoding algorithm; and let $q_{j}$ be the probability that a randomly picked right node is resolved at iteration $j$ of the decoding algorithm. The relation between $p_{j}$ and $p_{j+1}$ is determined by the following density evolution equations:

$$
\begin{equation*}
q_{j}=\sum_{i=1}^{t} \rho_{i}+\sum_{i=t+1}^{\min (K, r)} \rho_{i} \sum_{k=0}^{t-1}\binom{i-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{i-k-1} \tag{3.4}
\end{equation*}
$$



Figure 3.1: Tree-like representation of neighborhood of the edge between a left node $v$ and a right node $c$ in the pruned graph.

$$
\begin{equation*}
p_{j+1}=\left(1-q_{j}\right)^{\ell-1} \tag{3.5}
\end{equation*}
$$

where $t$ is the level of separability, and $\rho_{i}$ is the probability that a randomly picked edge in the pruned graph is connected to a right node of degree $i$.

Proof. As mentioned earlier, the pruned graph is left-regular and the degree of the left nodes is $\ell$, but the pruned graph is not right-regular any longer and the degree of the right nodes can be any integer in $\{0,1, \cdots, \min (K, r)\}$. A tree-like representation of the neighborhood of an edge between a left node $v$ of degree $\ell$ and a right node $c$ of degree $i$ is shown in Fig. 3.1. The left node $v$ sends a "not identified" message to the right node $c$ at iteration $j+1$ with probability $p_{j+1}$ if all of its neighboring nodes $\left\{c_{i}\right\}_{i=1}^{\ell-1}$ have not been resolved at iteration $j$ which it happens with probability $\left(1-q_{j}\right)^{l-1}$. The right node $c$ of degree $i$ with probability $q_{j}$ passes a "resolved" message to the left node $v$ at iteration $j$ if the number of defective items connected to node $c$, i.e., $i$, is equal to $t$ or less which it happens with probability $\sum_{i=1}^{t} \rho_{i}$, or if the number of defective items connected to node $c$ is more than $t(i>t)$, but only $k \in\{0,1, \cdots, t-1\}$ of the $i-1$ defective items connected to node $c$ other than $v$ are unidentified (we know that $v$ is not identified yet) which this case happens with probability $\sum_{i=t+1}^{\min (K, r)} \rho_{i} \sum_{k=0}^{t-1}\binom{i-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{i-k-1}$.

Note that $p_{j}$ is only a function of the variables $t, \ell$, and $\lambda$ when $\min (K, r) \rightarrow \infty$. Recall that the goal is to minimize the total number of tests, i.e., $M \times s$, where $M$ is the number of right nodes, and $s$ is the number of rows in the signature matrix. The number of rows, $s$, in the signature matrix depends only on the level of separability, $t$. For a given $t$, we can minimize the number of right nodes $M=\frac{\ell}{\lambda} K$ subject to the constraint $\lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=0$, so as to minimize the total number of the tests. The constraint $\lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=0$ guarantees that running the decoding algorithm for sufficiently large number of iterations, the probability that a randomly chosen defective item remains unidentified approaches zero. For any $\ell \geq 2$, let $\lambda_{T}(\ell) \triangleq \sup \left\{\lambda: \lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=0\right\}$. Then, for any $\ell \geq 2$ and $\lambda<\lambda_{T}(\ell)$, we have $\lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=0$. Accordingly, for any $\ell \geq 2$ and $M=\frac{\ell}{\lambda} K>\frac{\ell}{\lambda_{T}(\ell)} K$, it follows that $\lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=0$. Our goal is then to compute

$$
\begin{equation*}
\min _{\ell \in\{2,3, \ldots\}} \frac{\ell}{\lambda_{T}(\ell)} K \tag{3.6}
\end{equation*}
$$

We can solve this problem numerically and attain the optimal value of $\ell$, i.e., $\ell^{\star}$. Let $c(t) \triangleq \frac{\ell^{\star}}{\lambda_{T}\left(\ell^{\star}\right)}$. The number of right nodes can then be chosen as $M=c(t) K \beta$ for any $\beta>1$ to guarantee that $M>c(t) K=\frac{\ell^{\star}}{\lambda_{T}\left(\ell^{\star}\right)} K$. Substituting $M=c(t) K \beta$ in (3.2) results in $r=\frac{\ell N}{c(t) K \beta}$. Therefore, the total number of tests will become

$$
M \times s=c(t) K \beta\left(t \log _{2}\left(\frac{\ell N}{c(t) K \beta}+1\right)+1\right)
$$

Lemma 9. There exist some $\beta>1$ such that

$$
c(t) K\left(t \log _{2}\left(\frac{\ell N}{c(t) K}+1\right)+1\right)+1 \geq c(t) K \beta\left(t \log _{2}\left(\frac{\ell N}{c(t) K \beta}+1\right)+1\right)
$$

Proof. Let us define the following function,

$$
f(\beta) \triangleq c(t) K\left(t \log \left(\frac{\ell N}{c(t) K \beta}+1\right)+1\right)
$$

We need to show that there exists some $\beta>1$ such that $f(1)+1 \geq \beta f(\beta)$, or equivalently, $\beta f(\beta)-f(1) \leq 1$. Since $f(\beta)$ is a monotone decreasing function of $\beta, f(\beta)<f(1)$ for $\beta>1$. This inequality leads to $\beta f(\beta)-f(1)<(\beta-1) f(1)$. Hence, to guarantee that there exists some $\beta>1$ such that $\beta f(\beta)-f(1) \leq 1$, it suffices to show that $(\beta-1) f(1) \leq$ 1 for some $\beta>1$. It is easy to see that $1<\beta \leq \frac{1}{f(1)}+1$ is the satisfactory range.

By combining the result of Lemma 9 and the preceding arguments, it follows that with probability approaching one as $K, N \rightarrow \infty, m=c(t) K\left(t \log _{2}\left(\frac{\ell N}{c(t) K}+1\right)+1\right)+1$ tests would suffice for the proposed algorithm to recover all defective items. This completes the proof.

### 3.5.2 Proof of Theorem 7

Lemma 10. For any $t \leq 4$, the computational complexity of resolving each $t$-resolvable right node is $\mathcal{O}(\log r)$.

Proof. As mentioned in Lemma 7, the block vector $\mathbf{z}_{i}^{(2)}$ under modulo 2 can be interpreted as the syndrome corresponding to an error pattern of Hamming weight $j \leq t$. The location of the $j$ errors ( $j$ defective items) can be determined from $\mathbf{z}_{i}^{(2)}$ under modulo 2 by first using a Berlekamp-Massey algorithm for finding the error locator polynomial. This step involves a time complexity of $\mathcal{O}\left(t^{2} \log r\right)$ (all computations are performed in a finite field of size $2^{m}=r+1$ ). Once the error locator polynomial is determined, the roots of the error locator polynomial have to be found. A standard Chien search can be used to solve this step with complexity $\mathcal{O}(\operatorname{tr} \log r)$; however, when $t \leq 4$, the Chien search can be avoided and the roots can be found directly using the algorithm in [49] with a complexity that is
only $\mathcal{O}(t \log r)$. Therefore, for $t \leq 4$, the decoding complexity of resolving a $t$-resolvable right node is only logarithmic in $r$ (i.e., $\mathcal{O}(\log r)$ ).

The total number of right nodes, $M$, is $\mathcal{O}(K)$. From Lemma 10, it then follows that the complexity of the decoding algorithm is $\mathcal{O}(K \log r)$. Using (3.2), it is easy to see that for any $t \leq 4$ the decoding algorithm has complexity $\mathcal{O}\left(K \log \frac{N}{K}\right)$. The total number of measurements is $m$ and for each measurement $r$ summations are performed. Hence, the complexity of the encoding algorithm is $\mathcal{O}(m r)$, which becomes equivalent to $\mathcal{O}\left(N \log \frac{N}{K}\right)$ for any $t \leq 4$.

### 3.6 Evaluation of $c(t)$

In this section, we present the complete analysis for the case of $t=1$, and show how one can evaluate $c(t)$ at $t=1$, i.e., $c(1)$. The same procedure can be used for evaluating $c(t)$ at any $t>1$. To compute $c(1)=\frac{\ell^{\star}}{\lambda_{T}\left(\ell^{\star}\right)}$, we compute the ratio $\frac{\ell}{\lambda_{T}(\ell)}$ for each $\ell \geq 2$ and its corresponding $\lambda_{T}(\ell)$. The optimal $\ell$, i.e., $\ell^{\star}$, is the one that yields the minimum value for $\frac{\ell}{\lambda_{T}(\ell)}$. For the case of $t=1$, the density evolution equations (3.4) and (3.5) can be combined as

$$
\begin{equation*}
p_{j+1}=\left(1-\sum_{i=1}^{\min (K, r)} \rho_{i}\left(1-p_{j}\right)^{i-1}\right)^{\ell-1} \tag{3.7}
\end{equation*}
$$

Obviously, $p_{1}=1$. Substituting $\rho_{i}=e^{-\lambda} \frac{\lambda^{i-1}}{(i-1)!}$, we can rewrite (3.7) as

$$
\begin{equation*}
p_{j+1}=\left(1-e^{-\lambda} \sum_{i=1}^{\min (K, r)} \frac{\lambda^{i-1}}{(i-1)!}\left(1-p_{j}\right)^{i-1}\right)^{\ell-1} \tag{3.8}
\end{equation*}
$$

For the sub-linear regime, $\frac{K}{N} \rightarrow 0$ (by definition) as $K, N \rightarrow \infty$, and hence, $r \rightarrow \infty$ (by (3.2)). Thus, in the asymptotic regime of our interest, $\min (K, r) \rightarrow \infty$. Letting
$\min (K, r) \rightarrow \infty$, the equation (3.8) reduces to

$$
\begin{equation*}
p_{j+1}=\left(1-e^{-\lambda p_{j}}\right)^{\ell-1} \tag{3.9}
\end{equation*}
$$

Using (3.9), we can write

$$
\lambda=\left(\frac{\ln \left(1-p_{j+1}^{\frac{1}{\ell-1}}\right)}{-p_{j}}\right)
$$

The following two lemmas are useful for computing $\lambda_{T}(\ell)=\sup \left\{\lambda: \lim _{j \rightarrow \infty} p_{j}(\ell, \lambda)=\right.$ $0\}$ for each $\ell \geq 2$.

Lemma 11. For any $\ell \geq 2$ and any $\lambda>0$, the infinite sequence $\left\{p_{1}, p_{2}, \cdots\right\}$ converges.

Proof. Note that every bounded and monotonic sequence converges. From the definition, it is obvious that $0 \leq p_{j} \leq 1$ for any integer $\ell \geq 2$ and any real number $\lambda>0$. Then, it suffices to show the monotonicity of the sequence $\left\{p_{1}, p_{2}, \ldots\right\}$. The proof is based on induction. It is easy to see that $p_{2}<p_{1}$, i.e., $\left(1-e^{-\lambda}\right)^{\ell-1}<1$. The induction hypothesis is that $p_{j}<p_{j-1}$. We need to show that $p_{j+1}<p_{j}$. By the induction hypothesis, we have

$$
\left(1-e^{-\lambda p_{j-1}}\right)^{\ell-1}<p_{j-1}
$$

Then, it is easy to see that

$$
1-e^{-\lambda\left(1-e^{-\lambda p_{j-1}}\right)^{\ell-1}}<1-e^{-\lambda p_{j-1}}
$$

or equivalently,

$$
\begin{equation*}
\left(1-e^{-\lambda\left(1-e^{-\lambda p_{j-1}}\right)^{\ell-1}}\right)^{\ell-1}<\left(1-e^{-\lambda p_{j-1}}\right)^{\ell-1} \tag{3.10}
\end{equation*}
$$

Replacing $\left(1-e^{-\lambda p_{j-1}}\right)^{\ell-1}$ by $p_{j}$, we can rewrite (3.10) as

$$
\left(1-e^{-\lambda p_{j}}\right)^{\ell-1}<p_{j}
$$

which yields $p_{j+1}<p_{j}$, as was to be shown.

Lemma 12. Let $p^{*}$ be the limit of the sequence $\left\{p_{1}, p_{2}, \cdots\right\}$, and let

$$
\lambda_{T}(\ell) \triangleq \inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)
$$

Then, for any $\ell \geq 2$, we have

$$
\begin{cases}p^{*}=0, & 0<\lambda<\lambda_{T}(\ell) \\ p^{*}>0, & \lambda \geq \lambda_{T}(\ell)\end{cases}
$$

Proof. By Lemma 11, we know that $p^{*}$ exists, and it must be a solution to the following equation,

$$
\begin{equation*}
p^{*}=\left(1-e^{-\lambda p^{*}}\right)^{\ell-1} \tag{3.11}
\end{equation*}
$$

We first show that for $0<\lambda<\lambda_{T}(\ell)$, it holds that $p^{*}=0$. It suffices to show that for $0<\lambda<\lambda_{T}(\ell)$ and any integer $\ell \geq 2$, the only solution of (3.11) is $p^{*}=0$. Obviously, $p^{*}=0$ is a solution of (3.11) for any $0<\lambda<\lambda_{T}(\ell)$ and any integer $\ell \geq 2$. Thus, we need to show that for $0<\lambda<\lambda_{T}(\ell)$ and any integer $\ell \geq 2$, and any $0<\epsilon<$ 1 , we have $\epsilon \neq\left(1-e^{-\lambda \epsilon}\right)^{\ell-1}$. The proof is by the way of contradiction. Suppose that $\epsilon=\left(1-e^{-\lambda \epsilon}\right)^{\ell-1}$ for some $0<\epsilon<1$. By solving this equation for $\lambda$, we get

$$
\lambda=\frac{\ln \left(1-\epsilon^{\frac{1}{\ell-1}}\right)}{-\epsilon} .
$$

On the other hand, we know that

$$
\lambda<\lambda_{T}(\ell)=\inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right) .
$$

Thus, we have

$$
\frac{\ln \left(1-\epsilon^{\frac{1}{\ell-1}}\right)}{-\epsilon}<\inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)
$$

for some $0<\epsilon<1$. Obviously, this inequality cannot hold, and we reach a contradiction, as desired.

Next, we shall show that for any $\lambda \geq \lambda_{T}(\ell)$, we have $p^{*}>0$. From (3.11), it follows that

$$
\lambda=\frac{\ln \left(1-p^{*} \frac{1}{\ell-1}\right)}{-p^{*}} .
$$

Hence, $\lambda \geq \lambda_{T}(\ell)$ implies that

$$
\frac{\ln \left(1-p^{* \frac{1}{\ell-1}}\right)}{-p^{*}} \geq \inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)
$$

Again, the proof is by the way of contradiction. Suppose that $p^{*}=0$, i.e., the sequence $\left\{p_{1}, p_{2}, \ldots\right\}$ converges to 0 . Therefore, for any $\delta>0$, there exist a positive integer $i$ such that for any $j \geq i,\left|p^{*}-p_{j}\right|=p_{j}<\delta$. Consider an arbitrary $0<\delta<1$. Let $i$ be such that $p_{i-1} \geq \delta$ and $p_{j}<\delta$ for all $j \geq i$. Note that $p_{i}<\delta$ implies that $\left(1-e^{-\lambda p_{i-1}}\right)^{\ell-1}<\delta$. This inequality can be rewritten as $\lambda<\frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-p_{i-1}}$. Using the facts that $\lambda \geq \lambda_{T}(\ell)$ and $p_{i-1} \geq \delta$, we have

$$
\begin{equation*}
\inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)<\frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-p_{i-1}} \tag{3.12}
\end{equation*}
$$

and

$$
\begin{equation*}
\frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-p_{i-1}} \leq \frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-\delta} . \tag{3.13}
\end{equation*}
$$



Figure 3.2: The number of required tests $(m)$ to identify all $K$ defective items with probability approaching one (for different values of $K$ ) among $N=2^{16}$ items for different values of $t$ obtained via analysis.

Combining (3.12) and (3.13), we get

$$
\begin{equation*}
\inf _{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)<\frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-\delta} \tag{3.14}
\end{equation*}
$$

Let $f(x) \triangleq \frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}$, and let $x^{*}$ be such that

$$
\inf _{0<x<1} f(x)=\frac{\ln \left(1-x^{* \frac{1}{\ell-1}}\right)}{-x^{*}}
$$

Since $\lim _{x \rightarrow 0} f(x)=\lim _{x \rightarrow 1} f(x)=+\infty$, obviously we have $0<x^{*}<1$. Taking $\delta=x^{*}$, we will have

$$
\begin{equation*}
\operatorname{iif}_{0<x<1}\left(\frac{\ln \left(1-x^{\frac{1}{\ell-1}}\right)}{-x}\right)=\frac{\ln \left(1-\delta^{\frac{1}{\ell-1}}\right)}{-\delta} \tag{3.15}
\end{equation*}
$$

From (3.14) and (3.15), we arrive at a contradiction. This completes the proof.


Figure 3.3: The average fraction of unidentified defective items obtained via Monte Carlo simulations for $N=2^{16}$ items among which $K=100$ items are defective.

By the result of Lemma 12, for any $\ell \geq 2$ the value of $\lambda_{T}(\ell)$ can be computed numerically. One can then obtain the optimal value of $\ell$, i.e., $\ell^{\star}$, which minimizes the ratio of $\frac{\ell}{\lambda_{T}(\ell)}$, and accordingly $c(1)=\frac{\ell^{\star}}{\lambda_{T}\left(\ell^{\star}\right)}$ can be computed.

### 3.7 Comparison Results

In this section we will evaluate the performance of the proposed algorithm based on our theoretical analysis and the Monte Carlo simulations.

Based on the results in Theorem 6 and Table 3.1, Fig. 3.2 depicts the total number of tests ( $m$ ) required to identify all the defective items with probability approaching one for different values of $t$. The number of items is assumed to be $N=2^{16}$. As it can be seen, when $t \in\{1,2,3\}$ the required number of tests for identifying all the defective items is less than that for larger values of $t$.

Using the Monte Carlo simulation, we also compare the performance of the proposed
scheme for $t \in\{1,2,3\}$ with the performance of the Multi-Level Group Testing (MLGT) algorithm from [45]. The MLGT scheme is a semi-quantitative group testing scheme where the result of each test is an integer in the set $\{0,1,2, \cdots, L\}$. Letting $L \rightarrow \infty$, the MLGT scheme becomes a QGT scheme. Based on the optimization that we have performed, the optimal left degree for the MLGT scheme is $\ell^{\star}=3$ when $L \rightarrow \infty$. For $K=100$ defective items among a population of $N=2^{16}$ items, the average fraction of unidentified defective items for the MLGT scheme and the proposed scheme are shown in Fig. 3.3 for different values of $m / K$. As it can be observed, the proposed scheme for all the three tested values of $t$ outperforms the MLGT scheme significantly. For instance, when the fraction of unidentified defective items is $2 \times 10^{-4}$, the required number of tests for the MLGT scheme (for $\ell=3$ ) is 3 times, 5 times, and 7 times more than that of the proposed scheme for $t=1, t=2$, and $t=3$, respectively.

## 4. NON-ADAPTIVE QUANTITATIVE GROUP TESTING USING IRREGULAR SPARSE GRAPH CODES*

### 4.1 Introduction

In this chapter, we consider the Quantitative Group Testing (QGT) problem which is concerned with recovering all or a sufficiently large fraction of defective items in a given population of items, each of which is either defective or not. In the QGT problem, the result of a test on any group of items reveals the number of defective items in the tested group. The objective is to design a test plan for QGT with minimum number of tests.

There are two different models for the defective items in the literature: deterministic and randomized. In the deterministic model (a.k.a. the combinatorial model), the exact number of defective items is known, whereas in the randomized model (a.k.a. the probabilistic model), each item is defective with some probability, independent of the other items [11-14]. In this work, we consider the randomized model in which each item is defective with probability $\frac{K}{N}$, independently from the other items, where $N$ is the total number of items, and the parameter $K$ represents the expected number of defective items. It should be noted that the deterministic model can be readily justified using the fact that performing one initial test on all items reveals the number of defective items. Notwithstanding, in most practical applications, performing a test on all items may not be feasible, particularly when the number of items is very large. On the other hand, assuming that the expected number of defective items is known is a more reasonable assumption for many practical applications. Moreover, it should be noted that the QGT schemes designed for the scenarios in which the randomized model is considered are applicable to the scenarios

[^2]considering the deterministic model, but this relation does not work in reverse order.
In this study, we are interested in non-adaptive QGT schemes, where all tests are designed in advance. This is in contrast to adaptive QGT schemes, in which the design of each test depends on the results of the previous tests. In most practical applications, when compared to adaptive QGT schemes, non-adaptive QGT schemes are preferred because all tests can be executed at once in parallel.

### 4.1.1 Related Work and Applications

The QGT problem can be traced back to the seminal work by Shapiro in [35]. To date, several adaptive and non-adaptive QGT strategies have been proposed, see, e.g., [13, 14, $34,43,51]$ and references therein. Using a simple information theoretic argument, one can easily show the information-theoretic lower bound $\log _{K}\binom{N}{K} \approx(K \log (N / K)) / \log K$ on the minimum number of tests for any adaptive QGT scheme.* However, this lower bound is not tight for non-adaptive QGT schemes. In particular, it was shown in [36] and [52] that any non-adaptive QGT scheme requires at least $(2 K \log (N / K)) / \log K$ tests. For the linear regime in which the number of defective items is a constant fraction of the total number of items, the QGT problem has been fully solved [53,54]. However, for the sub-linear regime, i.e., when the number of defective items grow sub-linearly in the total number of items, the QGT problem is widely open. Recently, in [43], the first non-adaptive QGT scheme for the sub-linear regime that requires $m \approx 1.19 \mathrm{~K} \log \left(4.74 \frac{N}{K}\right)$ tests to recover all the defective items with probability approaching 1 was proposed. Shortly after, Gebhard et al. in [51] proposed a greedy non-adaptive QGT scheme that requires $m=\frac{1+\sqrt{\theta}}{1-\sqrt{\theta}} K \ln \left(\frac{N}{K}\right)$ tests to recover all $K=N^{\theta}$ (for $0<\theta<1$ ) defective items with high probability.

Aside from the theoretical endeavors, the QGT problem has also gained substantial attention over the last few years from the practical perspective. In particular, the QGT

[^3]problem has been studied for a wide range of applications from machine learning and computational biology $[55,56]$ to multi-access communication, traffic monitoring, and network tomography [57-59]. It should be noted that most of these applications are being run repeatedly over time, and for such applications, minimizing the constant factor hidden in the order is also of prominent importance. This observation is the primary motivation for this work.

### 4.1.2 Main Contributions

In this work, we propose a non-adaptive QGT scheme for the scenarios in which the randomized model is considered for defective items. The testing algorithm of the proposed scheme relies on sparse graph codes over irregular bipartite graphs with optimized leftdegree profiles as well as binary $t$-error-correcting BCH codes. As part of the process of optimizing the left-degree profile of the graph, we take advantage of the density-evolution technique to analyze the probability of error of the proposed peeling-based recovery algorithm, i.e., the probability that a defective item remains unidentified over the iterations of the recovery algorithm. We provide provable guarantees on the performance of the proposed scheme in terms of the required number of tests. In particular, we show that in the sub-linear regime the proposed scheme requires $m=c(t, d) K\left(t \log \left(\frac{\ell N}{c(t, d) K}+1\right)+1\right)$ tests to identify all defective items with high probability, where $d$ and $\ell$ are the maximum and average left degree, respectively, and $c(t, d)$ is constant with respect to $K$ and $N$, and depends only on $t$ and $d$. Moreover, we show that, for any $t \leq 4$, the testing and recovery algorithms of the proposed scheme have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

### 4.2 Problem Setup and Notations

We denote vectors and matrices by bold-face small and capital letters, respectively. For an integer $i \geq 1$, we denote $\{1, \ldots, i\}$ by $[i]$.

We define the support vector $\mathbf{x} \in\{0,1\}^{N}$ to represent the set of $N$ items. The $i$ th component of $\mathbf{x}$ is 1 if and only if the $i$-th item is defective. In a non-adaptive QGT problem, designing a test scheme consisting of $m$ tests is equivalent to the construction of a binary matrix with $m$ rows which is referred to as measurement matrix. We let matrix $\mathbf{A} \in\{0,1\}^{m \times N}$ denote the measurement matrix wherein the non-zero indices in the $i$ th row correspond to the items that are present in the $i$-th test. We also let vector $\mathbf{y} \in$ $\{0,1,2, \ldots\}^{m}$ denote the outcomes of the $m$ tests in the following matrix form.

$$
\begin{equation*}
\mathbf{y}=\left[y_{1}, \ldots, y_{m}\right]^{T}=\mathbf{A} \mathbf{x} \tag{4.1}
\end{equation*}
$$

The objective is to construct a measurement matrix with a small number of rows (tests) that successfully identifies the set of defective items with high probability given the test results vector $\mathbf{y}$.

### 4.3 Proposed Algorithm

### 4.3.1 Testing algorithm

We employ a framework similar to that proposed in [43] for designing the measurement matrix $\mathbf{A}$; however, in our design we utilize irregular bipartite graphs with carefully designed left-degree profile, instead of bi-regular bipartite graphs.

Consider a randomly generated bipartite graph with $N$ left nodes and $M$ right nodes where each right node is connected to $r$ left nodes. The left nodes are connected to the right nodes according to a left-node degree distribution given by $L(x) \triangleq \sum_{i=1}^{d} L_{i} x^{i}$ where $d$ and $L_{i}$ denote the maximum degree of a left node and the probability that a randomly selected left node in the graph has degree $i$, respectively. We denote the adjacency matrix of such a graph by $\mathbf{T} \in\{0,1\}^{M \times N}$ where each column in $\mathbf{T}$ corresponds to a left node, and each row in $\mathbf{T}$ corresponds to a right node and has exactly $r$ ones. The adjacency
matrix $\mathbf{T}$ can be represented in the matrix form $\mathbf{T}=\left[\mathbf{t}_{1}^{T}, \mathbf{t}_{2}^{T}, \ldots, \mathbf{t}_{M}^{T}\right]^{T}$, where $\mathbf{t}_{i}$ denotes the $i$-th row.

A carefully designed signature matrix $\mathbf{U} \in\{0,1\}^{s \times r}$ is used to assign $s$ tests to each right node. We place an all-ones row of length $r$ as the first row of the signature matrix. The first row in $\mathbf{U}$ corresponds to a test whose result reveals the number of defective items connected to a right node. The rest of the rows in U are the rows in the parity-check matrix of a binary $t$-error-correcting BCH code [47]. Given that the number of defective items connected to a right node is no more than $t$, the results of the tests corresponding to the rows in the parity-check matrix can be used to identify the defective items connected to the right node. Considering that the number of columns is $r$, the number of rows in the parity-check matrix of a $t$-error-correcting BCH code is given by $R=t \log (r+1)$. The signature matrix $\mathbf{U}$ can then be represented by $\mathbf{U}=\left[\mathbf{1}_{1 \times r}^{T}, \mathbf{H}_{t}^{T}\right]^{T}$, where $\mathbf{1}_{1 \times r}$ is an all-ones row of length $r$, and $\mathbf{H}_{t} \in\{0,1\}^{R \times r}$ is the parity-check matrix of a binary $t$ -error-correcting BCH code. One can readily observe that the number of rows in U is given by $s=R+1=t \log (r+1)+1$.

Now, we show the construction process of the measurement matrix using the adjacency matrix $\mathbf{T}$ and the signature matrix $\mathbf{U}$. Let the measurement matrix be given by $\mathbf{A}=\left[\mathbf{A}_{1}^{T}, \ldots, \mathbf{A}_{M}^{T}\right]^{T}$ where $\mathbf{A}_{i} \in\{0,1\}^{s \times N}$ is a block matrix that represents the $s$ tests at the $i$-th right node. Let $\mathbf{u}_{j}$ denote the $j$-th column of the signature matrix. Note that the number of columns in the signature matrix $\mathbf{U}$ is $r$, and there are exactly $r$ ones in each row of the adjacency matrix $\mathbf{T}$. The block matrix $\mathbf{A}_{i}$ is then constructed by replacing zeros and ones in the $i$-th row of the adjacency matrix, $\mathbf{t}_{i}$, by all-zero columns and the columns of the signature matrix, respectively, as follows:

$$
\begin{equation*}
\mathbf{A}_{i}=\left[\mathbf{0}, \ldots, \mathbf{0}, \mathbf{u}_{1}, \mathbf{0}, \ldots, \mathbf{u}_{2}, \mathbf{0}, \ldots, \mathbf{u}_{r}\right] \tag{4.2}
\end{equation*}
$$

where $\mathbf{t}_{i}=[0, \ldots, 0,1,0, \ldots, 1,0, \ldots, 1]$. In other words, we place the $r$ columns of the signature matrix at the coordinates of the $r$ ones in the row $\mathbf{t}_{i}$, and then we replace zeros in $\mathrm{t}_{i}$ by all-zero columns. The total number of rows in the measurement matrix A which is equivalent to the total number of tests in the proposed scheme is given by $m=M \times s=M(t \log (r+1)+1)$. The following example helps to better understand the construction process of the measurement matrix.

Example 5. Let T denote the adjacency matrix of an irregular bipartite graph with $N=14$ left nodes and $M=3$ right nodes of degree $r=7$. The edge connections of the left side satisfies the following left node degree distribution given by $L(x)=\frac{10}{14} x+\frac{1}{14} x^{2}+\frac{3}{14} x^{3}$.

$$
\mathbf{T}=\left[\begin{array}{llllllllllllll}
0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0
\end{array}\right]
$$

Also, we let $\mathbf{H}_{1}$ and $\mathbf{U}=\left[\mathbf{1}_{1 \times 7}^{T}, \mathbf{H}_{1}^{T}\right]^{T}$ denote the parity-check matrix of a binary $t=1$ -error-correcting BCH code of length $r=7$ and the signature matrix, respectively, where

$$
\mathbf{H}_{1}=\left[\mathbf{h}_{1}, \ldots, \mathbf{h}_{7}\right]=\left[\begin{array}{lllllll}
0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}\right]
$$

and

$$
\mathbf{U}=\left[\begin{array}{lllllll}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 1
\end{array}\right]
$$

The measurement matrix A can then be constructed by following the procedure explained earlier,

$$
\mathbf{A}=\left[\begin{array}{llllllllllllll}
0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
\hline 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0
\end{array}\right] .
$$

### 4.3.2 Recovery Algorithm

The recovery algorithm is similar to the peeling decoding algorithm, and it proceeds in an iterative manner as follows. During each iteration, the recovery algorithm inspects all the right nodes, and identifies and resolves any right node which is connected to $t$ or less number of defective items (for more details, see the proof of [43, Lemma 1]). Then, the recovery algorithm peels the edges connected to the identified defective items off the graph, and the next iteration begins. When no (not-yet-resolved) right node connected to $t$ or less number of defective items can be found, the recovery algorithm terminates. Below, we provide an illustrative example of the recovery algorithm.

Example 6. Consider the scenario in Example 5. Suppose that items 4,8, and 11 are defective. Let the support vector $\mathbf{x}=[0,0,0,1,0,0,0,1,0,0,1,0,0,0]^{T}$ represent the set of $N=14$ items. The test results vector $\mathbf{y}$ according to the testing algorithm using the measurement matrix A constructed in Example 5 can be expressed as follows:

$$
\mathbf{y}=\left[y_{1}, \cdots, y_{12}\right]^{T}=\mathbf{A} \mathbf{x}=\left[\begin{array}{c}
\mathbf{u}_{2} \\
\mathbf{u}_{2}+\mathbf{u}_{4} \\
\mathbf{u}_{2}+\mathbf{u}_{4}+\mathbf{u}_{6}
\end{array}\right] .
$$

The results of the tests corresponding to the right nodes $1,2,3$ are respectively given by

$$
\begin{gathered}
{\left[y_{1}, y_{2}, y_{3}, y_{4}\right]^{T}=\mathbf{u}_{2}=[1,0,1,0]^{T},} \\
{\left[y_{5}, y_{6}, y_{7}, y_{8}\right]^{T}=\mathbf{u}_{2}+\mathbf{u}_{4}=[2,0,2,1]^{T},} \\
{\left[y_{9}, y_{10}, y_{11}, y_{12}\right]^{T}=\mathbf{u}_{2}+\mathbf{u}_{4}+\mathbf{u}_{6}=[3,1,3,2]^{T} .}
\end{gathered}
$$

Since we used the parity-check matrix of a $t=1$-error-correcting BCH code to build the signature matrix, each right node can be resolved (i.e., all items connected to the right node can be identified) if it is connected to at most one defective item. The first test result associated to a right node shows the number of defective items connected to that right node.

In the first iteration, the decoding algorithm can only resolve the first right node because $y_{1}=1$ and $y_{5}, y_{9} \neq 1$. Using $\left[y_{2}, y_{3}, y_{4}\right]^{T}=\mathbf{h}_{2}=[0,1,0]^{T}$, by using a BCH decoding algorithm we can identify the second item connected to the first right node, i.e., item 4, as a defective item. Subtracting off the contribution of the item 4 from the test results corresponding to the unresolved right nodes, the updated test results will be as
follows:

$$
\begin{gathered}
{\left[y_{5}, y_{6}, y_{7}, y_{8}\right]^{T}=\mathbf{u}_{4}=[1,0,1,1]^{T}} \\
{\left[y_{9}, y_{10}, y_{11}, y_{12}\right]^{T}=\mathbf{u}_{4}+\mathbf{u}_{6}=[2,1,2,2]^{T}}
\end{gathered}
$$

In the second iteration, the recovery algorithm resolves the second right node because $y_{5}=1$ and $y_{9} \neq 1$. A BCH decoding algorithm uses $\left[y_{6}, y_{7}, y_{8}\right]^{T}=\mathbf{h}_{4}=[0,1,1]^{T}$, and declares the forth item connected to the second right node, i.e., item 8 , as a defective item. Similarly as in the case of item 4 in the first iteration, subtracting off the contribution of the item 8 from the test results corresponding to the unresolved right nodes, the updated test results will be as follows:

$$
\left[y_{9}, y_{10}, y_{11}, y_{12}\right]^{T}=\mathbf{u}_{6}=[1,1,1,1]^{T}
$$

Since $y_{9}=1$, the recovery algorithm is then able to resolve the third right node in the third iteration. Looking at $\left[y_{10}, y_{11}, y_{12}\right]^{T}=\mathbf{h}_{6}=[1,1,1]^{T}$, by using a BCH decoding algorithm we can identify the sixth item connected to the third right node, i.e., item 11 , as a defective item. Since all 3 right nodes are resolved, the recovery algorithm cannot find any not-yet-resolved right node (connected to 1 or less defective items), and hence the recovery algorithm terminates. For this example, the recovery algorithm successfully identified all 3 defective items.

### 4.4 Main Results

We present our main results in this section. Theorem 8 specifies the number of tests required by the proposed QGT scheme in the sub-linear regime. Theorem 9 states the computational complexity of the testing and recovery algorithms of the proposed QGT scheme. The proofs of Theorems 8 and 9 are given in Section 4.5.

Theorem 8. In the sub-linear regime, the proposed $Q G T$ scheme requires $m=c(t, d) K\left(t \log \left(\frac{\ell N}{c(t, d) K}+1\right)+1\right)$ tests to identify all defective items with probability approaching 1, where $d$ and $\ell$ are the maximum and average left degree, respectively; and $c(t, d)$ is constant in $N$ and $K$, and depends only on tand $d$. Table 4.1 shows the values of $c(t, d)$ for $t=1$ and $d \in\{3,4, \cdots, 18\}$, and Table 4.2 (or respectively, Table 4.3) shows the values of $c(t, d)$ for $t=2$ (or respectively, $t=3$ ) and $d \in\{2,3, \cdots, 17\}$.

Table 4.1: The constant $c(t, d)$ for $t=1$ and $d \in\{3,4, \cdots, 18\}$.

| d | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\lambda_{3}$ | 1 | 0.785 | 0.765 | 0.746 | 0.723 | 0.705 | 0.69 | 0.676 | 0.658 | 0.646 | 0.634 | 0.621 | 0.611 | 0.595 | 0.579 | 0.564 |
| $\lambda_{4}$ |  | 0.215 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{5}$ |  |  | 0.235 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{6}$ |  |  |  | 0.254 |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{7}$ |  |  |  |  | 0.277 |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{8}$ |  |  |  |  |  | 0.295 |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{9}$ |  |  |  |  |  |  | 0.31 |  |  |  |  |  |  |  |  |  |
| $\lambda_{10}$ |  |  |  |  |  |  |  | 0.324 |  |  |  |  |  |  |  |  |
| $\lambda_{11}$ |  |  |  |  |  |  |  |  | 0.342 |  |  |  |  |  |  |  |
| $\lambda_{12}$ |  |  |  |  |  |  |  |  |  | 0.354 |  |  |  |  |  |  |
| $\lambda_{13}$ |  |  |  |  |  |  |  |  |  |  | 0.366 |  |  |  |  |  |
| $\lambda_{14}$ |  |  |  |  |  |  |  |  |  |  |  | 0.379 |  |  |  |  |
| $\lambda_{15}$ |  |  |  |  |  |  |  |  |  |  |  |  | 0.389 |  |  |  |
| $\lambda_{16}$ |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.405 | 0.005 |  |
| $\lambda_{17}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.416 | 0.003 |
| $\lambda_{18}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.433 |
| $\ell$ | 3 | 3.17 | 3.312 | 3.437 | 3.563 | 3.678 | 3.783 | 3.88 | 3.993 | 4.084 | 4.177 | 4.273 | 4.356 | 4.473 | 4.592 | 4.709 |
| $c(t, d)$ | 1.222 | 1.217 | 1.208 | 1.197 | 1.186 | 1.175 | 1.164 | 1.153 | 1.142 | 1.133 | 1.123 | 1.114 | 1.106 | 1.098 | 1.093 | 1.09 |

Table 4.2: The constant $c(t, d)$ for $t=2$ and $d \in\{2,3, \cdots, 17\}$.

| d | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\lambda_{2}$ | 1 | 0.659 | 0.69 | 0.681 | 0.666 | 0.653 | 0.639 | 0.619 | 0.592 | 0.57 | 0.56 | 0.554 | 0.549 | 0.546 | 0.541 | 0.536 |
| $\lambda_{3}$ |  | 0.341 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{4}$ |  |  | 0.31 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{5}$ |  |  |  | 0.319 |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{6}$ |  |  |  |  | 0.334 |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{7}$ |  |  |  |  |  | 0.347 |  |  |  | 0.001 | 0.049 | 0.09 | 0.059 | 0.022 | 0.001 |  |
| $\lambda_{8}$ |  |  |  |  |  |  | 0.361 |  |  |  |  | 0.004 | 0.074 | 0.144 | 0.187 | 0.199 |
| $\lambda_{9}$ |  |  |  |  |  |  |  | 0.381 | 0.002 |  |  |  |  |  |  |  |
| $\lambda_{10}$ |  |  |  |  |  |  |  |  | 0.406 |  |  |  |  |  |  |  |
| $\lambda_{11}$ |  |  |  |  |  |  |  |  |  | 0.429 |  |  |  |  |  |  |
| $\lambda_{12}$ |  |  |  |  |  |  |  |  |  |  | 0.391 |  |  |  |  |  |
| $\lambda_{13}$ |  |  |  |  |  |  |  |  |  |  |  | 0.352 |  |  |  |  |
| $\lambda_{14}$ |  |  |  |  |  |  |  |  |  |  |  |  | 0.317 |  |  |  |
| $\lambda_{15}$ |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.288 |  |  |
| $\lambda_{16}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.271 |  |
| $\lambda_{17}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.265 |
| $\ell$ | 2 | 2.257 | 2.367 | 2.474 | 2.573 | 2.659 | 2.741 | 2.843 | 2.969 | 3.085 | 3.126 | 3.15 | 3.174 | 3.193 | 3.214 | 3.242 |
| $c(t, d)$ | 0.597 | 0.582 | 0.572 | 0.562 | 0.553 | 0.545 | 0.538 | 0.531 | 0.528 | 0.527 | 0.526 | 0.526 | 0.526 | 0.525 | 0.525 | 0.525 |

Table 4.3: The constant $c(t, d)$ for $t=3$ and $d \in\{2,3, \cdots, 17\}$.

| d | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $\lambda_{2}$ | 1 | 0.97 | 0.889 | 0.844 | 0.807 | 0.784 | 0.759 | 0.737 | 0.72 | 0.704 | 0.686 | 0.668 | 0.653 | 0.639 | 0.632 | 0.63 |
| $\lambda_{3}$ |  | 0.03 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{4}$ |  |  | 0.111 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{5}$ |  |  |  | 0.156 |  |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{6}$ |  |  |  |  | 0.193 |  |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{7}$ |  |  |  |  |  | 0.216 |  |  |  |  |  |  |  |  |  |  |
| $\lambda_{8}$ |  |  |  |  |  |  | 0.241 |  |  |  |  |  |  |  |  |  |
| $\lambda_{9}$ |  |  |  |  |  |  |  | 0.263 |  |  |  |  |  |  |  |  |
| $\lambda_{10}$ |  |  |  |  |  |  |  |  | 0.28 |  |  |  |  |  |  |  |
| $\lambda_{11}$ |  |  |  |  |  |  |  |  |  | 0.296 |  |  |  |  |  |  |
| $\lambda_{12}$ |  |  |  |  |  |  |  |  |  |  | 0.314 |  |  |  |  |  |
| $\lambda_{13}$ |  |  |  |  |  |  |  |  |  |  |  | 0.332 | 0.001 |  | 0.045 | 0.11 |
| $\lambda_{14}$ |  |  |  |  |  |  |  |  |  |  |  |  | 0.346 |  |  |  |
| $\lambda_{15}$ |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.361 |  |  |
| $\lambda_{16}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.323 |  |
| $\lambda_{17}$ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0.26 |
| $\ell$ | 2 | 2.021 | 2.118 | 2.207 | 2.295 | 2.366 | 2.442 | 2.515 | 2.577 | 2.639 | 2.709 | 2.781 | 2.848 | 2.909 | 2.945 | 2.952 |
| $c(t, d)$ | 0.388 | 0.388 | 0.387 | 0.384 | 0.381 | 0.378 | 0.375 | 0.372 | 0.37 | 0.367 | 0.365 | 0.363 | 0.363 | 0.362 | 0.362 | 0.362 |

Theorem 9. For any $t \leq 4$, the testing and recovery algorithms of the proposed QGT scheme have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively.

### 4.5 Proof of Main Theorems

### 4.5.1 Proof of Theorem 8

Consider a group of $N$ items where each item is defective with probability $\gamma \triangleq \frac{K}{N}$. Also, consider an irregular bipartite graph with $N$ left nodes and $M$ right nodes where each right node is connected to $r$ left nodes. The left nodes are connected to the right nodes according to a left-node degree distribution given by $L(x)=\sum_{i=1}^{d} L_{i} x^{i}$ where $d$ and $L_{i}$ denote the maximum degree of a left node and the probability that a randomly selected left node in the graph has degree $i$, respectively. The average left degree can be computed by $\ell=\sum_{i=1}^{d} i L_{i}$. Since the number of edges connected to the left nodes is equal to the number of edges connected to the right nodes, the following equation holds:

$$
\begin{equation*}
N \ell=M r \tag{4.3}
\end{equation*}
$$

The left edge degree distribution can be defined by $\lambda(x) \triangleq \sum_{i=1}^{d} \lambda_{i} x^{i-1}=\frac{L^{\prime}(x)}{L^{\prime}(1)}$ where $\lambda_{i}$ denotes the probability that a randomly selected edge in the graph is connected to a left


Figure 4.1: A tree-like representation of the neighborhood of an edge $e$ between a left node $v$ of degree $i$ and a right node $c$ of degree $r$ in the right-regular bipartite graph.
node of degree $i$. It is easy to see that $L^{\prime}(1)=\ell$. Thus, one can readily compute $\lambda_{i}=\frac{i L_{i}}{\ell}$. Using the fact that $\sum_{i=1}^{d} L_{i}=1$, we can rewrite the last equation as follows:

$$
\begin{equation*}
\frac{1}{\ell}=\sum_{i=1}^{d} \frac{\lambda_{i}}{i} \tag{4.4}
\end{equation*}
$$

We leverage the density evolution technique to analyze the fraction of defective items remains unidentified at the end of each iteration of the recovery algorithm.

Lemma 13. Let the probability that a randomly picked item is a defective item and remains unidentified at the end of iteration $j$ of the recovery algorithm be denoted by $p_{j}$. Also, let the probability that a randomly selected right node is resolved at iteration $j$ of the recovery algorithm be denoted by $q_{j}$. The following density evolution equations illustrates the relation between $p_{j}$ and $p_{j+1}$.

$$
\begin{gather*}
q_{j}=\sum_{k=0}^{t-1}\binom{r-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{r-k-1},  \tag{4.5}\\
p_{j+1}=\gamma \sum_{i=1}^{d} \lambda_{i}\left(1-q_{j}\right)^{i-1}, \tag{4.6}
\end{gather*}
$$

where $t, r$, and $d$ are the error correction capability of the BCH code, the degree of right nodes, and the maximum degree of left nodes, respectively.

Proof. A tree-like representation of the neighborhood of an edge $e$ between a left node $v$ of degree $i$ and a right node $c$ of degree $r$ is shown in Fig. 4.1. The left node $v$ sends a "not identified" message to the right node $c$ at iteration $j+1$ through the edge $e$ if none of its other neighboring right nodes $\left\{c_{k}\right\}_{k=1}^{i-1}$ have been resolved at iteration $j$. This event happens with probability $\left(1-q_{j}\right)^{i-1}$. A randomly selected edge is connected to a left node of degree $i$ with probability $\lambda_{i}$. Thus, a randomly selected left node remains unidentified at the end of iteration $j$ with probability $\sum_{i=1}^{d} \lambda_{i}\left(1-q_{j}\right)^{i-1}$. Also, we know that each item is defective with probability $\gamma$. Hence, the probability that a randomly picked item is a defective item and remains unidentified at the end of iteration $j$ of the recovery algorithm is $p_{j+1}=\gamma \sum_{i=1}^{d} \lambda_{i}\left(1-q_{j}\right)^{i-1}$. The right node $c$ passes a "resolved" message to the left node $v$ at iteration $j$ through the edge $e$ if among the other $r-1$ left nodes connected to it only $k \in\{0,1, \cdots, t-1\}$ items are unidentified. This event happens with probability $\sum_{k=0}^{t-1}\binom{r-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{r-k-1}$. A randomly selected edge is connected to a right node of degree $r$ with probability one. Thus, a randomly selected right node is resolved at iteration $j$ of the decoding algorithm with probability $q_{j}=\sum_{k=0}^{t-1}\binom{r-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{r-k-1}$.

The density evolution equations (4.5) and (4.6) can be combined as

$$
\begin{equation*}
p_{j+1}=\gamma \sum_{i=1}^{d} \lambda_{i}\left(1-\sum_{k=0}^{t-1}\binom{r-1}{k} p_{j}^{k}\left(1-p_{j}\right)^{r-k-1}\right)^{i-1} \tag{4.7}
\end{equation*}
$$

Letting $r \rightarrow \infty$ and using the Poisson approximation, the equation (4.7) reduces to

$$
\begin{equation*}
p_{j+1}=\gamma \sum_{i=1}^{d} \lambda_{i}\left(1-\sum_{k=0}^{t-1} \frac{\left(r p_{j}\right)^{k} e^{-r p_{j}}}{k!}\right)^{i-1} . \tag{4.8}
\end{equation*}
$$

Let $\phi_{j} \triangleq \frac{p_{j}}{\gamma}$ and $\psi \triangleq r \gamma$. We can rewrite (4.8) as follows:

$$
\begin{equation*}
\phi_{j+1}=\sum_{i=1}^{d} \lambda_{i}\left(1-\sum_{k=0}^{t-1} \frac{\left(\psi \phi_{j}\right)^{k} e^{-\psi \phi_{j}}}{k!}\right)^{i-1} \tag{4.9}
\end{equation*}
$$

where $\phi_{j}$ denotes the probability that a randomly chosen defective item remains unidentified at the end of iteration $j$ of the recovery algorithm. The objective is to minimize the total number of tests, $m=M \times s$, where $M$ is the number of right nodes and $s$ is the number of rows in signature matrix. Substituting $\gamma=\frac{K}{N}$ in (4.3) results in $M=\frac{\ell}{r \gamma} K$. Using the fact that $\psi=r \gamma$, we can rewrite the number of right nodes as $M=\frac{\ell}{\psi} K$.

For a given $t$ and $d$, we can minimize the number of right nodes, $M=\frac{\ell}{\psi} K$, subject to the constraint $\phi_{j+1}<\phi_{j}$, so as to minimize the total number of the tests. The constraint $\phi_{j+1}<\phi_{j}$ guarantees that $\lim _{j \rightarrow \infty} \phi_{j} \rightarrow 0$. In other words, this constraint guarantees that the probability that a randomly selected defective item remains unidentified after running the recovery algorithm for sufficiently large number of iterations, approaches zero. Note that knowing $N$ and $\gamma$ means that $K$ is also known. Thus, the optimization problem reduces to minimizing the fraction $\frac{\ell}{\psi}$. It should be noted that minimizing the fraction $\frac{\ell}{\psi}$ is equivalent to minimizing the fraction $\frac{-\psi}{\ell}$. Using (4.4), one can readily see that $\frac{-\psi}{\ell}=-\psi \sum_{i=1}^{d} \frac{\lambda_{i}}{i}$. We perform a two-step optimization procedure as follows. First, given the parameters $t$ and $d$, we solve the following Linear Programming (LP) problem for any $\psi>0$.

$$
\begin{array}{ll}
\min _{\substack{\lambda_{i} \\
i \in[d]}} & -\psi \sum_{i=1}^{d} \frac{\lambda_{i}}{i} \\
\text { s.t. } \quad & \sum_{i=1}^{d} \lambda_{i}\left(1-\sum_{k=0}^{t-1} \frac{(\psi \phi)^{k} e^{-\psi \phi}}{k!}\right)^{i-1}<\phi \\
& \sum_{i=1}^{d} \lambda_{i}=1 \\
& \lambda_{i} \geq 0, \forall i \in[d] \tag{4.10d}
\end{array}
$$



Figure 4.2: The number of required tests $(m)$ to identify all defective items (for different values of $K$ ) among $N=2^{32}$ items obtained via analysis.

For any $\psi>0$, let $f(\psi) \triangleq-\psi \sum_{i=1}^{d} \frac{\lambda_{i}^{\star}}{i}$, where $\lambda_{i}^{\star}$ 's denote the optimal value of $\lambda_{i}$ 's attained by solving this LP problem. We then minimize $f(\psi)$ over all values of $\psi>0$ as follows.

$$
\begin{equation*}
\min _{\psi>0} f(\psi) \tag{4.11}
\end{equation*}
$$

We can solve this problem numerically and attain the optimal value of $\psi$ which is denoted by $\psi^{\star}$. Let $c(t, d) \triangleq \frac{-1}{f\left(\psi^{\star}\right)}$. Then, the minimum number of right nodes is given by $M=c(t, d) K$. Substituting $M=c(t, d) K$ in (4.3), one can easily compute $r=\frac{\ell N}{c(t, d) K}$. Therefore, the total number tests will become

$$
m=M \times s=c(t, d) K\left(t \log \left(\frac{\ell N}{c(t, d) K}+1\right)+1\right)
$$



Figure 4.3: The probability of error obtained via Monte Carlo simulations for $N=2^{16}$ items among which $K=100$ items are defective.

### 4.5.2 Proof of Theorem 9

The total number of tests is $m=\mathcal{O}\left(K \log \frac{N}{K}\right)$. For each test, $r$ summations are executed. Thus, the testing algorithm has the computational complexity of $\mathcal{O}\left(r K \log \frac{N}{K}\right)$. From (4.3), one can easily see that $r=\mathcal{O}\left(\frac{N}{K}\right)$. Then, the computational complexity of the testing algorithm can be stated as $\mathcal{O}\left(N \log \frac{N}{K}\right)$.

The total number of right nodes is $M=\mathcal{O}(K)$. The computational complexity of resolving each right node is given by $\mathcal{O}(\log r)$ when $t \leq 4$ (see the proof of [43, Lemma 4]). Therefore, the computational complexity of the recovery algorithm is $\mathcal{O}\left(K \log \frac{N}{K}\right)$.

### 4.6 Comparison Results

In this section, we evaluate the performance of the proposed scheme via extensive simulations.

We compare the performance of the proposed scheme with the performance of two non-adaptive QGT schemes recently proposed in [51] and [43] based on our theoretical analysis. Fig. 4.2 illustrates the total number of tests $(m)$ required to identify all defective items. The total number of items is considered to be $N=2^{32}$. As it can be seen, the proposed scheme, for $t=2$, requires the minimum number of tests to identify all the defective items. Also, it can be observe that the gap between the proposed scheme and the two other schemes increases as the number of defective items $(K)$ grows.

We also compare the performance of the proposed scheme with the performance of non-adaptive QGT schemes in [51] and [43] using the Monte Carlo simulation. The probability of error, defined as the probability of a defective item to remain unidentified, is depicted in Fig. 4.3 for $K=100$ defective items among a population of $N=2^{16}$ items. For a target error probability, e.g., $10^{-5}$, the required number of tests is minimum for the proposed scheme for $t=3$.

## 5. SCHEDULING IMPROVES THE PERFORMANCE OF BELIEF PROPAGATION FOR NOISY GROUP TESTING

### 5.1 Introduction

In this chapter, we consider the noisy Group Testing (GT) problem which is concerned with recovering all defective items in a given population of items. In the group testing problem, the result of a test on any group of items is binary. The objective is to design a test plan for the group testing problem with a minimum number of tests. Aside from the theoretical endeavors, the GT problem has also gained substantial attention from the practical perspective. In particular, the GT problem has been studied for a wide range of applications from biology and medicine [2] to information and communication technology [3, 4], and computer science [5]. Very recently, group testing has also been used for COVID-19 detection [6-9].

There are two different scenarios for the defective items. In the combinatorial model, the exact number of defective items is known, whereas in the probabilistic model, each item is defective with some probability, independent of the other items [43,60,61]. In this work, we consider the combinatorial model. In the combinatorial model, we assume that there are exactly $K$ defective items among a population of $N$ items.

In this study, we are interested in non-adaptive group testing schemes, where all tests are designed in advance. This is in contrast to adaptive schemes, in which the design of each test depends on the results of the previous tests $[14,50,62]$. In most practical applications, when compared to adaptive group testing schemes, non-adaptive schemes are preferred because all tests can be executed at once in parallel. Different decoding algorithms such as linear programming, combinatorial orthogonal matching pursuit, definite defectives, belief propagation (BP), and separate decoding of items have been proposed
for noisy non-adaptive group testing. A thorough review and comparison of these algorithms is provided in [63]. It is difficult to analyze the performance of these algorithms in the non-asymptotic regime. However, empirical evidence suggests that the BP algorithm results in lower error probabilities compared to other algorithms.

BP decoding is an iterative algorithm that passes messages over the edges in the underlying factor graph according to a schedule. For a cycle-free factor graph, BP decoding is equivalent to maximum-likelihood decoding. However, in the presence of loops in the factor graph, BP becomes suboptimal. The most popular scheduling strategy in BP decoding is flooding, or simultaneous scheduling, where in every iteration all the variable nodes are updated simultaneously using the same pre-update information, followed by updating all the test nodes of the graph, again, using the same pre-update information. Several studies have investigated the effects of different types of sequential, or non-simultaneous, scheduling strategies in BP for decoding low-density parity-check (LDPC) codes among which are random scheduling BP and node-wise residual BP (see [64] and references therein). It has been shown that sequential BP algorithms converge faster than traditional BP. Also, sequential updating solves some standard trapping set errors [64, 65]. To the best of our knowledge, these algorithms have not been used in the context of group testing.

### 5.1.1 Main Contributions

In this chapter, we focus on a practical regime in which the number of items is in the order of hundreds, and investigate the performance of two variants of BP algorithm for decoding of noisy non-adaptive group testing under the combinatorial model for defective items. Through extensive simulations, we show that the proposed algorithms achieve higher success probability and lower false-negative and false-positive rates when compared to the traditional BP algorithm.

### 5.2 Problem Setup and Notations

Throughout the chapter, we denote vectors and matrices by bold-face small and capital letters, respectively. For an integer $i \geq 1$, we denote $\{1, \ldots, i\}$ by $[i]$.

In this chapter, we consider a noisy non-adaptive group testing problem under the combinatorial model. In the combinatorial model, there are $K$ defective items among a group of $N$ items. The problem is to identify all defective items by testing groups of items, with the minimum possible number of tests. The outcome of each test is a binary number. The focus of this work is when $N$ is limited rather than on the asymptotic regime.

We define the support vector $\mathbf{x} \in\{0,1\}^{N}$ to represent the set of $N$ items. The $i$-th component of $\mathbf{x}$, i.e., $x_{i}$, is 1 if and only if the $i$-th item is defective. In non-adaptive group testing, designing a testing scheme consisting of $M$ tests is equivalent to the construction of a binary matrix with $M$ rows which is referred to as measurement matrix. We let matrix $\mathbf{A} \in\{0,1\}^{M \times N}$ denote the measurement matrix. If $a_{t i}=1$ in the measurement matrix $\mathbf{A}$, it means that the $i$-th item is present in the $t$-th test. The design of measurement matrices for group testing has been studied extensively [63]. Our proposed algorithms are applicable for any measurement matrix; however, evaluation results are presented for Bernoulli designs in this study.

The standard noiseless group testing is formulated component-wise using the Boolean OR operation as $y_{t}=\bigvee_{i=1}^{N} a_{t i} x_{i}$ where $y_{t}$ and $\bigvee$ are the $t$ th test result and a Boolean OR operation, respectively. In this chapter, we consider the widely-adopted binary symmetric noise model where the values $\bigvee_{i=1}^{N} a_{t i} x_{i}$ are flipped independently at random with a given probability. The $t$ th test result in a binary symmetric noise model is given by

$$
y_{t}= \begin{cases}\bigvee_{i=1}^{N} a_{t i} x_{i} & \text { with probability } 1-\rho \\ 1 \oplus \bigvee_{i=1}^{N} a_{t i} x_{i} & \text { with probability } \rho\end{cases}
$$



Figure 5.1: An example of a factor graph representing a group testing scheme.
where $\oplus$ is the XOR operation. Note that this model and the proposed algorithms can be easily extended to include the general binary noise model where the values $\bigvee_{i=1}^{N} a_{t i} x_{i}$ are flipped from 0 to 1 and from 1 to 0 with different probabilities. However, for ease of exposition, we focus only on the binary symmetric noise model. We let vector $\mathbf{y} \in\{0,1\}^{M}$ denote the outcomes of the $M$ tests. The objective is to minimize the number of tests required to identify the set of defective items while meeting a target success probability, false positive and false negative rates. These metrics are formally defined in Section 5.4.

### 5.3 Decoding Algorithms

### 5.3.1 Belief Propagation

The Belief Propagation (BP) algorithm have gained promising success in different applications in recent years. It has been applied successfully to the problems in the area of coding theory and compressed sensing. Most of these works consider the asymptotic regime; however, we want to apply the BP algorithm to the practical regime where the number of item is limited. To apply the BP algorithm we consider the factor graph (Tan-
ner graph) representation of the group testing scheme as shown in Figure 5.1. In the Tanner graph, there are $N$ nodes at the left side of the graph corresponding to items. Also, there are $M$ nodes at the right side corresponding to the tests. This graph shows the connections between the items and the tests. Each item node is connected to the test nodes that the item participates in, according to the measurement matrix. In general, in a BP algorithm, messages are exchanged between the nodes of the graph. For a loopy BP algorithm, the messages are passed iteratively from items to tests and vice versa. We let $\mu_{i \rightarrow t}=\left[\mu_{i \rightarrow t}(0) \mu_{i \rightarrow t}(1)\right]$ and $\mu_{t \rightarrow i}=\left[\mu_{t \rightarrow i}(0) \mu_{t \rightarrow i}(1)\right]$ denote the message from item $i$ to test $t$ and the message from test $t$ to item $i$, respectively, where

$$
\left\{\begin{array}{l}
\mu_{i \rightarrow t}(0) \propto\left(1-\frac{K}{N}\right) \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(0),  \tag{5.1}\\
\mu_{i \rightarrow t}(1) \propto \frac{K}{N} \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(1)
\end{array}\right.
$$

where $\propto$ indicates equality up to a normalizing constant, and $\mathcal{N}(i)$ denotes the neighbours of the item node $i$. Note that these messages follow a probability distribution, i.e., $\mu_{i \rightarrow t}(0)+\mu_{i \rightarrow t}(1)=1$. For both combinatorial and probabilistic models, we initialize the messages by

$$
\begin{equation*}
\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=\frac{K}{N} . \tag{5.2}
\end{equation*}
$$

The messages from tests to items are given as follows. If $y_{t}=0$, we have

$$
\left\{\begin{array}{l}
\mu_{t \rightarrow i}(0) \propto \rho+(1-2 \rho) \prod_{i^{\prime} \in \mathcal{N}(t) \backslash\{t\}} \mu_{i^{\prime} \rightarrow t}(0),  \tag{5.3}\\
\mu_{t \rightarrow i}(1) \propto \rho
\end{array}\right.
$$

and if $y_{t}=1$, we have

$$
\left\{\begin{array}{l}
\mu_{t \rightarrow i}(0) \propto 1-\rho-(1-2 \rho) \prod_{i^{\prime} \in \mathcal{N}(t) \backslash\{t\}} \mu_{i^{\prime} \rightarrow t}(0)  \tag{5.4}\\
\mu_{t \rightarrow i}(1) \propto 1-\rho
\end{array}\right.
$$

Again, these messages follow a probability distribution, i.e., $\mu_{t \rightarrow i}(0)+\mu_{t \rightarrow i}(1)=1$.
A fixed point iteration is performed using the BP equations (5.1), (5.3), and (5.4). The algorithm stops after a fixed number $T$ of iterations. We choose the parameter $T$ experimentally. In the end, we compute the marginals of the posterior distribution as follows.

$$
\left\{\begin{array}{l}
q\left(x_{i}=0\right) \propto\left(1-\frac{K}{N}\right) \prod_{t^{\prime} \in \mathcal{N}(i)} \mu_{t^{\prime} \rightarrow i}(0)  \tag{5.5}\\
q\left(x_{i}=1\right) \propto \frac{K}{N} \prod_{t^{\prime} \in \mathcal{N}(i)} \mu_{t^{\prime} \rightarrow i}(1)
\end{array}\right.
$$

It is more convenient to compute the Log-Likelihood Ratio (LLR) of a marginal and work with it instead.

$$
\begin{equation*}
\lambda_{i}=\ln \frac{q\left(x_{i}=1\right)}{q\left(x_{i}=0\right)}=\ln \frac{K}{N-K}+\sum_{t^{\prime} \in \mathcal{N}(i)} \ln \frac{\mu_{t^{\prime} \rightarrow i}(1)}{\mu_{t^{\prime} \rightarrow i}(0)} \tag{5.6}
\end{equation*}
$$

For the combinatorial model, we sort the LLRs of the marginals in decreasing order and announce the items corresponding to the top $K$ LLRs to be the defective items. Algorithm 1 defines the BP algorithm.

### 5.3.2 Random Scheduling Belief Propagation

Traditional BP algorithm utilizes flooding scheduling, i.e., in each iteration all messages are updated simultaneously. However, in random scheduling, we update the messages from test nodes to item nodes in a randomized fashion. We start from initialization

```
Algorithm 1 Belief Propagation
    Initialize \(\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=\frac{K}{N} \forall i \in[N], \forall t \in \mathcal{N}(i)\)
    for \(j=1,2, \cdots, \mathrm{~T}\) do
        Compute \(\mu_{t \rightarrow i}(0)\) and \(\mu_{t \rightarrow i}(1) \forall t \in[M], \forall i \in \mathcal{N}(t)\)
        Compute \(\mu_{i \rightarrow t}(0)\) and \(\mu_{i \rightarrow t}(1) \forall i \in[N], \forall t \in \mathcal{N}(i)\)
    end for
    Compute \(\lambda_{i} \forall i \in[N]\)
```

of the messages from test nodes to their neighboring item nodes as follows.

$$
\begin{equation*}
\mu_{t \rightarrow i}(0)=\mu_{t \rightarrow i}(1)=\frac{1}{2} . \tag{5.7}
\end{equation*}
$$

Also, we initialize the messages from item nodes to their neighboring test nodes as in (6.5). After that, for each iteration $j \geq 1$, when we want to update the messages from tests to items, we randomly choose a test node and only send messages from that test node to its neighbouring item nodes. Next, we update the messages from the neighbouring item nodes of this test node. In the end, we identify the defective items in a similar way that was explained for the traditional BP algorithm. Random scheduling BP is formally described in Algorithm 2.

```
Algorithm 2 Random Scheduling Belief Propagation
    Initialize \(\mu_{t \rightarrow i}(0)=\mu_{t \rightarrow i}(1)=\frac{1}{2} \forall t \in[M], \forall i \in \mathcal{N}(t)\)
    Initialize \(\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=\frac{K}{N} \forall i \in[N], \forall t \in \mathcal{N}(i)\)
    for \(j=1,2, \cdots, \mathrm{~T}\) do
        Select a test node \(t^{\prime}\) at random
        Compute \(\mu_{t^{\prime} \rightarrow i}(0)\) and \(\mu_{t^{\prime} \rightarrow i}(1) \forall i \in \mathcal{N}\left(t^{\prime}\right)\)
        for each \(i \in \mathcal{N}\left(t^{\prime}\right)\) do
            Compute \(\mu_{i \rightarrow s}(0)\) and \(\mu_{i \rightarrow s}(1) \forall s \in \mathcal{N}(i)\)
        end for
    end for
    Compute \(\lambda_{i} \forall i \in[N]\)
```


### 5.3.3 Node-wise Residual Belief Propagation

In order to obtain a better performance over the traditional BP algorithm for decoding LDPC codes, the authors of [64] propose node-wise Residual Belief Propagation (RBP). Here, we adopt this algorithm for the noisy group testing problem. Similar to the random scheduling, in node-wise RBP, we choose a test node in each iteration and only update the messages from that test node to its neighboring item nodes. However, unlike the random scheduling, this test node is not chosen at random. We choose this test node using an ordering metric called the residual. In order to compute the residual for the messages from the test node $t \in[M]$ to the item node $i \in \mathcal{N}(t)$, we first compute $\lambda_{t \rightarrow i}=\ln \frac{\mu_{t \rightarrow i}(1)}{\mu_{t \rightarrow i}(0)}$ as the LLR of the most updated messages $\mu_{t \rightarrow i}(0)$ and $\mu_{t \rightarrow i}(1)$ from the test node $t$ to the item node $i$ in previous iterations. For each $t \in[M]$ and $i \in \mathcal{N}(t)$, we also compute $\lambda_{t \rightarrow i}^{*}=\ln \frac{\mu_{t \rightarrow i}^{*}(1)}{\mu_{t \rightarrow i}^{*}(0)}$, where $\mu_{t \rightarrow i}^{*}(0)$ and $\mu_{t \rightarrow i}^{*}(1)$ are computed using (6.7) or (6.8) depending on $y_{t}=0$ or $y_{t}=1$, as the pseudo-updated messages from the test node $t$ to the item node $i$ in the current iteration assuming that the test node $t$ is to be scheduled in this iteration. The residual $r_{t \rightarrow i}$ is then defined as the absolute value of the difference between the LLR of the most updated messages in previous iterations and the LLR of the pseudo-updated messages from the test node $t$ to item node $i$, i.e., $r_{t \rightarrow i}=\left|\lambda_{t \rightarrow i}^{*}-\lambda_{t \rightarrow i}\right|$. In each iteration, we select a test node $t$ such that $r_{t \rightarrow i}$ has the highest value among all $t \in[M]$ and all $i \in$ $\mathcal{N}(t)$, and update the messages from the selected test node to its neighboring item nodes. (Note that the messages from other test nodes to their neighboring item nodes will not be updated in this iteration.) The idea behind this strategy is that the differences between the LLRs before and after an update approaches zero as loopy BP converges. Hence, a large residual means that the corresponding test node is located in a part of the graph that has not converged yet [64]. Node-wise RBP is formally described in Algorithm 3.

```
Algorithm 3 Node-wise Residual Belief Propagation
    Initialize \(\mu_{t \rightarrow i}(0)=\mu_{t \rightarrow i}(1)=\frac{1}{2} \forall t \in[M], \forall i \in \mathcal{N}(t)\)
    Initialize \(\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=\frac{K}{N} \forall i \in[N], \forall t \in \mathcal{N}(i)\)
    Compute \(r_{t \rightarrow i} \forall t \in[M], \forall i \in \mathcal{N}(t)\)
    for \(j=1,2, \cdots, \mathbf{T}\) do
        Let \(t^{\prime}=\underset{t \in[M]}{\operatorname{argmax}} \max _{i \in \mathcal{N}(t)} r_{t \rightarrow i}\)
        for each \(i \in \mathcal{N}\left(t^{\prime}\right)\) do
            Compute \(\mu_{t^{\prime} \rightarrow i}(0)\) and \(\mu_{t^{\prime} \rightarrow i}(1)\)
            Set \(r_{t^{\prime} \rightarrow i}=0\)
            for each \(t^{\prime \prime} \in \mathcal{N}(i) \backslash\left\{t^{\prime}\right\}\) do
                Compute \(\mu_{i \rightarrow t^{\prime \prime}}(0)\) and \(\mu_{i \rightarrow t^{\prime \prime}}(1)\)
                for each \(i^{\prime} \in \mathcal{N}\left(t^{\prime \prime}\right) \backslash\{i\}\) do
                    Compute \(r_{t^{\prime \prime} \rightarrow i^{\prime}}\)
                end for
            end for
        end for
    end for
    Compute \(\lambda_{i} \forall i \in[N]\)
```


### 5.4 Simulation Results

In this section, we compare the performance of the BP algorithm, the Random Scheduling BP (RSBP) algorithm, and the Node-Wise Residual BP (NW-RBP) algorithm for the combinatorial model of the defective items using three metrics. The first metric, success probability, shows the probability that an algorithm identifies all the defective items correctly. We also use False-Negative Rate (FNR) and False-Positive Rate (FPR) in our performance comparison. FNR is defined as the ratio of the number of defective items falsely classified as non-defective over the number of all defective items. Similarly, FPR is defined as the number of non-defective items falsely classified as defective over the number of all non-defective items. In our simulations, we consider the binary symmetric noise model with parameter $\rho$. Each result is averaged over 3000 experiments. The measurement matrix is constructed according to a Bernoulli design [63]. In a Bernoulli design,

| $N=100$ |  |  |  |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| K | $\rho$ | M | BP | RSBP |  |  | NW-RBP |  |  | Optimal |  |  |
|  |  |  | Suc. Pr. | Suc. Pr. | FNR | FPR | Suc. Pr. | FNR | FPR | Suc. Pr. | FNR | FPR |
| 2 | 0.01 | 20 | 0.8135 | 0.8559 | 0.0805 | 0.0016 | 0.8617 | 0.0756 | 0.0016 | 0.8768 | 0.0672 | 0.0014 |
|  |  | 25 | 0.9368 | 0.9521 | 0.0257 | 0.0005 | 0.9602 | 0.0233 | 0.0005 | 0.9689 | 0.0161 | 0.0003 |
|  |  | 30 | 0.9788 | 0.9821 | 0.0101 | 0.0002 | 0.9880 | 0.0072 | 0.0001 | 0.9914 | 0.0043 | 0.0001 |
|  | 0.03 | 25 | 0.8437 | 0.8832 | 0.0659 | 0.0013 | 0.9120 | 0.0480 | 0.0010 | 0.9147 | 0.0463 | 0.0009 |
|  |  | 30 | 0.9270 | 0.9470 | 0.0298 | 0.0006 | 0.9607 | 0.0211 | 0.0004 | 0.9673 | 0.0171 | 0.0003 |
|  |  | 35 | 0.9635 | 0.9750 | 0.0140 | 0.0003 | 0.9788 | 0.0113 | 0.0003 | 0.9892 | 0.0054 | 0.0001 |
|  | 0.05 | 30 | 0.8580 | 0.8893 | 0.0631 | 0.0013 | 0.9216 | 0.0441 | 0.0009 | 0.9325 | 0.0372 | 0.0008 |
|  |  | 35 | 0.9151 | 0.9441 | 0.0307 | 0.0006 | 0.9574 | 0.0239 | 0.0005 | 0.9701 | 0.0157 | 0.0003 |
|  |  | 40 | 0.9524 | 0.9620 | 0.0213 | 0.0004 | 0.9818 | 0.0092 | 0.0002 | 0.9854 | 0.0076 | 0.0002 |
| 4 | 0.01 | 40 | 0.8037 | 0.8395 | 0.0412 | 0.0015 | 0.8359 | 0.0415 | 0.0017 | 0.8388 | 0.0412 | 0.0016 |
|  |  | 45 | 0.8961 | 0.9040 | 0.0249 | 0.0010 | 0.9055 | 0.0233 | 0.0010 | 0.9145 | 0.0220 | 0.0009 |
|  |  | 50 | 0.9465 | 0.9471 | 0.0137 | 0.0006 | 0.9482 | 0.0135 | 0.0005 | 0.9488 | 0.0131 | 0.0005 |
|  | 0.03 | 50 | 0.8583 | 0.8650 | 0.0360 | 0.0015 | 0.8799 | 0.0331 | 0.0014 | 0.8851 | 0.0316 | 0.0013 |
|  |  | 55 | 0.9065 | 0.9268 | 0.0193 | 0.0009 | 0.9291 | 0.0181 | 0.0008 | 0.9366 | 0.0169 | 0.0007 |
|  |  | 60 | 0.9458 | 0.9538 | 0.0099 | 0.0004 | 0.9559 | 0.0098 | 0.0004 | 0.9598 | 0.0095 | 0.0004 |
|  | 0.05 | 60 | 0.8751 | 0.8991 | 0.0294 | 0.0012 | 0.9013 | 0.0283 | 0.0011 | 0.9071 | 0.0242 | 0.0010 |
|  |  | 65 | 0.9207 | 0.9422 | 0.0165 | 0.0006 | 0.9450 | 0.0146 | 0.0006 | 0.9474 | 0.0136 | 0.0005 |
|  |  | 70 | 0.9471 | 0.9564 | 0.0116 | 0.0005 | 0.9583 | 0.0109 | 0.0004 | 0.9596 | 0.0104 | 0.0004 |

Table 5.1: The performance of different decoding algorithms for the combinatorial model when $N=100$.
each item is included in each test independently at random with some fixed probability $\nu / K$ where $K$ is the number of defective items and we set $\nu=\ln 2$.

In Table 5.1, we present experimental simulation results for $N=100$ items and $K=$ 2, 4 defective items. Corresponding to each value of $K$, we consider three different values for the noise parameter $\rho=0.01,0.03,0.05$. And for each value of $\rho$, we consider three different number of tests. The optimal decoder which is used as a benchmark here is the maximum-likelihood decoder. The results in Table 5.1 show that the NW-RBP algorithm outperforms the BP algorithm and the RSBP algorithm for all problem parameters being considered. The NW-RBP algorithm perform fairly close to the optimal decoder for a large set of parameters, particularly when the success probability approaches one. Note that although the RSBP algorithm is inferior to the NW-RBP algorithm, it outperforms the BP algorithm.

## 6. NOISY GROUP TESTING WITH SIDE INFORMATION

### 6.1 Introduction

Identifying infected people is a critical step in dealing with pandemics caused by viral diseases. However, testing a large number of people individually might be prohibitively expensive for practical reasons. For this reason, we need to deploy strategies that allow efficient testing. Group Testing (GT) has been shown as an efficient strategy in reducing the number of tests required to test for pandemics. An instance of the GT problem includes a set $\mathbf{S}$ of $N$ individuals which includes a small subset of infected individuals. The GT procedure consists of a sequence of tests, such that each test indicates whether there are one or more infected individuals in a given subset of $\mathbf{S}$. The goal of the GT procedure is to identify the subset of infected individuals through the minimum number of tests.

The GT problem has been the subject of many studies. Most studies have focused on the following two models $[15,43,60,66]$ : (i) a combinatorial model which assumes that the number of infected individuals is fixed and known; (ii) a probabilistic model which assumes that each individual is infected with a certain probability. There are also two types of GT algorithms: non-adaptive, and adaptive. In this study, we are interested in non-adaptive GT strategies, where all tests are designed in advance. This is in contrast to adaptive strategies, in which the design of each test depends on the results of the previous tests [14, 50, 63, 67].

Motivated by practical scenarios where the outcome of the tests can be affected by noise, we focus on the noisy GT setting, in which the outcome of a test can be flipped with some probability. In the noisy GT setting, the goal is to identify the set of infected individuals with high probability $(1-\varepsilon)$, for small values of $\varepsilon$. We also focus on a variation of a probabilistic GT model in which the prior infection probability is not uniform and in
which there is a certain amount of side information on the distribution of the infected individuals available to the GT algorithm.

A GT algorithm consists of two parts: encoding and decoding. The encoding part is concerned with the test design, i.e., the decision on which individuals to include in each test. The decoding part is concerned with identifying the infected individuals given the test design and outcomes of the tests. Different decoding algorithms such as linear programming, combinatorial orthogonal matching pursuit, definite defectives, belief propagation (BP), and separate decoding of items have been proposed for noisy non-adaptive GT. A thorough review and comparison of these algorithms is provided in [63]. In the context of GT, it is extremely difficult to analyze the performance of BP algorithms even for the asymptotic regime. To the best of our knowledge, no theoretical analysis has been provided for the BP-based GT algorithms so far. However, empirical evidence suggests that the BP algorithm results in lower error probabilities compared to other algorithms for the probabilistic model. BP is a message passing algorithm that passes messages over the edges in the underlying factor graph representation of the GT problem. For a cycle-free factor graph, BP decoding is equivalent to Maximum a Posteriori (MAP) decoding. However, in the presence of loops in the factor graph, BP becomes suboptimal.

This study focuses on leveraging the side information for improving the performance of BP-based decoding algorithms for noisy GT. In the context of testing for viral infections, different forms of side information can be exploited including the prevalence rate, individuals' symptoms, family structure, community structure, and contact tracing information. It has been shown that side information can be used to reduce the required tests $[61,62,68-70]$. For example, Zhu et al. [68] show that the number of tests can be reduced if the prior information about the prevalence rate is takend into account. Nikolopoulos et al. $[61,69]$ and Ahn et al. [62] show that utilizing community structure also leads to a lower number of tests. While the focus of these works is on the encoder design, in our
work we focus on leveraging the side information for the efficient decoder design.

### 6.1.1 Contribution.

In this work, first, we propose a probabilistic model, referred to as an interaction model, that captures the side information about the probability distribution of the infected individuals. Our model is motivated by the availability of contact tracing information which can be collected from surveys and mobile phone applications [71-73]. Next, we present a decoding scheme, based on belief propagation, that leverages the interaction model to improve the decoding accuracy. Our results indicate that the proposed algorithm achieves higher success probability and lower false-negative and false-positive rates when compared to the traditional belief propagation especially in the high noise regime.

### 6.2 Probabilistic Model

Throughout the chapter, we denote vectors and matrices by bold-face small and capital letters, respectively. For an integer $i \geq 1$, we denote $\{1, \ldots, i\}$ by $[i]$. Let $\mathbf{S}$ be a set. The set of all subsets of size $\ell$ for set $S$ is denoted by $\{S\}_{\ell}$.

Our model assumes that there are two points in time, namely time 0 and time 1 such that time 0 occurs prior to time 1 . Let $N$ be the total number of individuals. We define the vector $\mathbf{x}^{(0)} \in\{0,1\}^{N}$ to represent the status of $N$ individuals at time 0 , such that $x_{i}^{(0)}$ is 1 if the $i$-th individual is infected at time 0 , and is 0 otherwise. Similarly, we define the vector $\mathbf{x}^{(1)} \in\{0,1\}^{N}$ to represent the status of $N$ individuals at time 1 . We assume that at time 0 , the probability of an individual being infected is equal to the prevalence rate $p$, and that the infection of each individual at time 0 occurs independently of other individuals. The probability of the individual to be infected at time 1 depends on their probability to be infected at time 0 as well as their interaction with other individuals. The interactions of individuals between time 0 and time 1 is captured by the interaction graph. For each individual $i$, the graph includes nodes $x_{i}^{(0)}$ and $x_{i}^{(1)}$ that represent that individual


Figure 6.1: An example of an interaction graph. Nodes $x_{i}^{(0)}$ and $x_{i}^{(1)}$ represent individual $i$ at times 0 and 1 , respectively. An interaction node $I_{i}$ captures interactions between individual $i$ and other individuals from time 0 to time 1 .
at times 0 and 1 , respectively. For each individual $i$, the graph has an interaction node $I_{i}$ that captures interactions between individual $i$ and other individuals from time 0 to time 1. In particular, the graph contains an edge $\left(x_{j}^{(0)}, I_{i}\right)$ for each individual $j$ who have been in contact with individual $i$ from time 0 to time 1 . An example of an interaction graph is shown in Fig. 6.1.

We assume that an infected individual infects a healthy individual with probability $q$, referred to as contagion probability. It is also assumed that if an individual is infected at time 0 , they remain infected by time 1 . The interaction model can be used to find the probability that an individual at time 1 is infected. The $i$ th individual is not infected at time 1 if the following holds: 1) the $i$ th individual is not infected at time 0 , and 2) other individuals in contact with the $i$ th individual either are not infected at time 0 or, if infected, they do not infect the $i$ th individual. Thus, the probability of individual $i$ to be infected at time 1 can be calculated as follows:

$$
P\left(x_{i}^{(1)}=0\right)=(1-p)(1-p+p(1-q))^{d_{i}}=(1-p)(1-p q)^{d_{i}}
$$

where $d_{i}$ is the number of individuals which interact with individual $i$ from time 0 to time 1. The probability that individual $i$ is infected at time 1 is given by

$$
\begin{equation*}
\pi_{i}=P\left(x_{i}^{(1)}=1\right)=1-P\left(x_{i}^{(1)}=0\right)=1-(1-p)(1-p q)^{d_{i}} . \tag{6.1}
\end{equation*}
$$

Our ultimate goal is to test and identify infected individuals at time 1 , assuming the knowledge of the probabilistic model described above. This model can be extended to capture interactions between individuals in more than one round. For ease of exposition, we consider only one round of interactions in this study.

In non-adaptive group testing, designing a testing scheme consisting of $M$ tests is equivalent to the construction of a binary matrix with $M$ rows which is referred to as a testing matrix. We let matrix $\mathbf{A} \in\{0,1\}^{M \times N}$ denote the testing matrix. The entry $(t, i)$ of matrix $\mathbf{A}$ is denoted by $a_{t, i}$. If $a_{t, i}=1$, it means that the $i$-th item is present in the $t$-th test. The design of testing matrices for group testing has been studied extensively (see e.g., [63]). Our proposed algorithms are applicable for any testing matrix; however, simulation results are presented for Bernoulli designs. In a Bernoulli design, each individual is included in each test independently at random with some fixed probability $\nu / K$ where $K$ is the average number of defective items and $\nu$ is a constant.

The standard noiseless group testing is formulated component-wise using the Boolean OR operation as $y_{t}=\bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)}$ where $y_{t}$ and $\bigvee$ are the $t$ th test result and a Boolean OR operation, respectively. In this study, we consider the widely-adopted binary symmetric noise model where the values $\bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)}$ are flipped independently at random with a given probability. The $t$ th test result in a binary symmetric noise model is given by

$$
y_{t}= \begin{cases}\bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)} & \text { with probability } 1-\rho, \\ 1 \oplus \bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)} & \text { with probability } \rho\end{cases}
$$

where $\oplus$ is the XOR operation and $\rho$ is the probability that the values $\bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)}$ are flipped. Note that this model and the proposed algorithms can be easily extended to include the general binary noise model where the values $\bigvee_{i=1}^{N} a_{t, i} x_{i}^{(1)}$ are flipped from 0 to 1 and from 1 to 0 with different probabilities independently. However, for ease of exposition, we focus only on the binary symmetric noise model. We let vector $\mathbf{y} \in\{0,1\}^{M}$ denote the outcomes of the $M$ tests.

Our objective is to design a decoding algorithm that performs well under the following three metrics: (i) success probability which captures the probability that all infected individuals are identified correctly; (ii) False-Negative Rate (FNR), which is the number of infected individuals falsely classified as healthy over the total number of infected individuals, (iii) False-Positive Rate (FPR), defined as the ratio of the number of healthy individuals falsely classified as infected and the total number of healthy individuals.

### 6.3 Proposed Decoding Algorithms

In this section, we describe the proposed decoding algorithms for retrieving the status vector $\mathbf{x}^{(1)}$ from the test results vector $\mathbf{y}$ and the testing matrix $\mathbf{A}$.

### 6.3.1 Belief Propagation Using Initial Prior Probabilities

Message passing algorithms are utilized to solve inference problems, optimization problems, and constraint satisfaction problems. In an inference problem, there are some noisy measurements as input, and the goal is to infer the value of some unobserved variables from those measurements. It is impossible, in general, to make those inferences with complete certainty, but one can try to obtain the most probable value of the unobserved variables [74,75]. In a probabilistic noisy group testing, we intend to perform a Maximum a Posteriori (MAP) estimation to find the status vector $\hat{\mathbf{x}}^{(1)}$ given the test results vector $\mathbf{u}$.

$$
\begin{equation*}
\underset{\hat{\mathbf{x}}^{(1)}}{\operatorname{argmax}} P\left(\mathbf{x}^{(1)}=\hat{\mathbf{x}}^{(1)}\right) P\left(\mathbf{y}=\mathbf{u} \mid \mathbf{x}^{(1)}=\hat{\mathbf{x}}^{(1)}\right) \tag{6.2}
\end{equation*}
$$

This problem can be solved using exhaustive search when the vector $\mathbf{x}^{(1)}$ is small, e.g., $N \approx 10-20$. However, the exhaustive search approach rapidly becomes intractable when $N$ increases. For instance, when $N=100$ (a relatively small problem), the number of different configurations for the vector $\mathbf{x}^{(1)}$ is $2^{100}$. An alternative solution is to find the marginals of the posterior distribution for each item using Belief Propagation (BP). BP is a message passing algorithm for performing inference on factor graphs with the purpose of calculating the marginal distribution for each unobserved variable, conditional on observed variables. A factor graph is a type of probabilistic graphical model which is used to visualize and precisely define the underlying optimization problem. Factor graphs are bipartite graphs with two types of nodes referred to as variable nodes and factor nodes. See Fig. 6.2 for an example of a factor graph. The variable nodes which represent the variables in the optimization problem are represented by circles. The factor nodes show how the overall cost function can be factorized into local cost functions and are represented by squares. There is an edge between the variables that are involved in a local cost function and the factor node representing that local cost function.

We assume that the only side information we have is the prevalence rate at time 0 . Since we have no information about the status of individuals at time 1 , we consider the prior probability of each individual being infected at time 1 to be equal to the prevalence rate $p$, independent of other individuals. The overall cost function in (6.2) can be factorized as follows.

$$
\begin{align*}
P\left(\mathbf{x}^{(1)}=\hat{\mathbf{x}}^{(1)}\right) P\left(\mathbf{y}=\mathbf{u} \mid \mathbf{x}^{(1)}=\hat{\mathbf{x}}^{(1)}\right) & =\left[\prod_{i=1}^{N} P\left(x_{i}^{(1)}=\hat{x}_{i}^{(1)}\right)\right] \\
& \times\left[\prod_{t=1}^{M} P\left(y_{t}=u_{t} \mid\left\{x_{i}^{(1)}=\hat{x}_{i}^{(1)}\right\}_{i \in \mathcal{N}(t)}\right)\right] \tag{6.3}
\end{align*}
$$



Figure 6.2: An example of a factor graph representing a group testing scheme. A variable node $x_{i}^{(1)}$ represents individual $i$ at time 1 . A factor node $y_{t}$ represents test $t$. Factor nodes represented by dotted squares correspond to the a priori probability distribution of the variable nodes.
where $\mathcal{N}(t)$ denotes the indices of individuals involved in the $t$ th test.
In order to apply BP, we consider the factor graph (Tanner graph) representation of the group testing scheme. In the Tanner graph, there are $N$ variable nodes that represent individuals at time 1. There are also $M$ factor nodes that represent the tests. Each test factor node corresponds to the conditional probability distribution of a test result, given the observed variable nodes. Each individual in the Tanner graph is connected to the test in which the individual participates, according to the testing matrix. There are also $N$ factor nodes that correspond to the a priori probability distribution of the variable nodes. Since these factor nodes are usually not exhibited in a Tanner graph, we show them using dotted squares in Fig. 6.2.

For a cycle-free factor graph, BP decoding is equivalent to MAP decoding. However, in the presence of loops in the factor graph, BP becomes suboptimal. In other words, loopy BP computes an approximation of the marginals of the posterior distribution for each variable node. For a loopy BP algorithm, the messages are passed iteratively from
variable nodes to factor nodes and vice versa. We let $\mu_{i \rightarrow t}=\left[\mu_{i \rightarrow t}(0) \mu_{i \rightarrow t}(1)\right]$ and $\mu_{t \rightarrow i}=$ $\left[\mu_{t \rightarrow i}(0) \mu_{t \rightarrow i}(1)\right]$ denote the message from individual $i$ to test $t$ and the message from test $t$ to individual $i$, respectively. In general, the message from variable node $i$ to factor node $t$ is given by computing the product of all incoming messages from the neighboring factor nodes of variable node $i$ excluding the message from factor node $t$.

$$
\left\{\begin{array}{l}
\mu_{i \rightarrow t}(0) \propto(1-p) \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(0),  \tag{6.4}\\
\mu_{i \rightarrow t}(1) \propto p \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(1),
\end{array}\right.
$$

where $\propto$ indicates equality up to a normalizing constant, and $\mathcal{N}(i)$ denotes the indices of tests in which item $i$ participates. Note that these messages are probability distributions on $\{0,1\}$, i.e., $\mu_{i \rightarrow t}(0)+\mu_{i \rightarrow t}(1)=1$. Since we assume that the prior probability of each individual being infected at time 1 is equal to $p$, the messages are initialized by

$$
\begin{equation*}
\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=p \tag{6.5}
\end{equation*}
$$

The messages from factor nodes to variable nodes are computed as follows. The message from test $t$ to individual $i$ is given by

$$
\begin{align*}
& \mu_{t \rightarrow i}\left(\hat{x}_{i}^{(1)}\right)= \\
& \sum_{\substack{\hat{x}_{i^{\prime}}^{(1)} \in\{0,1\} \\
u_{t} \in\{0,1\}}}\left[P\left(y_{t}=u_{t} \mid x_{i}^{(1)}=\hat{x}_{i}^{(1)},\left\{x_{i^{\prime}}^{(1)}=\hat{x}_{i^{\prime}}^{(1)}\right\}_{i^{\prime} \in \mathcal{N}(t) \backslash\{i\}}\right) \times \prod_{i^{\prime} \in \mathcal{N}(t) \backslash\{i\}} \mu_{i^{\prime} \rightarrow t}\left(\hat{x}_{i^{\prime}}^{(1)}\right)\right] . \tag{6.6}
\end{align*}
$$

As was shown in [76-79], the equation (6.6) can be simplified as follows. If $y_{t}=0$, we
have

$$
\left\{\begin{array}{l}
\mu_{t \rightarrow i}(0) \propto \rho+(1-2 \rho) \prod_{i^{\prime} \in \mathcal{N}(t) \backslash\{i\}} \mu_{i^{\prime} \rightarrow t}(0),  \tag{6.7}\\
\mu_{t \rightarrow i}(1) \propto \rho
\end{array}\right.
$$

and if $y_{t}=1$, we have

$$
\left\{\begin{array}{l}
\mu_{t \rightarrow i}(0) \propto 1-\rho-(1-2 \rho) \prod_{i^{\prime} \in \mathcal{N}(t) \backslash\{i\}} \mu_{i^{\prime} \rightarrow t}(0)  \tag{6.8}\\
\mu_{t \rightarrow i}(1) \propto 1-\rho
\end{array}\right.
$$

We perform a fixed point iteration using the BP equations (6.4), (6.7), and (6.8). We stop the algorithm after a fixed number $T$ of iterations. The parameter $T$ is chosen experimentally. In the end, we compute the marginals of the posterior distribution for each variable node by computing the product of all incoming messages from the neighboring factor nodes of that variable node.

$$
\left\{\begin{array}{l}
\phi\left(x_{i}^{(1)}=0\right) \propto(1-p) \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(0)  \tag{6.9}\\
\phi\left(x_{i}^{(1)}=1\right) \propto p \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(1)
\end{array}\right.
$$

For convenience, we work with the Log-Likelihood Ratio (LLR) of a marginal defined as

$$
\begin{equation*}
\lambda_{i}=\ln \frac{\phi\left(x_{i}^{(1)}=1\right)}{\phi\left(x_{i}^{(1)}=0\right)}=\ln \frac{p}{1-p}+\sum_{t \in \mathcal{N}(i)} \ln \frac{\mu_{t \rightarrow i}(1)}{\mu_{t \rightarrow i}(0)} \tag{6.10}
\end{equation*}
$$

We consider a threshold, $\tau$, and announce the $i$ th individual infected if $\lambda_{i} \geq \tau$. A natural threshold one can choose is $\tau=0$. Note that values other than 0 are also permissible. Algorithm 4 defines the belief propagation using initial prior probabilities algorithm.

```
Algorithm 4 Belief Propagation Using Initial Prior Probabilities
    Initialize \(\mu_{i \rightarrow t}(1)=1-\mu_{i \rightarrow t}(0)=p \forall i \in[N], \forall t \in \mathcal{N}(i)\)
    for \(\ell=1,2, \cdots, T\) do
        Compute \(\mu_{t \rightarrow i}(0)\) and \(\mu_{t \rightarrow i}(1) \forall t \in[M], \forall i \in \mathcal{N}(t)\) using (6.7) and (6.8)
        Compute \(\mu_{i \rightarrow t}(0)\) and \(\mu_{i \rightarrow t}(1) \forall i \in[N], \forall t \in \mathcal{N}(i)\) using (6.4)
    end for
    Compute \(\lambda_{i} \forall i \in[N]\) using (6.10)
```


### 6.3.2 Belief Propagation Using Updated Prior Probabilities.

In this scheme, instead of using the prevalence rate at time 0 for the probability that an individual is infected at time 1 , we use the updated prior probability $\pi_{i}, i \in[N]$, given by (6.1). We perform the BP algorithm in Section 6.3 .1 where in the equations (6.4)(6.10), the initial prior probability $p$ is replaced by the updated prior probability $\pi_{i}$, for each $i \in[N]$.

### 6.3.3 Belief Propagation on Combined Graphs

In this scheme, assuming that the contact tracing information is available, we form the interaction graph and combine it with the Tanner graph corresponding to the testing matrix. An example of a combined graph is presented in Fig. 6.3. We then perform a BP algorithm over the combined graph. Note that there are two sets of variable nodes in the combined graph, $\left\{x_{i}^{(0)}\right\}_{i \in[N]}$ and $\left\{x_{i}^{(1)}\right\}_{i \in[N]}$. We are interested in computing the marginals of the posterior distribution for $\left\{x_{i}^{(1)}\right\}_{i \in[N]}$. There are also three different types of factor nodes. The interaction node $I_{i}$ corresponds to the conditional probability that individual $i$ at time 1 is infected or not, given the status of individuals at time 0 who have been in contact with individual $i$. The test factor node $y_{t}$ corresponds to the conditional probability that the result of test $t$ is equal to a one or zero, given the status of neighboring individuals at time 1 . Furthermore, there are $N$ factor nodes, represented by dashed squares, that


Figure 6.3: An example of a combined graph. Nodes $x_{i}^{(0)}$ and $x_{i}^{(1)}$ represent individual $i$ at times 0 and 1, respectively. An interaction node $I_{i}$ captures interactions between individual $i$ and other individuals from time 0 to time 1 . A factor node $y_{t}$ represents test $t$. Factor nodes represented by dotted squares correspond to the a priori probability distribution of the status of individuals at time 0 .
correspond to the a priori probability that each of the individuals at time 0 is infected or not. The combined graph represents the factorization in the following joint probability mass function.

$$
\begin{aligned}
P\left(\mathbf{x}^{(0)}=\hat{\mathbf{x}}^{(0)}, \mathbf{x}^{(1)}=\hat{\mathbf{x}}^{(1)}, \mathbf{y}=\mathbf{u}\right) & =\left[\prod_{i=1}^{N} P\left(x_{i}^{(0)}=\hat{x}_{i}^{(0)}\right)\right] \\
& \times\left[\prod_{i=1}^{N} P\left(x_{i}^{(1)}=\hat{x}_{i}^{(1)} \mid\left\{x_{i^{\prime}}^{(0)}=\hat{x}_{i^{\prime}}^{(0)}\right\}_{i^{\prime} \in \mathcal{N}\left(I_{i}\right)}\right)\right] \\
& \times\left[\prod_{t=1}^{M} P\left(y_{t}=u_{t} \mid\left\{x_{i}^{(1)}=\hat{x}_{i}^{(1)}\right\}_{i \in \mathcal{N}(t)}\right)\right]
\end{aligned}
$$

where $\mathcal{N}\left(I_{i}\right)$ denotes the indices of individuals at time 0 who are connected to interaction node $I_{i}$.

As it has been mentioned before, in a loopy BP algorithm, the messages are passed iteratively from variable nodes to factor nodes and vice versa. In what follows, we show the flow of messages in one iteration.

First, individuals at time 0 send their messages to interaction nodes. We let $\gamma_{i \rightarrow I_{j}}=$ $\left[\gamma_{i \rightarrow I_{j}}(0) \gamma_{i \rightarrow I_{j}}(1)\right]$ and $\gamma_{I_{j} \rightarrow i}=\left[\gamma_{I_{j} \rightarrow i}(0) \gamma_{I_{j} \rightarrow i}(1)\right]$ denote the message from individual $i$ at time 0 to interaction node $I_{j}$ and the message from interaction node $I_{j}$ to individual $i$ at time 0 , respectively. It is easy to show that the messages $\gamma_{i \rightarrow I_{j}}$ can be computed as follows:

$$
\left\{\begin{array}{l}
\gamma_{i \rightarrow I_{j}}(0) \propto(1-p) \prod_{j^{\prime} \in \mathcal{N}(i) \backslash\{j\}} \gamma_{I_{j^{\prime}} \rightarrow i}(0),  \tag{6.11}\\
\gamma_{i \rightarrow I_{j}}(1) \propto p \prod_{j^{\prime} \in \mathcal{N}(i) \backslash\{j\}} \gamma_{I_{j^{\prime}} \rightarrow i}(1)
\end{array}\right.
$$

where these messages are initialized by $\gamma_{i \rightarrow I_{j}}(1)=1-\gamma_{i \rightarrow I_{j}}(0)=p$.
Then, the interaction nodes send their messages to individuals at time 1 . We denote the message from interaction node $I_{j}$ to individual $j$ at time 1 and the message from individual $j$ at time 1 to interaction node $I_{j}$ by $\delta_{I_{j} \rightarrow j}=\left[\delta_{I_{j} \rightarrow j}(0) \delta_{I_{j} \rightarrow j}(1)\right]$ and $\delta_{j \rightarrow I_{j}}=$ $\left[\delta_{j \rightarrow I_{j}}(0) \delta_{j \rightarrow I_{j}}(1)\right]$, respectively. It can be shown that the message $\delta_{I_{j} \rightarrow j}$ is given by

$$
\left\{\begin{array}{l}
\delta_{I_{j} \rightarrow j}(0) \propto \gamma_{j \rightarrow I_{j}}(0) \prod_{i \in \mathcal{N}\left(I_{j}\right) \backslash\{j\}}\left(1-q \gamma_{i \rightarrow I_{j}}(1)\right),  \tag{6.12}\\
\delta_{I_{j} \rightarrow j}(1) \propto \gamma_{j \rightarrow I_{j}}(1)-\gamma_{j \rightarrow I_{j}}(0) \sum_{\ell=1}^{\left|\mathcal{N}\left(I_{j}\right)\right|-1} \sum_{S \in\left\{\mathcal{N}\left(I_{j}\right) \backslash\{j\}\right\}_{\ell}}(-q)^{\ell} \prod_{i \in S} \gamma_{i \rightarrow I_{j}}(1) .
\end{array}\right.
$$

In the next step, individuals at time 1 send their messages to test factor nodes. We let $\mu_{i \rightarrow t}=\left[\mu_{i \rightarrow t}(0) \mu_{i \rightarrow t}(1)\right]$ and $\mu_{t \rightarrow i}=\left[\mu_{t \rightarrow i}(0) \mu_{t \rightarrow i}(1)\right]$ denote the message from individual $i$ at time 1 to test $t$ and the message from test $t$ to individual $i$ at time 1 , respectively.

$$
\left\{\begin{array}{l}
\mu_{i \rightarrow t}(0) \propto \delta_{I_{i} \rightarrow i}(0) \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(0),  \tag{6.13}\\
\mu_{i \rightarrow t}(1) \propto \delta_{I_{i} \rightarrow i}(1) \prod_{t^{\prime} \in \mathcal{N}(i) \backslash\{t\}} \mu_{t^{\prime} \rightarrow i}(1) .
\end{array}\right.
$$

Now, test nodes send their messages to individuals at time 1. The message $\mu_{t \rightarrow i}$ is calculated in a similar way as in (6.7) and (6.8). Next, individuals at time 1 send their messages to interaction nodes. It can be shown that the message from individual $i$ at time 1 to interaction node $I_{i}$ is given by

$$
\left\{\begin{array}{l}
\delta_{i \rightarrow I_{i}}(0) \propto \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(0)  \tag{6.14}\\
\delta_{i \rightarrow I_{i}}(1) \propto \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(1)
\end{array}\right.
$$

Finally, interaction nodes send their messages to individuals at time 0 . The message from interaction node $I_{j}$ to individual $j$ at time 0 is given as follows.

$$
\left\{\begin{align*}
\gamma_{I_{j} \rightarrow j}(0) & \propto \delta_{j \rightarrow I_{j}}(0) \prod_{\substack{i \in \mathcal{N}\left(I_{j}\right) \backslash\{j\}}}\left(1-q \gamma_{i \rightarrow I_{j}}(1)\right)  \tag{6.15}\\
& -\delta_{j \rightarrow I_{j}}(1) \sum_{\ell=1}^{\left|\mathcal{N}\left(I_{j}\right)\right|-1} \sum_{S \in\left\{\mathcal{N}\left(I_{j}\right) \backslash\{j\}\right\}_{\ell}}(-q)^{\ell} \prod_{i \in S} \gamma_{i \rightarrow I_{j}}(1), \\
\gamma_{I_{j} \rightarrow j}(1) & \propto \delta_{j \rightarrow I_{j}}(1) .
\end{align*}\right.
$$

Let us define

$$
f(i, j) \triangleq \sum_{\ell=1}^{\left|\mathcal{N}\left(I_{j}\right)\right|-2} \sum_{S \in\left\{\mathcal{N}\left(I_{j}\right) \backslash\{i, j\}\right\}_{\ell}}(-q)^{\ell} \prod_{i^{\prime} \in S} \gamma_{i^{\prime} \rightarrow I_{j}}(1)
$$

The message from interaction node $I_{j}$ to individual $i$ at time 0 , where $i \in \mathcal{N}\left(I_{j}\right) \backslash\{j\}$, is given by

$$
\left\{\begin{align*}
\gamma_{I_{j} \rightarrow i}(0) & \propto \gamma_{j \rightarrow I_{j}}(1) \delta_{j \rightarrow I_{j}}(1)+\gamma_{j \rightarrow I_{j}}(0) \delta_{j \rightarrow I_{j}}(0) \prod_{i^{\prime} \in \mathcal{N}\left(I_{j}\right) \backslash\{i, j\}}\left(1-q \gamma_{i^{\prime} \rightarrow I_{j}}(1)\right)  \tag{6.16}\\
& -\gamma_{j \rightarrow I_{j}}(0) \delta_{j \rightarrow I_{j}}(1) f(i, j), \\
\gamma_{I_{j} \rightarrow i}(1) & \propto \gamma_{j \rightarrow I_{j}}(1) \delta_{j \rightarrow I_{j}}(1)+\gamma_{j \rightarrow I_{j}}(0) \delta_{j \rightarrow I_{j}}(1)[q-(1-q) f(i, j)] \\
& +(1-q) \gamma_{j \rightarrow I_{j}}(0) \delta_{j \rightarrow I_{j}}(0) \prod_{i^{\prime} \in \mathcal{N}\left(I_{j}\right) \backslash\{i, j\}}\left(1-q \gamma_{i^{\prime} \rightarrow I_{j}}(1)\right) .
\end{align*}\right.
$$

In the end, when the algorithm is run for a fixed number $T$ of iterations, we compute the marginals of the posterior distribution for individuals at time 1 as follows.

$$
\left\{\begin{array}{l}
\phi\left(x_{i}^{(1)}=0\right) \propto \delta_{I_{i} \rightarrow i}(0) \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(0) \\
\phi\left(x_{i}^{(1)}=1\right) \propto \delta_{I_{i} \rightarrow i}(1) \prod_{t \in \mathcal{N}(i)} \mu_{t \rightarrow i}(1)
\end{array}\right.
$$

The LLRs of the marginals are given by

$$
\begin{equation*}
\lambda_{i}=\ln \frac{\phi\left(x_{i}^{(1)}=1\right)}{\phi\left(x_{i}^{(1)}=0\right)}=\ln \frac{\delta_{I_{i} \rightarrow i}(1)}{\delta_{I_{i} \rightarrow i}(0)}+\sum_{t \in \mathcal{N}(i)} \ln \frac{\mu_{t \rightarrow i}(1)}{\mu_{t \rightarrow i}(0)} \tag{6.17}
\end{equation*}
$$

The interpretation of the LLRs is done in the same way that has been explained in Section 6.3.1. For a given threshold $\tau$, individual $i$ at time 1 is announced infected if $\lambda_{i} \geq \tau$. Algorithm 5 defines the belief propagation on combined graphs algorithm.

Example 7. Consider the combined graph shown in Fig. 6.3. We want to compute the BP messages exchanged over the edges of the combined graph. Calculating the messages from variable nodes to factor nodes is straightforward. Thus, we intend to compute messages from factor nodes to variable nodes. Since in [76-79] it was shown that the messages from tests to individuals at time 1 are computed using (6.7) and (6.8), we only show how

```
Algorithm 5 Belief Propagation on Combined Graphs
    Initialize \(\gamma_{i \rightarrow I_{j}}(1)=1-\gamma_{i \rightarrow I_{j}}(0)=p \forall i \in[N], \forall j \in \mathcal{N}(i)\)
    Initialize \(\mu_{t \rightarrow i}(0)=\mu_{t \rightarrow i}(1)=\frac{1}{2} \forall t \in[M], \forall i \in \mathcal{N}(t)\)
    for \(\ell=1,2, \cdots, T\) do
        Compute \(\delta_{I_{i} \rightarrow i}(0)\) and \(\delta_{I_{i} \rightarrow i}(1) \forall i \in[N]\) using (6.12)
        Compute \(\mu_{i \rightarrow t}(0)\) and \(\mu_{i \rightarrow t}(1) \forall i \in[N], \forall t \in \mathcal{N}(i)\) using (6.13)
        Compute \(\mu_{t \rightarrow i}(0)\) and \(\mu_{t \rightarrow i}(1) \forall t \in[M], \forall i \in \mathcal{N}(t)\) using (6.7) and (6.8)
        Compute \(\delta_{i \rightarrow I_{i}}(0)\) and \(\delta_{i \rightarrow I_{i}}(1) \forall i \in[N]\) using (6.14)
        Compute \(\gamma_{I_{j} \rightarrow i}(0)\) and \(\gamma_{I_{j} \rightarrow i}(1) \forall j \in[N], \forall i \in \mathcal{N}\left(I_{j}\right)\) using (6.15) and (6.16)
        Compute \(\gamma_{i \rightarrow I_{j}}(0)\) and \(\gamma_{i \rightarrow I_{j}}(1) \forall i \in[N], \forall j \in \mathcal{N}(i)\) using (6.11)
    end for
    Compute \(\lambda_{i} \forall i \in[N]\) using (6.17)
```

to compute messages from interaction nodes to individuals at time 0 and time 1 . The message from interaction node $I_{j}$ to individual $j$ at time 1 is computed using

$$
\delta_{I_{j} \rightarrow j}\left(\hat{x}_{j}^{(1)}\right)=\sum_{\hat{x}_{i}^{(0)} \in\{0,1\}}\left[P\left(x_{j}^{(1)}=\hat{x}_{j}^{(1)} \mid\left\{x_{i}^{(0)}=\hat{x}_{i}^{(0)}\right\}_{i \in \mathcal{N}\left(I_{j}\right)}\right) \prod_{i \in \mathcal{N}\left(I_{j}\right)} \gamma_{i \rightarrow I_{j}}\left(\hat{x}_{i}^{(0)}\right)\right]
$$

For instance, the message from interaction node $I_{1}$ to individual 1 at time 1 is given by

$$
\begin{align*}
\delta_{I_{1} \rightarrow 1}\left(\hat{x}_{1}^{(1)}\right)=\sum_{\hat{x}_{1}^{(0)}, \hat{x}_{3}^{(0)}, \hat{x}_{5}^{(0)} \in\{0,1\}}[ & {\left[P\left(x_{1}^{(1)}=\hat{x}_{1}^{(1)} \mid x_{1}^{(0)}=\hat{x}_{1}^{(0)}, x_{3}^{(0)}=\hat{x}_{3}^{(0)}, x_{5}^{(0)}=\hat{x}_{5}^{(0)}\right)\right.} \\
& \left.\times \gamma_{1 \rightarrow I_{1}}\left(\hat{x}_{1}^{(0)}\right) \gamma_{3 \rightarrow I_{1}}\left(\hat{x}_{3}^{(0)}\right) \gamma_{5 \rightarrow I_{1}}\left(\hat{x}_{5}^{(0)}\right)\right] . \tag{6.18}
\end{align*}
$$

We first consider the case that $\hat{x}_{1}^{(1)}=0$, and form Table 6.1. It is easy to see that (6.18) can be expanded into the following

$$
\begin{aligned}
\delta_{I_{1} \rightarrow 1}(0) & =\gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(0)+(1-q) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1) \\
& +(1-q) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(0)+(1-q)^{2} \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1)
\end{aligned}
$$

| $\hat{x}_{1}^{(0)}$ | $\hat{x}_{3}^{(0)}$ | $\hat{x}_{5}^{(0)}$ | $P\left(x_{1}^{(1)}=0\right.$ | $\left.x_{1}^{(0)}=\hat{x}_{1}^{(0)}, x_{3}^{(0)}=\hat{x}_{3}^{(0)}, x_{5}^{(0)}=\hat{x}_{5}^{(0)}\right)$ |
| :---: | :---: | :---: | :---: | :---: |
| 0 | 0 | 0 | 1 |  |
| 0 | 0 | 1 | $1-q$ |  |
| 0 | 1 | 0 | $1-q$ |  |
| 0 | 1 | 1 | $(1-q)^{2}$ |  |
| 1 | 0 | 0 | 0 |  |
| 1 | 0 | 1 | 0 |  |
| 1 | 1 | 0 | 0 |  |
| 1 | 1 | 1 | 0 |  |

Table 6.1: The conditional probability that individual 1 at time 1 is not infected, given the status of individuals 1,3 , and 5 at time 0 .
where it can be simplified using the fact that messages $\gamma_{i \rightarrow I_{j}}$ are probability distributions on $\{0,1\}$, i.e., $\mu_{i \rightarrow I_{j}}(0)+\mu_{i \rightarrow I_{j}}(1)=1$.

$$
\delta_{I_{1} \rightarrow 1}(0)=\gamma_{1 \rightarrow I_{1}}(0)\left(1-q \gamma_{3 \rightarrow I_{1}}(1)\right)\left(1-q \gamma_{5 \rightarrow I_{1}}(1)\right)
$$

We now consider the case that $\hat{x}_{1}^{(1)}=1$. Expansion of (6.18) results in

$$
\begin{aligned}
\delta_{I_{1} \rightarrow 1}(1) & =q \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1)+q \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(0) \\
& +\left(1-(1-q)^{2}\right) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1)+\gamma_{1 \rightarrow I_{1}}(1) \gamma_{3 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(0) \\
& +\gamma_{1 \rightarrow I_{1}}(1) \gamma_{3 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1)+q \gamma_{1 \rightarrow I_{1}}(1) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(0) \\
& +q \gamma_{1 \rightarrow I_{1}}(1) \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1),
\end{aligned}
$$

where we can simplify it to

$$
\delta_{I_{1} \rightarrow 1}(1)=\gamma_{1 \rightarrow I_{1}}(1)-\gamma_{1 \rightarrow I_{1}}(0)\left(-q \gamma_{3 \rightarrow I_{1}}(1)-q \gamma_{5 \rightarrow I_{1}}(1)+q^{2} \gamma_{3 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1)\right)
$$

The messages from interaction nodes to individuals at time 0 are computed as follows.

The message from interaction node $I_{j}$ to individual $i$ at time 0 is computed using

$$
\begin{aligned}
& \gamma_{I_{j} \rightarrow i}\left(\hat{x}_{i}^{(0)}\right)=\sum_{\hat{x}_{j}^{(1)}, \hat{x}_{i^{\prime}}^{(0)} \in\{0,1\}}\left[P\left(x_{j}^{(1)}=\hat{x}_{j}^{(1)} \mid x_{i}^{(0)}=\hat{x}_{i}^{(0)},\left\{x_{i^{\prime}}^{(0)}=\hat{x}_{i^{\prime}}^{(0)}\right\}_{i^{\prime} \in \mathcal{N}\left(I_{j}\right) \backslash\{i\}}\right)\right. \\
&\left.\times \delta_{j \rightarrow I_{j}}\left(\hat{x}_{j}^{(1)}\right) \prod_{i^{\prime} \in \mathcal{N}\left(I_{j}\right) \backslash\{i\}} \gamma_{i^{\prime} \rightarrow I_{j}}\left(\hat{x}_{i^{\prime}}^{(0)}\right)\right] .
\end{aligned}
$$

For instance, the message from interaction node $I_{1}$ to individual 3 at time 0 is given by

$$
\begin{align*}
\gamma_{I_{1} \rightarrow 3}\left(\hat{x}_{3}^{(0)}\right)=\sum_{\hat{x}_{1}^{(1)}, \hat{x}_{1}^{(0)}, \hat{x}_{5}^{(0)} \in\{0,1\}} & {\left[P\left(x_{1}^{(1)}=\hat{x}_{1}^{(1)} \mid x_{1}^{(0)}=\hat{x}_{1}^{(0)}, x_{3}^{(0)}=\hat{x}_{3}^{(0)}, x_{5}^{(0)}=\hat{x}_{5}^{(0)}\right)\right.} \\
& \left.\times \delta_{1 \rightarrow I_{1}}\left(\hat{x}_{1}^{(1)}\right) \gamma_{1 \rightarrow I_{1}}\left(\hat{x}_{1}^{(0)}\right) \gamma_{5 \rightarrow I_{1}}\left(\hat{x}_{5}^{(0)}\right)\right] . \tag{6.19}
\end{align*}
$$

First, we consider the case that $\hat{x}_{3}^{(0)}=0$, and expand (6.19) as follows.

$$
\begin{aligned}
\gamma_{I_{1} \rightarrow 3}(0) & =\delta_{1 \rightarrow I_{1}}(0) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(0)+(1-q) \delta_{1 \rightarrow I_{1}}(0) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1) \\
& +q \delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1)+\delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(0) \\
& +\delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1),
\end{aligned}
$$

where after simplification becomes

$$
\begin{aligned}
\gamma_{I_{1} \rightarrow 3}(0) & =\gamma_{1 \rightarrow I_{1}}(1) \delta_{1 \rightarrow I_{1}}(1)+\gamma_{1 \rightarrow I_{1}}(0) \delta_{1 \rightarrow I_{1}}(0)\left(1-q \gamma_{5 \rightarrow I_{1}}(0)\right) \\
& -\gamma_{1 \rightarrow I_{1}}(0) \delta_{1 \rightarrow I_{1}}(1)\left(-q \gamma_{5 \rightarrow I_{1}}(1)\right)
\end{aligned}
$$

Then, we consider the case that $\hat{x}_{3}^{(0)}=1$.

$$
\begin{aligned}
\gamma_{I_{1} \rightarrow 3}(1) & =(1-q) \delta_{1 \rightarrow I_{1}}(0) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(0)+(1-q)^{2} \delta_{1 \rightarrow I_{1}}(0) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1) \\
& +q \delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(0)+\left(1-(1-q)^{2}\right) \delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(0) \gamma_{5 \rightarrow I_{1}}(1) \\
& +\delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(0)+\delta_{1 \rightarrow I_{1}}(1) \gamma_{1 \rightarrow I_{1}}(1) \gamma_{5 \rightarrow I_{1}}(1)
\end{aligned}
$$

where can be simplified to

$$
\begin{aligned}
\gamma_{I_{1} \rightarrow 3}(1) & =\gamma_{1 \rightarrow I_{1}}(1) \delta_{1 \rightarrow I_{1}}(1)+(1-q) \gamma_{1 \rightarrow I_{1}}(0) \delta_{1 \rightarrow I_{1}}(0)\left(1-q \gamma_{5 \rightarrow I_{1}}(0)\right) \\
& +\gamma_{1 \rightarrow I_{1}}(0) \delta_{1 \rightarrow I_{1}}(1)\left[q-(1-q)\left(-q \gamma_{5 \rightarrow I_{1}}(1)\right)\right]
\end{aligned}
$$

### 6.4 Simulation Results

In this section, we compare the performance of the BP using Initial Prior probabilities (BPIP) algorithm, the BP using Updated Prior probabilities (BPUP) algorithm, and the BP on Combined Graphs (BPCG) algorithm using three metrics, success probability, FNR, and FPR. Each result is averaged over 1000 experiments. The testing matrix is constructed according to a Bernoulli design with parameters $\nu=\ln 2$. In the BPUP algorithm, the updated prior probabilities given by (6.1) are computed using the contact tracing information. In our simulations, we assume that individual $i$ at time 0 , for each $i \in[N]$, interacts with individual $j$ at time 0 , for each $j \in[N] \backslash\{i\}$, with some fixed probability $\theta$, referred to as interaction probability. It should be noted that $d_{i}$, the number of individuals which interact with individual $i$, follows a binomial distribution with parameters $N-1$ and $\theta$, i.e., $d_{i} \sim B(N-1, \theta)$. The expected value of $x_{i}^{(1)}$ is computed as follows:

$$
\mathbb{E}\left[x_{i}^{(1)}\right]=\mathbb{E}\left[\mathbb{E}\left[x_{i}^{(1)} \mid d_{i}\right]\right]=\mathbb{E}\left[1-(1-p)(1-p q)^{d_{i}}\right]=1-(1-p) \mathbb{E}\left[(1-p q)^{d_{i}}\right]
$$



Figure 6.4: Success probability as a function of the number of tests $M$ based on simulation results for $N=500$ individuals, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, and the interaction probability $\theta=0.008$, under the binary symmetric noise model with parameter $\rho \in\{0.01,0.05\}$.
where the term $\mathbb{E}\left[(1-p q)^{d_{i}}\right]$ is given by

$$
\mathbb{E}\left[(1-p q)^{d_{i}}\right]=\sum_{d=0}^{N-1}(1-p q)^{d}\binom{N-1}{d} \theta^{d}(1-\theta)^{N-d-1}=(1-p q \theta)^{N-1}
$$

Thus, we have $\mathbb{E}\left[x_{i}^{(1)}\right]=1-(1-p)(1-p q \theta)^{N-1}$. Accordingly, the average number of infected individuals for the BPUP and the BPCG algorithms is given by

$$
K=N\left(1-(1-p)(1-p q \theta)^{N-1}\right)
$$

In Fig. 6.4, we plot success probability as a function of the number of tests $M$ based on simulation results for $N=500$ individuals, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, and the interaction probability $\theta=0.008$, under the binary symmetric noise model with parameter $\rho \in\{0.01,0.05\}$. The value of success probability for each number of test is optimized over the threshold in the range $\tau \in[-10,10]$. The number of


Figure 6.5: FNR vs. FPR based on simulation results for threshold $\tau \in[-10,10], N=$ 500 individuals, $M=350$ tests, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, and the interaction probability $\theta=0.008$, under the binary symmetric noise model with parameter $\rho \in\{0.01,0.05\}$.
iterations for the BPUP and the BPIP algorithms is $T=15$. We consider $T=30$ iterations for the BPCG algorithm. It can be observed that the BPCG algorithm outperforms the other algorithms for all values of $M$. For instance, when $\rho=0.01$ and the number of test is $M=350$, the BPCG algorithm provides a success probability $4 \%$ and $24 \%$ greater than that of the BPUP and the BPIP algorithms, respectively. Also, it can be seen that for the high noise regime, i.e., $\rho=0.05$, the advantage of BPCG algorithm over the other algorithms in terms of success probability becomes more evident. For example, for $\rho=0.05$ and $M=350$ tests, the success probability of the BPCG algorithm is $7.4 \%$ and $43.5 \%$ greater than that of the BPUP and the BPIP algorithms, respectively.

In Fig. 6.5, we depict the FNR vs. FPR for all three decoding algorithms for threshold $\tau \in[-10,10], N=500$ individuals, $M=300$ tests, the prevalence rate $p=0.01$, the contagion probability $q=0.1$, the interaction probability $\theta=0.008$, and the noise parameter $\rho=0.01$. A point on a curve corresponding to a decoding algorithm represents the pair (FNR,FPR) which has been computed for the same value of $\tau$. The closer a curve
is to the origin of the FNR-FPR plane, the better the performance of the corresponding decoding algorithm in terms of FNR and FPR. It can be observed that for the BPCG algorithm the operating point that minimizes the total error rate, i.e., the sum of FPR and FNR is closer to the origin than that of the BPUP and the BPIP algorithms.

## 7. SINGLE-SERVER SINGLE-MESSAGE ONLINE PRIVATE INFORMATION RETRIEVAL WITH SIDE INFORMATION*

### 7.1 Introduction

In this chapter, we study the problem of single-server online Private Information Retrieval (PIR) with side information. The goal of the PIR schemes [24] is to enable a user to download a message or a set of messages belonging to a database whose copies are stored on a single or multiple remote servers, without revealing which message it is requesting. In a single server scenario, the entire database needs to be downloaded to preserve the privacy of the requested message. However, when the user has some side information about the database [25-33], the information-theoretic privacy can be achieved more efficiently than downloading the whole database.

In the PIR with side information setting, the user has access to a random subset of the messages in the database as side information, which are unknown to the server. This side information could have been obtained from other trusted users or through previous interactions with the server. In this setting, the savings in the download cost depend on whether the user wants to protect only the privacy of the requested message, or the privacy of both the requested message and the messages in the side information.

To the best of our knowledge, all of the prior works on PIR focus on retrieval of a single or multiple messages at once. However, in many practical settings, the user needs to retrieve multiple messages periodically, over multiple rounds. For example, a user might retrieve a book or a movie from an on-line repository on a daily basis. We refer to this setting as online PIR, inspired by the fact that the user does not know the identities

[^4]of the future items that need to be retrieved from the server. The key requirement in such scenarios is to protect the identity of all the requested messages up to the current round. By leveraging previously downloaded messages, the user can significantly reduce the number of bits that need to be downloaded. Accordingly, we analyze both the fundamental limits as well as the achievability schemes for the online PIR schemes.

### 7.1.1 Main Contributions

In this study, we consider the problem of single-server online PIR with side information. In this problem, there is a user who wishes to download a sequence of messages $\mathcal{X}_{W}=\left\{X_{W_{1}}, X_{W_{2}}, \ldots, X_{W_{t}}\right\}$ from a database $\mathcal{X}$ of $K$ messages, stored on a single server. The communication is performed in rounds, such that at round $i$, the user wishes to retrieve a message $X_{W_{i}}$ for some $W_{i} \in[K]$. We assume that the user decides on which message $W_{i}$ to request at round $i$ at the beginning of that round and that the identity of the future messages $W_{j}, j>i$ are not known at that time. We also assume that at the beginning of the first round the user has access to $M$ messages which are selected uniformly at random from the database. The identity of these $M$ messages are not known to the server.

We focus on the scenario where at round $i$, the user wishes to protect the identity of all the requested messages individually up to round $i,\left\{W_{1}, \ldots, W_{i}\right\}$ for $1 \leq i \leq t$. That is, after the user makes a request to the server at round $i$, the server cannot decide which of the $K$ messages is more likely to get requested at that round and at the previous rounds. Focusing on scalar-linear settings, we characterize the per-round capacity, i.e., the maximum achievable download rate at each round. Note that the tightness of the scalar-linear capacity for general schemes is still open. We also present a scalar-linear coding scheme that achieves this capacity. The key idea of our scheme is to combine the data downloaded during the current round and the previous rounds, with the original side information (unknown to server) so as to construct new side information for the subsequent rounds. We
show that for the setting with $K$ messages stored at the server and a random subset of $M$ messages available to the user at the first round, the per-round capacity of the scalar-linear scheme is $C_{1}=(M+1) / K$ for the first round and $C_{i}=\left(2^{i-1}(M+1)\right) / K M$ for round $i \geq 2$, provided that $K /(M+1)=2^{l}$ for some $l \geq 1$. The generalization of these results for the cases in which $K /(M+1)$ is not a power of 2 is not straightforward, and remains an open problem.

### 7.1.2 Related Work

The classical PIR problem with multiple servers each of which stores the full copy of the database, has been extensively studied [81, 82]. The most relevant to our study is the line of work that focuses on setting with multiple retrieved messages $[31,83,84]$ as well as settings in which the user access to certain files as side information before the information retrieval process begins. The side information settings have been studied in [25,26] for the single server setting and in [27-31] for the multi-server setting.

Kadhe et al. [25] initiated the study of the single-server single-message PIR with side information. References [30] and [31] studied the multi-server scenario where the user wants to protect the privacy of both the requested message(s) and the messages in the side information, for the single-message and the multi-message PIR problems, respectively. Another notion of privacy, termed individual privacy, was also recently introduced in [33] for the multi-user setting of PIR with side information. Recently, the settings in which the side information is a linear combination of a subset of messages was studied in [26,32].

To the best our knowledge, none of the prior works on the private information retrieval focused on the online settings in which the requests are issued one at a time such that the identities of future requests are unknown.

### 7.2 Problem Formulation and Main Results

Throughout, we denote random variable and their realizations by bold-face letters and regular letters, respectively. For a positive integer $i$, denote $[i] \triangleq\{1, \ldots, i\}$. Let $\mathbb{F}_{q}$ be a finite field for some prime $q$, and $\mathbb{F}_{q^{m}}$ be an extension field of $\mathbb{F}_{q}$ for some integer $m \geq 1$. We assume that there is a server storing a set $\mathcal{X}$ of $K$ messages, $\mathcal{X} \triangleq$ $\left\{X_{1}, \ldots, X_{K}\right\}$, with each message $X_{i}$ being independently and uniformly distributed over $\mathbb{F}_{q^{m}}$, i.e., $H\left(X_{1}\right)=\cdots=H\left(X_{K}\right)=L$ and $H\left(X_{1}, \ldots, X_{K}\right)=K L$, where $L \triangleq m \log _{2} q$. We assume that there is a user that wishes to retrieve a sequence of messages $\mathcal{X}_{W}=$ $\left\{X_{W_{1}}, X_{W_{2}}, \ldots, X_{W_{t}}\right\}$ from the server so that at round $i$, the user wishes to retrieve the message $X_{W_{i}}$ for some $W_{i} \in[K]$. We assume that the identity of the index $W_{i}$ of the message retrieved at round $i$ is not known to the user before round $i$. We also assume that initially the user knows a random subset $\mathcal{X}_{\mathcal{S}}$ of $\mathcal{X}$ that includes $M$ messages for some $\mathcal{S} \subset[K],|\mathcal{S}|=M$. We refer to $W_{i}$ as the demand index at round $i, X_{W_{i}}$ as the demand at round $i, \mathcal{S}$ as the side information index set, $\mathcal{X}_{\mathcal{S}}$ as the side information set and $M$ as the size of the side information set.

Let $\boldsymbol{S}$ and $\boldsymbol{W}_{i}$ be random variables corresponding to $\mathcal{S}$ and $W_{i}$, respectively. Denote the probability mass function (pmf) of $\boldsymbol{S}$ by $p_{\boldsymbol{S}}(\cdot)$, and the conditional pmf of $\boldsymbol{W}_{i}$ given $\boldsymbol{S}$ by $p_{\boldsymbol{W}_{i} \mid \boldsymbol{S}}(\cdot \mid \cdot)$. We assume that $\mathcal{S}$ is uniformly distributed over all subsets of $[K]$ of size $M$, i.e., $p_{\boldsymbol{S}}(\mathcal{S})=\binom{K}{M}^{-1}$ for all $\mathcal{S} \subset[K],|\mathcal{S}|=M$; and $\boldsymbol{W}_{i}$ 's are independent and uniformly distributed over $[K] \backslash \mathcal{S}$, i.e.,

$$
p_{\boldsymbol{W}_{i} \mid \boldsymbol{S}}\left(W_{i} \mid \mathcal{S}\right)= \begin{cases}\frac{1}{K-M}, & W_{i} \notin \mathcal{S} \\ 0, & \text { otherwise }\end{cases}
$$

Also, we assume that the server knows the size of $\mathcal{S}$ (i.e., $M$ ), the $\operatorname{pmf} p_{\boldsymbol{S}}($.$) and$ $p_{\boldsymbol{W}_{i} \mid \boldsymbol{S}}(. \mid$.$) , but the realizations \mathcal{S}$ and $W_{i}$ are unknown to the server before round $i$.

At round $i$ in order to retrieve $X_{W_{i}}$, the user sends to the server a query $Q^{\left[W_{i}, \mathcal{S}\right]}$, and upon receiving $Q^{\left[W_{i}, \mathcal{S}\right]}$, the server sends to the user an answer $A^{\left[W_{i}, \mathcal{S}\right]}$. We define $Q^{\left[W_{1: i}, \mathcal{S}\right]} \triangleq\left\{Q^{\left[W_{1}, \mathcal{S}\right]}, Q^{\left[W_{2}, \mathcal{S}\right]}, \ldots, Q^{\left[W_{i}, \mathcal{S}\right]}\right\}$ and $A^{\left[W_{1: i}, \mathcal{S}\right]} \triangleq\left\{A^{\left[W_{1}, \mathcal{S}\right]}, A^{\left[W_{2}, \mathcal{S}\right]}, \ldots, A^{\left[W_{i}, \mathcal{S}\right]}\right\}$ as the sets of all queries and answers up to the round $i$, respectively.

Note that the query $Q^{\left[W_{i}, \mathcal{S}\right]}$ at round $i$ is a (potentially stochastic) function of $W_{i}, \mathcal{S}, X_{\mathcal{S}}, Q^{\left[W_{1: i-1}, \mathcal{S}\right]}$, and $A^{\left[W_{1: i-1}, \mathcal{S}\right]}$. We assume that the answer at round $i, A^{\left[W_{i}, \mathcal{S}\right]}$ is a (deterministic) function of $Q^{\left[W_{1: i}, \mathcal{S}\right]}$ and the messages in $\mathcal{X}$, i.e,

$$
H\left(\boldsymbol{A}^{\left[\boldsymbol{W}_{i}, \boldsymbol{\mathcal { S }}\right]} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}, \mathcal{X}\right)=0
$$

The queries $Q^{\left[W_{1: i}, \mathcal{S}\right]}$ from the first round up to round $i$ all together must protect the privacy of every demand index up to round $i$ individually (not jointly) from the server, i.e.,

$$
\mathbb{P}\left(\boldsymbol{W}_{j}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \mathcal{X}=\mathcal{X}\right)=\frac{1}{K}
$$

for all $W^{\prime} \in[K]$ and all $j \in[i]$. This means that some correlations between the demand indices of different rounds (or correlations between the demands and the side information) can be revealed to the server, but every demand index up to round $i$ must be kept private individually at each round. This condition is referred to as the privacy condition.

All the answers from the first round up to round $i, A^{\left[W_{1: i}, \mathcal{S}\right]}$ along with the side information $\mathcal{X}_{\mathcal{S}}$ must enable the user to retrieve the demand $X_{W_{i}}$. This condition is referred to as the recoverability condition, as follows:

$$
H\left(\boldsymbol{X}_{\boldsymbol{W}_{i}} \mid \boldsymbol{A}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}, \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}, \boldsymbol{\mathcal { X }}_{\mathcal{S}}, \boldsymbol{W}_{1}, \ldots, \boldsymbol{W}_{i}, \boldsymbol{S}\right)=0
$$

The problem of the single-server Online Private Information Retrieval (OPIR) is to design a protocol that at round $i \geq 1$, constructs a query $Q^{\left[W_{i}, \mathcal{S}\right]}$ for any given $\mathcal{S}$ and $W_{i}$,
and the corresponding answer $A^{\left[W_{i}, \mathcal{S}\right]}$ that satisfy the privacy and recoverability conditions.
The per-round rate of an OPIR algorithm at round $i$ denoted by $R_{i}$, is defined as the ratio of the entropy of a message, i.e., $L$, to the maximum entropy of the answer at round i, i.e.,

$$
R_{i}=\min _{W_{1}, \ldots, W_{i}, \mathcal{S}} \frac{L}{H\left(\boldsymbol{A}^{\left[W_{i}, \mathcal{S}\right]}\right)},
$$

where the minimum is taken over all possible realizations $W_{1}, \ldots, W_{i}$ and $\mathcal{S}$.
The per-round capacity of $O P I R$ at round $i$ denoted by $C_{i}$, is defined as the supremum of rates over all OPIR algorithms that achieve the capacity up to round $i-1$.

In this work, we focus on scalar-linear (per-round) capacity, which corresponds to the maximum (per-round) rate that can be achieved by scalar-linear schemes.

In scalar-linear schemes, the answer $A^{\left[W_{i}, \mathcal{S}\right]}$ at round $i$ is a set of $m_{i}$ messages, i.e., $A^{\left[W_{i}, \mathcal{S}\right]} \triangleq\left\{y_{i, 1}, \ldots, y_{i, m_{i}}\right\}$. Each message $y_{i, j}$ for $1 \leq j \leq m_{i}$ is a scalar linear combination of the original messages in $\mathcal{X}$, i.e. $y_{i, j}=\sum_{m=1}^{K} \gamma_{i, j}^{m} X_{m}$, where $\gamma_{i, j}^{m} \in \mathbb{F}_{q}$ are the encoding coefficients of $y_{i, j}$. We refer to the vector $\gamma_{i, j}=\left[\gamma_{i, j}^{1}, \gamma_{i, j}^{2}, \ldots, \gamma_{i, j}^{K}\right]$ as the encoding vector of $y_{i, j}$. The $i$-th unit encoding vector that corresponds to the original packet $X_{i}$ is denoted by $u_{i}=\left[u_{i}^{1}, u_{i}^{2}, \ldots, u_{i}^{K}\right]$, where $u_{i}^{i}=1$ and $u_{i}^{j}=0$ for $i \neq j$. Consider the set of $K$ linearly independent unit vectors $\left\{u_{1}, u_{2}, \ldots, u_{K}\right\}$ as a basis of a vector space $\mathcal{V}$ of dimension $K$. Then, the encoding vector of $y_{i, j}$, i.e., $\gamma_{i, j}$, is a vector in $\mathcal{V}$. We also define the answer matrix at round $i, A_{i}$, of dimension $\left(m_{i} \times K\right)$ with $\gamma_{i, j}$ being the $j$-th row of $A_{i}$. Note that the entropy $H\left(\boldsymbol{A}^{\left[W_{i}, \mathcal{S}\right]}\right)$ of the answer is proportional to number of messages in $A^{\left[W_{i}, \mathcal{S}\right]}$, or equivalently, the number of rows of matrix $A_{i}$.

The goal is to establish the scalar-linear per-round capacity of OPIR, and present an algorithm that achieves this capacity. Theorem 10 characterizes the capacity of scalarlinear OPIR problem for the case when $K /(M+1)$ is a power of 2 . It should be noted that the tightness of the scalar-linear capacity for general (vector-linear and non-linear)
schemes remains an open problem.

Theorem 10. For the OPIR problem with $K$ messages, and side information size $M$, when $K /(M+1)$ is a power of 2 , the scalar-linear per-round capacity at round $i$ is given by:

$$
C_{i}= \begin{cases}\frac{M+1}{K} & i=1 \\ \frac{2^{i-1}(M+1)}{K M} & i \geq 2\end{cases}
$$

### 7.3 Converse Proof

In this section, we prove the converse part of Theorem 10. Suppose that the user wishes to retrieve a sequence of messages $\mathcal{X}_{W}=\left\{X_{W_{1}}, X_{W_{2}}, \ldots, X_{W_{t}}\right\}$ from the server so that at round $i$, the user wants to download the message $X_{W_{i}}$ for some $W_{i} \in[K]$, and knows $\mathcal{X}_{\mathcal{S}}$ for a given $\mathcal{S} \subseteq[K] \backslash W,|\mathcal{S}|=M$. By assumption, $K /(M+1)$ is a power of 2 . At round $i$, for any $\mathcal{S}$ and $W_{i}$, in order to retrieve $X_{W_{i}}$, the user sends to the server a query $Q^{\left[W_{i}, \mathcal{S}\right]}$, and the server responds to the user by an answer $A^{\left[W_{i}, \mathcal{S}\right]}$.

For the first round $(i=1)$, the proof of converse follows from the prior results for PIR with side information (see [25, Lemma 1]). It is easy to verify that at round 1, any optimal scalar-linear scheme can be converted to the partition-based scheme of [25] by row operations. The answer matrix $A_{1}$ corresponding to the optimal scheme has exactly $K /(M+1)$ rows. Followed by a column permutation, the matrix $A_{1}$ can be represented as:

$$
A_{1}=\left[\begin{array}{cccc}
\overbrace{*}^{*} \quad \ldots & * & \overbrace{3}^{M+1} & \\
& * \cdots & * & \\
& & \ddots & \\
& & & \overbrace{* \cdots}^{M+1}
\end{array}\right]_{\frac{K}{M+1} \times K}^{M+1}
$$

where $*$ 's indicate non-zero entries in matrix $A_{1}$ and all other entries in matrix $A_{1}$ are zero. Each row of $A_{1}$ corresponds to one of the messages in the answer. For instance, the first row corresponds to $X_{1}+\cdots+X_{M+1}$, the second row corresponds to $X_{M+2}+\cdots+X_{2 M+2}$, and so on. The support set of each message in the answer is called a partition set. Thus, the optimal scheme in the first round has $n=K /(M+1)$ partition sets, denoted by $\left\{P_{1}, P_{2}, \ldots, P_{n}\right\}$.

In Theorem 11 below we prove that for round $i \geq 2$, the maximum entropy of the answer, i.e., $H\left(A^{\left[W_{i}, \mathcal{S}\right]}\right)$, where the maximum is taken over all $W_{i}$ and $\mathcal{S}$, is lower bounded by $K M /\left(2^{i-1}(M+1)\right)$.

Theorem 11. The maximum entropy of the answer $H\left(A^{\left[W_{i}, \mathcal{S}\right]}\right)$ at round $i \geq 2$ over all $W_{i}$ and $\mathcal{S}$, is lower bounded by $K M /\left(2^{i-1}(M+1)\right)$.

Proof. For linear schemes it is sufficient to prove that the maximum number of rows of matrix $A_{i}$ for $i \geq 2$ is lower bounded by $K M /\left(2^{i-1}(M+1)\right)$. The proof is based on an inductive argument and uses a simple yet powerful observation, formally stated in Lemma 14.

Lemma 14. For any $i, W_{1: i}, \mathcal{S}$, and any $W^{*} \in[K]$, there must exist $\mathcal{S}^{*}$ and $P_{j}$ (for some $j \in[n]$ ) such that: (i) $\mathcal{S}^{*} \subseteq P_{j}$, (ii) $\left|\mathcal{S}^{*}\right|=M$, and (iii) $W^{*} \notin P_{j}$ and it holds that

$$
H\left(X_{W^{*}} \mid A^{\left[W_{1: i}, \mathcal{S}\right]}, Q^{\left[W_{1: i}, \mathcal{S}\right]}, \mathcal{X}_{\mathcal{S}^{*}}\right)=0
$$

Proof. Assume, by the way of contradiction, that there does not exist any $\mathcal{S}^{\star}$ such that $X_{W^{\star}}$ is recoverable from $A^{\left[W_{1: i}, \mathcal{S}\right]}$ and $\mathcal{X}_{\mathcal{S}^{\star}}$, then the server knows that $W^{\star}$ cannot be the user's demand index, and this violates the privacy condition. Now, assume that there exist some $\mathcal{S}^{\star}$ that $X_{W^{\star}}$ is recoverable from $A^{\left[W_{1: i}, \mathcal{S}\right]}, Q^{\left[W_{1: i}, \mathcal{S}\right]}$ and $\mathcal{X}_{\mathcal{S}^{\star}}$. Given the optimal scheme in the first round, if all such $\mathcal{S}^{\star}$ 's do not satisfy the conditions of the lemma, then
the server knows that none of these $\mathcal{S}^{\star}$ 's can be the user's real side information index set. Thus, the server realizes that $W^{\star}$ cannot be the user's demand index, and this violates the privacy condition.

Lemma 15. At round $i$ for $i \geq 2$, in the vector space spanned by the rows of matrices $A_{1}, A_{2}, \ldots, A_{i}$, corresponding to any $W^{*} \in[K]$, there must exist a vector which is a linear combination of at most $M+1$ messages including $\mathcal{X}_{W^{*}}$ itself and at most $M$ other messages which are a subset of $\mathcal{X}_{\mathcal{S}^{*}}$, a potential side information for $\mathcal{X}_{W^{*}}$ defined in Lemma 14.

Proof. The proof is based on contradiction. Assume that at round $i$, for $i \geq 2$, in the vector space spanned by the rows of matrices $A_{1}, A_{2}, \ldots, A_{i}$, for a given $W^{*} \in[K]$, there does not exist such a vector described in Lemma 15. This means that $X_{W^{*}}$ is not recoverable from $A^{\left[W_{1: i}, \mathcal{S}\right]}$ and $\mathcal{X}_{\mathcal{S}^{*}}$, which contradicts the result of Lemma 14.

In fact, in the vector space spanned by the rows of $A_{1}, A_{2}, \ldots, A_{i}$, there must exist $K$ of such vectors, one for each potential value of $W^{*} \in[K]$. Define matrix $\Gamma$ with these $K$ vectors being as the rows of $\Gamma$. An instance of matrix $\Gamma$ would be as follows:


Lemma 16. The rank of matrix $\Gamma$ is lower bounded by $K / 2$.

Proof. Since by Lemma $14, \mathcal{S}^{*} \subseteq P_{j}$ for some $j \in[n]$ and $W^{*} \nsubseteq P_{j}$, then $\mathcal{S}^{*}$ is either in the left side of $W^{*}$, or in the right side of $W^{*}$ in each row of matrix $\Gamma$. Accordingly, the rows of matrix $\Gamma$ can be classified into two types: L and R , based on the criteria that $\mathcal{S}^{*}$ is in the left side of $W^{*}$, or in the right side of $W^{*}$, respectively. Let $z_{1}$ and $z_{2}$ denote the number of rows of type L and the number of rows of type R, respectively. Note that the maximum of $z_{1}$ and $z_{2}$ is greater than or equal to $K / 2$, i.e., $\max \left(z_{1}, z_{2}\right) \geq K / 2$. Without loss of generality, assume $\max \left(z_{1}, z_{2}\right)=z_{1}$. Then, $z_{1} \geq K / 2$. By removing $z_{2}$ rows of type R from matrix $\Gamma$, we are left with $z_{1} \geq K / 2$ rows of type L that constitute a matrix of size $z_{1} \times K$, in which there exists a lower triangular submatrix of size $z_{1} \times z_{1}$ and rank $z_{1} \geq K / 2$. Thus, the rank of matrix $\Gamma$ is at least $z_{1}$ which is lower bounded by $K / 2$.

For the second round ( $i=2$ ), we need to show that the number of rows of matrix $A_{2}$ is lower bounded by $K M /(2(M+1))$. Based on Lemma 15, in the vector space spanned by the rows of matrices $A_{1}$ and $A_{2}$, there must exist all $K$ rows of matrix $\Gamma$ which based on Lemma 16 is of rank greater than or equal to $K / 2$. On the other hand, as mentioned earlier, the optimal scheme in the first round is partitioning where each row of $A_{1}$ corresponds to a linear combination of $M+1$ messages. One can readily confirm that corresponding to any $M+1$ number of linearly independent rows of matrix $\Gamma$, there exists at most one linear combination of these rows in the span of the rows of matrix $A_{1}$. Thus, there must exist at least $M$ linearly independent combinations of these rows in the span of the rows of matrix $A_{2}$. Then, we have:

$$
\operatorname{rank}\left(A_{2}\right) \geq \frac{M}{M+1} \times \operatorname{rank}(\Gamma) \geq \frac{M}{M+1} \times \frac{K}{2}=\frac{K M}{2(M+1)}
$$

In other words, matrix $\Gamma$ has at least $K / 2$ linearly independent rows. Thus, there exist at most $K /(2(M+1))$ linearly independent combinations of these rows in the rows of matrix $A_{1}$. Therefore, there must exist at least $K / 2-K /(2(M+1))=K M /(2(M+1))$
linearly independent combinations of these rows in the rows of matrix $A_{2}$, which indicates that the number of rows of matrix $A_{2}$ is lower bounded by $K M /(2(M+1))$. The optimal scheme achieves the lower bound. Thus, in the optimal scheme, the number of rows of matrix $A_{2}$ is exactly $K M /(2(M+1))$.

For the third round $(i=3)$, it suffices to show that the number of rows of matrix $A_{3}$ is lower bounded by $K M /(4(M+1))$. By the result of Lemma 15 , the vector space spanned by the rows of matrices $A_{1}, A_{2}, A_{3}$ contains all $K$ rows of matrix $\Gamma$, which itself has rank greater than or equal to $K / 2$ (by Lemma 16). Similarly as in the case of $i=2$, there exist at most two linearly independent combinations of these rows among the rows of matrix $A_{1}$, corresponding to any $2(M+1)$ number of linearly independent rows of matrix $\Gamma$. On the other hand, as shown earlier, in an optimal scheme, the rank of $A_{2}$ is given by $K M /(2(M+1))$. This shows that there exist at most $M$ linearly independent combinations of these rows among the rows of matrix $A_{2}$, corresponding to any $2(M+1)$ number of linearly independent rows of matrix $\Gamma$. Thus, there must exist at least $2(M+$ 1) $-2-M=M$ linearly independent combinations of these rows among the rows of matrix $A_{3}$. Thus, we have:

$$
\operatorname{rank}\left(A_{3}\right) \geq \frac{M}{2(M+1)} \times \operatorname{rank}(\Gamma) \geq \frac{K M}{4(M+1)}
$$

Thus, there must exist at least $K / 2-K /(2(M+1))-K M /(4(M+1))$, or equivalently, $K M /(4(M+1))$ linearly independent combinations of these rows among the rows of matrix $A_{3}$. This further indicates that the number of rows of matrix $A_{3}$ is lower bounded by $K M /(4(M+1))$.

Using the same proof technique and similar reasoning as in the cases of $i=2$ and $i=3$, it can be shown that the number of rows of $A_{i}$ for $i \geq 2$ is lower bounded by $K M /\left(2^{i-1}(M+1)\right)$. By the result of Lemma 15, in the vector space spanned by the
rows of matrices $A_{1}, A_{2}, \ldots, A_{i}$, there must exist all $K$ rows of matrix $\Gamma$, which is of rank greater than or equal to $K / 2$ (by Lemma 16). Again, similarly as in the cases of $i=2$ and $i=3$, it follows that corresponding to any $2^{i-2}(M+1)$ number of linearly independent rows of matrix $\Gamma$, there exist at most $2^{i-2}, 2^{i-3} M, 2^{i-4} M, \ldots, M$, linearly independent combinations of these rows in the rows of matrix $A_{1}, A_{2}, A_{3}, \ldots, A_{i-1}$, respectively. Thus, there must exist at least $2^{i-2}(M+1)-\left(2^{i-2}\right)-\left(\sum_{j=0}^{i-3} 2^{j}\right) M=M$ linearly independent combinations of these rows in the rows of matrix $A_{i}$. Then, we have:

$$
\operatorname{rank}\left(A_{i}\right) \geq \frac{M}{2^{i-2}(M+1)} \times \frac{K}{2}=\frac{K M}{2^{i-1}(M+1)}
$$

This shows that the number of rows of matrix $A_{i}$ is lower bounded by $K M /\left(2^{i-1}(M+1)\right)$.

### 7.4 Achievability Scheme

In this section, we propose an OPIR protocol, referred to as the Online Partitioning (OP) Protocol, for arbitrary $K$ and $M$ where $K /(M+1)$ is a power of 2 . The proposed scheme achieves the rate $(M+1) / K$ in the first round and the rate $\left(2^{i-1}(M+1)\right) / K M$ at rounds $i \geq 2$.

Each round of the OP protocol consists of four steps described as follows.

## Round $i=1$ :

Step 1: The user creates a partition of the $K$ messages into $n_{1} \triangleq K /(M+1)$ sets as follows. First, it selects an index $\mu_{1} \in\left[n_{1}\right]$, by randomly picking an element in $\left[n_{1}\right]$ with uniform probability. Then, the user forms the partition, $P_{\mu_{1}}^{1}$ by combining the demand index and the side information index set $\mathcal{S}: P_{\mu_{1}}^{1} \triangleq\left\{W_{1}\right\} \cup \mathcal{S}$. The user randomly partitions the set of remaining indices $[K] \backslash P_{\mu_{1}}^{1}$ into $n_{1}-1$ sets, each of size $M+1$, denoted as $P_{1}^{1}, \ldots, P_{\mu_{1}-1}^{1}, P_{\mu_{1}+1}^{1}, \ldots, P_{n_{1}}^{1}$.

Step 2: The user sends to the server the partition $\left\{P_{1}^{1}, \ldots, P_{n_{1}}^{1}\right\}$. Note that the server does not know the value of $\mu_{1}$, hence it cannot identify which partition includes $W_{1}$.

Step 3: The server generates the answer $A^{\left[W_{1}, S\right]}=\left\{A_{P_{1}^{1}}, \ldots, A_{P_{n_{1}}^{1}}\right\}$ as a set of $n_{1}$ $\operatorname{sums} A_{P_{1}^{1}}, \ldots, A_{P_{n_{1}}^{1}}$, where $A_{P_{j}^{1}}=\sum_{k \in P_{j}^{1}} X_{k}$ for $j=1, \ldots, n_{1}$.

Step 4: Upon receiving the answer from the server, the user decodes $X_{W_{1}}$ by subtracting the sum of its side information set $\mathcal{X}_{\mathcal{S}}$ from $A_{P_{\mu_{1}}^{1}}$.

## Round $i \geq 2$ :

If the user is able to decode $X_{W_{i}}$, based on the information obtained from the previous rounds, it will not send any request to the server, and proceed to the next round. Otherwise, the user executes the steps below.

Step 1: Let $Q^{\left[W_{i-1}, \mathcal{S}\right]}=\left\{P_{1}^{i-1}, P_{2}^{i-1}, \ldots, P_{n_{i-1}}^{i-1}\right\}$ be the query sent at round $i-1$. Let $\lambda$ be the index of the partition set of $Q^{\left[W_{i-1}, \mathcal{S}\right]}$ that includes $W_{i}$, i.e., $W_{i} \in P_{\lambda}^{i-1}$. Since the user is not able to decode $X_{W_{i}}$ before round $i$, it holds that $W_{i}$ and $\mathcal{S}$ belong to the two different partition sets at round $i-1$, i.e., $\lambda \neq \mu_{i-1}$. The user then creates a partition of $K$ indices into $n_{i}=n_{i-1} / 2$ sets $\left\{P_{1}^{i}, P_{2}^{i}, \ldots, P_{n_{i}}^{i}\right\}$ as follows. First, it selects an index $\mu_{i} \in$ [ $n_{i}$ ], by randomly picking an element in $\left[n_{i}\right]$ with uniform probability. Then, the user forms the partition set $P_{\mu_{i}}^{i}$ by combining $P_{\lambda}^{i-1}$ and $P_{\mu_{i-1}}^{i-1}$, i.e., $P_{\mu_{i}}^{i} \triangleq P_{\lambda}^{i-1} \cup P_{\mu_{i-1}}^{i-1}$. The user then forms $n_{i}-1$ partition sets, each of size $K / n_{i}$, denoted as $P_{1}^{i}, \ldots, P_{\mu_{i}-1}^{i}, P_{\mu_{i}+1}^{i}, \ldots, P_{n_{i}}^{i}$ by randomly pairing partition sets in $\left\{P_{1}^{i-1}, P_{2}^{i-1}, \ldots, P_{n_{i-1}}^{i-1}\right\} \backslash\left\{P_{\lambda}^{i-1}, P_{\mu_{i-1}}^{i-1}\right\}$. The partition $\left\{P_{1}^{i}, P_{2}^{i}, \ldots, P_{n_{i}}^{i}\right\}$ contains $P_{\mu_{i}}^{i}$ and a partition set $P_{j_{1}}^{i-1} \cup P_{j_{2}}^{i-1}$ for any resulting pair $P_{j_{1}}^{i-1}$ and $P_{j_{2}}^{i-1}$ where $j_{1}, j_{2} \in\left[n_{i-1}\right] \backslash\left\{\lambda, \mu_{i-1}\right\}$.

Step 2: The user sends to the server the partition $\left\{P_{1}^{i}, P_{2}^{i}, \ldots, P_{n_{i}}^{i}\right\}$.
Step 3: The server computes $n_{i} \cdot M$ linearly independent combinations of the messages in $\mathcal{X}$. Specifically, for each $P_{j}^{i}, 1 \leq j \leq n_{i}$ the server constructs $M$ linear combinations of two messages as follows. First, recall that each partition set $P_{j}^{i}$ is a union of two partition
sets from the previous round. We denote these partition subsets as $P_{j_{1}}^{i-1}$ and $P_{j_{2}}^{i-1}$, i.e., $P_{j_{1}}^{i-1} \cup P_{j_{2}}^{i-1}=P_{j}^{i}$. Note that $P_{j_{1}}^{i-1}$ is a union of $2^{i-2}$ partition sets in $\left\{P_{1}^{1}, P_{2}^{1}, \ldots, P_{n_{1}}^{1}\right\}$. The server randomly selects one of such partition sets say $P_{i_{1}}^{1}$. Similarly, $P_{j_{2}}^{i-1}$ is a union of $2^{i-2}$ partition sets in $\left\{P_{1}^{1}, P_{2}^{1}, \ldots, P_{n_{1}}^{1}\right\}$. The server randomly selects one of such partition sets say $P_{i_{2}}^{1}$. Finally, the server arbitrarily selects $M$ indices from $P_{i_{1}}^{1}$ and $M$ indices from $P_{i_{2}}^{1}$ and constructs $M$ sums $\left(A_{P_{j}^{i}}\right)_{1}, \ldots,\left(A_{P_{j}^{i}}\right)_{M}$, such that each sum $\left(A_{P_{j}^{i}}\right)_{l}$ for $l \in[M]$ includes one message whose index is selected from $P_{i_{1}}^{1}$ and one message whose index is selected from $P_{i_{2}}^{1}$ and each message is only included in one of the sums. The resulting $M \cdot n_{i}$ linear combinations constitute an answer $A^{\left[W_{i}, \mathcal{S}\right]}$.

Step 4: Upon receiving the answer $A^{\left[W_{i}, \mathcal{S}\right]}$ from the server, the user retrieves $X_{W_{i}}$ by using linear combinations corresponding to the partition $P_{\mu_{i}}^{i}$ and the answers of the previous rounds $A^{\left[W_{1: i-1}\right]}$.

Lemma 17. The $O P$ protocol satisfies the recoverability and individual privacy conditions, while achieving the rate $(M+1) / K$ at first round, and the rate $\left(2^{i-1}(M+1)\right) / K M$ at round $i \geq 2$, when $K /(M+1)$ is a power of 2 .

Proof. The OP protocol for the first round is based the Partition and Code PIR Scheme which satisfies the recoverability and the privacy conditions and achieves the rate $(M+1) / K \quad$ [25]. At round $i \geq 2$, the answer $A^{\left[W_{i}, \mathcal{S}\right]}$ consists of $n_{i} M=K M /\left(2^{i-1}(M+1)\right)$ linear combinations of the messages in $\mathcal{X}$, i.e., $A^{\left[W_{i}, \mathcal{S}\right]}=\left\{\left(A_{P_{j}^{i}}\right)_{1}, \ldots,\left(A_{P_{j}^{i}}\right)_{M}\right\}$ for $j \in\left[n_{i}\right]$. It should be noted that $\left\{\left(A_{P_{1}^{i}}\right)_{1}, \ldots,\left(A_{P_{n_{i}}^{i}}\right)_{M}\right\}$ are linearly independent combinations of the messages in $\mathcal{X}$. (This is because each sum constructed in any round $i \geq 2$ includes two (distinct) messages, one from a partition set $P_{i_{1}}^{1}$ and one from another partition set $P_{i_{2}}^{1}$, such that there do not exist any other linear combinations including both of these two messages). Since the messages in $\mathcal{X}$ are uniformly and indepen-
dently distributed over $\mathbb{F}_{q^{m}}$, then $\left\{\left(A_{P_{1}^{i}}\right)_{1}, \ldots,\left(A_{P_{n_{i}}^{i}}\right)_{M}\right\}$ are uniformly and independently distributed over $\mathbb{F}_{q^{m}}$, i.e., $H\left(\left(\boldsymbol{A}_{P_{1}^{i}}\right)_{1}\right)=\cdots=H\left(\left(\boldsymbol{A}_{P_{n_{i}}}\right)_{M}\right)=m \log _{2} q=L$, and $H\left(\boldsymbol{A}^{\left[W_{i}, \mathcal{S}\right]}\right)=H\left(\left(\boldsymbol{A}_{P_{1}^{i}}\right)_{1}\right)+\cdots+H\left(\left(\boldsymbol{A}_{P_{n_{i}}}\right)_{M}\right)=n_{i} M L$. Therefore, the rate of the OP protocol at round $i \geq 2$ is equal to $L / H\left(\boldsymbol{A}^{\left[W_{i}, \mathcal{S}\right]}\right)=\left(2^{i-1}(M+1)\right) / K M$.

It should be obvious that the recoverability condition is satisfied in the first round. It is also easy to verify that at the beginning of any round $i \geq 2$ (excluding the rounds for which the demand has been recovered previously), all $2^{i-2}(M+1)$ messages whose indices belong to $P_{\mu_{i-1}}^{i-1}$ for some (unique) $\mu_{i-1} \in\left[n_{i-1}\right]$ are recovered in the previous round(s). (For instance, at the beginning of the second round, all $M+1$ messages of the partition set $P_{\mu_{1}}^{1}$ are recovered in the first round.) Suppose that the user demands the message $X_{W_{i}}$ in the round $i \geq 2$ where $X_{W_{i}}$ has not been already recovered in the previous round(s). That is, $W_{i}$ belongs to $P_{j}^{i-1}$ for some $j \in\left[n_{i-1}\right] \backslash\left\{\mu_{i-1}\right\}$. By the step 3 of the OP protocol, the answer of round $i$ includes $M$ sums of distinct pairs of messages, where for some fixed $j^{\prime}, j^{\prime \prime}$, for all of these pairs of messages, the index of one message belongs to the partition set $P_{j^{\prime}}^{1} \subset P_{\mu_{i-1}}^{i-1}$ and the index of the other message belongs to the partition set $P_{j^{\prime \prime}}^{1} \subset P_{j}^{i-1}$. Using these $M$ (linearly independent) sums, and by the fact that all messages with indices belonging to $P_{\mu_{i-1}}^{i-1}$ (including $P_{j^{\prime}}^{1}$ ) have been recovered previously, it follows that all $M+1$ messages with indices in the partition set $P_{j^{\prime \prime}}^{1}$ can be recovered. Given these new $M+1$ recovered messages, from the construction of $P_{j}^{i-1}$, it readily follows that all other messages with indices belonging to $P_{j}^{i-1} \backslash P_{j^{\prime \prime}}^{1}$ can be recovered. That is, all messages whose indices belong to $P_{j}^{i-1}$ can be recovered in round $i$. This confirms that all messages, including the message $X_{W_{i}}$, with indices in $P_{\mu_{i}}^{i}=P_{\mu_{i-1}}^{i-1} \cup P_{j}^{i-1}$ are recovered by the end of the round $i$.

To prove that the OP protocol satisfies the privacy condition at round $i \geq 2$, we need to show that $\mathbb{P}\left(\boldsymbol{W}_{j}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \boldsymbol{\mathcal { X }}=\mathcal{X}\right)=1 / K$ for all $W^{\prime} \in[K]$ and all $j \in[i]$. Since the OP protocol does not depend on the contents of the messages in $\mathcal{X}$, it is
sufficient to prove that $\mathbb{P}\left(\boldsymbol{W}_{j}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)=1 / K$ for all $W^{\prime} \in[K]$ and all $j \in[i]$. Here we only give the proof for the case of $j=i$; and the proof for the cases of $1 \leq j \leq i-1$, not presented here to avoid repetition, is based on a similar technique. For the case of $j=i$, we have:

$$
\begin{aligned}
& \mathbb{P}\left(\boldsymbol{W}_{i}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right) \\
& \quad=\sum_{\mathcal{S}^{\star}} \mathbb{P}\left(\boldsymbol{W}_{i}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \boldsymbol{\mathcal { S }}=\mathcal{S}^{\star}\right) \mathbb{P}\left(\boldsymbol{\mathcal { S }}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)
\end{aligned}
$$

where the sum is over all possible $\mathcal{S}^{\star}$ of size $M$, each of which is a potential side information index set for the demand index $W^{\prime}$. First, we compute $\mathbb{P}\left(\boldsymbol{W}_{i}=W^{\prime} \mid \boldsymbol{Q}^{\left[W_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \mathcal{S}=\mathcal{S}^{\star}\right)$. Without loss of generality, assume $W^{\prime}$ belongs to the $k$ th partition set of round $i$, i.e. $P_{k}^{i}$. As mentioned earlier, at round $i \geq 2$, each partition set is a union of two partition sets of round $i-1$. Without loss of generality, assume that: (i) the $k$ th partition set of round $i$ (of size $2^{i-1}(M+1)$ ) is the union of $w$ th and $v$ th partition sets of round $i-1$ (each of size $2^{i-2}(M+1)$ ), i.e., $P_{k}^{i}=P_{w}^{i-1} \cup P_{v}^{i-1}$, and (ii) $W^{\prime}$ is located in the $P_{w}^{i-1}$. Any potential side information index set $\mathcal{S}^{\star}$ for $W^{\prime}$ must be a subset of $P_{v}^{i-1}$ of size $M$. Any such subset belongs to one of the partition sets of the first round. On the other hand, $P_{v}^{i-1}$ (of size $2^{i-2}(M+1)$ ) is a union of $2^{i-2}$ partition sets of the first round. Let $\Lambda$ be the index set of all such partition sets. For each partition set $P_{\ell}^{1}$ for $\ell \in \Lambda$, all $\binom{M+1}{M}=M+1$ subsets of size $M$ of $P_{\ell}^{1}$ can be considered as a potential side information $\mathcal{S}^{\star}$ for $W^{\prime}$. Thus, there exist $2^{i-2}(M+1)$ potential side information index sets $\mathcal{S}^{\star}$ for the demand index $W_{i}=W^{\prime}$. For any specific $\mathcal{S}^{\star}$, given that $\mathcal{S}^{\star}$ belongs to $P_{v}^{i-1}$, each of the $2^{i-2}(M+1)$ elements in the partition set $P_{w}^{i-1}$ is equally likely to be the user's demand index. That is,

$$
\mathbb{P}\left(\boldsymbol{W}_{i}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \boldsymbol{\mathcal { S }}=\mathcal{S}^{\star}\right)=\frac{1}{2^{i-2}(M+1)}
$$

for all potential side information index sets $\mathcal{S}^{\star}$ for $W^{\prime}$.
Next, we compute $\mathbb{P}\left(\mathcal{S}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[W_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)$. By the application of the total probability theorem and the chain rule of conditional probability, we have:

$$
\begin{aligned}
& \mathbb{P}\left(\mathcal{S}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right) \\
& \quad=\sum_{l=1}^{\frac{K}{M+1}} \mathbb{P}\left(\boldsymbol{\mathcal { S }}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \mathcal{S} \in P_{\ell}^{1}\right) \mathbb{P}\left(\boldsymbol{\mathcal { S }} \in P_{\ell}^{1} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \boldsymbol{\mathcal { S }}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)
\end{aligned}
$$

Note that $\mathbb{P}\left(\mathcal{S} \in P_{\ell}^{1} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i} \mathcal{S}\right]}\right)=\frac{M+1}{K}$ for all $\ell \in\left[\frac{K}{M+1}\right]$. In each partition set of the first round, there exist $\binom{M+1}{M}=M+1$ subsets of size $M$, each of which is equally likely to be the potential side information index set. Note also that any given $\mathcal{S}^{\star}$ belongs to one (and only one) partition set of the first round, say $P_{\ell^{\star}}^{1}$. Thus, $\mathbb{P}\left(\mathcal{S}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[W_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}, \mathcal{S} \in P_{\ell}^{1}\right)=\frac{1}{M+1}$ for $\ell=\ell^{\star}$, and it is zero for any $\ell \in$ $\left[\frac{K}{M+1}\right] \backslash\left\{\ell^{\star}\right\}$. Thus, we have:

$$
\mathbb{P}\left(\mathcal{S}=\mathcal{S}^{\star} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)=\frac{1}{M+1} \times \frac{M+1}{K}=\frac{1}{K} .
$$

Putting the above arguments together, we get

$$
\mathbb{P}\left(\boldsymbol{W}_{i}=W^{\prime} \mid \boldsymbol{Q}^{\left[\boldsymbol{W}_{1: i}, \mathcal{S}\right]}=Q^{\left[W_{1: i}, \mathcal{S}\right]}\right)=2^{i-2}(M+1) \times \frac{1}{2^{i-2}(M+1)} \times \frac{1}{K}=\frac{1}{K} .
$$

This completes the proof for the case of $j=i$.

### 7.5 Example of a protocol execution

Assume that the server has $K=12$ messages $\left\{X_{1}, X_{2}, \ldots, X_{12}\right\}$, and the user has $M=2$ messages, $X_{2}$ and $X_{3}$, as side information, i.e., $\mathcal{S}=\{2,3\}$.

First round: Suppose that the user requires the message $X_{1}$, i.e., $W_{1}=1$ at the first
round. The user creates four sets of size $3 P_{1}^{1}, \ldots, P_{4}^{1}$ as follows. First, the user randomly picks one of the partitions, say $P_{1}^{1}$. The user constructs $P_{1}^{1}=\left\{W_{1}, \mathcal{S}\right\}=\{1,2,3\}$ and randomly partitions the set of remaining indices into sets $P_{2}^{1}, P_{3}^{1}, P_{4}^{1}$. Assume the user has chosen $P_{2}^{1}=\{4,5,6\}, P_{3}^{1}=\{7,8,9\}, P_{4}^{1}=\{10,11,12\}$. Then, the user sends to the server the partition $\left\{P_{1}^{1}, \ldots, P_{4}^{1}\right\}$. The server sends back to the user four coded packets:

$$
\begin{aligned}
& Y_{1}=X_{1}+X_{2}+X_{3}, Y_{2}=X_{4}+X_{5}+X_{6} \\
& Y_{3}=X_{7}+X_{8}+X_{9}, Y_{4}=X_{10}+X_{11}+X_{12}
\end{aligned}
$$

It is clear that the user retrieves $X_{1}$ by replacing the values of $X_{2}$ and $X_{3}$ in $Y_{1}$.
Second round: The user demands the message $X_{4}$, i.e., $W_{2}=4$. The user creates a partition of the indices [12] into 2 sets $P_{1}^{2}, P_{2}^{2}$, each of size 6 as follows. The user randomly picks one of these two partitions, say $P_{1}^{2}$, and forming $P_{1}^{2}=P_{1}^{1} \cup P_{2}^{1}=\{1, \ldots, 6\}$ (Since $W_{2}=4 \in P_{2}^{1}$ and $\mathcal{S} \in P_{1}^{1}$ ). Thus, $P_{2}^{2}=P_{3}^{1} \cup P_{4}^{1}=\{7, \ldots, 12\}$. The user sends to the server the partition $\left\{P_{1}^{2}, P_{2}^{2}\right\}$. The server computes 4 linearly independent combinations of the messages as follows. Since $P_{1}^{2}=P_{1}^{1} \cup P_{2}^{1}$, the server arbitrarily selects two indices from $P_{1}^{1}$, say $\{1,2\}$ and two indices from $P_{2}^{1}$, say $\{5,6\}$. Then, the server constructs 2 linear combinations of two messages such that in each linear combination one message index is selected from $\{1,2\}$ and the other message index is picked from $\{5,6\}$. Also, Since $P_{2}^{2}=P_{3}^{1} \cup P_{4}^{1}$, the server arbitrarily selects two indices from $P_{3}^{1}$, say $\{7,8\}$ and two indices from $P_{4}^{1}$, say $\{10,12\}$. Then, the server constructs 2 other linear combinations of two messages such that in each linear combination one message index is selected from $\{7,8\}$ and the other message index is picked from $\{10,12\}$. The server sends back to the user four coded packets as follows: $Z_{1}=X_{1}+X_{5}, Z_{2}=X_{2}+X_{6}, Z_{3}=X_{7}+X_{10}$, and $Z_{4}=X_{8}+X_{12}$. The user has already downloaded $X_{1}$ from the first round. Thus, the user can retrieve $X_{4}$ from the answers of the first and second rounds.

Third round: The user demands the message $X_{11}$, i.e., $W_{3}=11$. Since $W_{3}=11 \in P_{2}^{2}$, and $\mathcal{S} \in P_{1}^{2}$, the user forms $P_{1}^{3}=\left\{P_{1}^{2} \cup P_{2}^{2}\right\}$ and sends it to the server. Since $P_{1}^{2}=P_{1}^{1} \cup P_{2}^{1}$ and $P_{2}^{2}=P_{3}^{1} \cup P_{4}^{1}$, the server randomly chooses one of the two partitions $\left\{P_{1}^{1}, P_{2}^{1}\right\}$ and one of the two partitions $\left\{P_{3}^{1}, P_{4}^{1}\right\}$. Suppose the server has chosen $P_{2}^{1}, P_{3}^{1}$. Then, the server arbitrarily chooses two indices from $P_{2}^{1}$, say $\{4,6\}$ and two indices from $P_{3}^{1}$, say $\{7,9\}$. The server constructs 2 linearly independent combinations of two messages such that in each linear combination one message index is picked from $\{4,6\}$ and the other message index is selected from $\{7,9\}$. Finally, the server sends back to the user the following two coded packets: $T_{1}=X_{4}+X_{7}, T_{2}=X_{6}+X_{9}$. The user has already downloaded $X_{1}$ from the first round and $X_{4}, X_{5}$, and $X_{6}$ from the second round. It is easy to verify that the user can retrieve $X_{11}$ from the answers of the first, second and third rounds.

## 8. CONCLUSIONS AND FUTURE DIRECTIONS

In this dissertation, we addressed some of the challenges that arise in the Group Testing (GT) problem and its two variants, the Quantitative GT (QGT) and Coin Weighing problems, by developing novel algorithms for various settings. We also addressed some of the challenges in the Private Information Retrieval (PIR) problem. In what follows, we outline the contributions of this dissertation briefly and provide a number of related open problems and potential future directions.

In Chapter 2, we studied a generalized version of the CW problem with a spring scale. We proposed and analyzed a simple and effective adaptive weighing strategy for $d=k=2$, where the weight of each coin is an unknown integer in the range of $\{0,1, \ldots, k\}$ and $d$ is the total weight of the coins. The results of our theoretical analysis show that the proposed strategy requires $2 \log _{2} n-1$ number of weighings in worst case, and it requires about $1.365 \log n-0.5$ number of weighings on average, where $n$ is the total number of coins. This is the first non-trivial achievable upper bound on the minimum expected required number of weighings for $d=k=2$. Additionally, for the average-case setting, we designed and analyzed an optimal strategy within the class of nested strategies, which are mostly being used in today's applications, that requires $\frac{2 n+1}{n+1} \log n-\frac{2(n-1)}{n+1}$ weighings on average. A simple analysis showsed that as $n$ grows unbounded, the proposed strategy, when compared to the optimal nested strategy, requires about $31.75 \%$ less number of weighings on average; and when compared to the information-theoretic lower bound, the proposed strategy requires at most about $8.16 \%$ extra number of weighings on average. The proposed algorithms are for the adaptive setting of CW problem. Thus, an immediate future direction is to propose and analyze non-adaptive weighing strategies for the case $d=k=2$. Also, it should be noted that, a special case of the CW prob-
lem, i.e., $d=k=2$, was considered in Chapter 2. Proposing and analyzing adaptive and non-adaptive weighing strategies for arbitrary values of $d$ and $k$ remains as another open problems.

In Chapter 3 and Chapter 4, we studied the QGT problem for the combinatorial and probabilistic models of defective items, respectively. We proposed non-adaptive QGT algorithms using sparse graph codes over bi-regular and irregular bipartite graphs, and binary $t$-error-correcting BCH codes. The proposed schemes provide exact recovery with probabilistic guarantee, i.e. recover all the defective items with high probability. For any $t \leq 4$, the testing and recovery algorithms of the proposed schemes have the computational complexity of $\mathcal{O}\left(N \log \frac{N}{K}\right)$ and $\mathcal{O}\left(K \log \frac{N}{K}\right)$, respectively. The proposed schemes outperforms existing non-adaptive QGT schemes for the sub-linear regime in terms of the number of tests required to identify all defective items with high probability. It should be noted that although the proposed QGT algorithms outperform all other existing algorithms, their testing matrix is not optimal. Thus, designing an optimal testing matrix that yields an efficient recovery algorithm appears to be a very interesting open problem. The proposed QGT algorithms strongly relies on the assumption that there is no noise in the system. A future direction one can follow is to design QGT algorithms for noisy scenarios. There are a variety of noise models such as binary symmetric noise, erasure noise, dilution noise, etc. Depending on the noise model, the algorithms and their performance may vary in different ways. Another interesting future direction is to consider a variant of the QGT problem in which the number of items involved in a test is limited. This restriction changes the problem completely, and implies the need for a novel idea.

In Chapter 5, we studied the noisy GT problem which is concerned with recovering all defective items in a given population of items. We considered a practical regime in which the number of items is in the order of hundreds, and investigated the performance of two variants of Belief Propagation (BP) algorithm for decoding of noisy non-adaptive

GT under the combinatorial model for defective items. Through extensive simulations, we showed that the proposed algorithms achieve higher success probability and lower false-negative and false-positive rates when compared to the traditional BP algorithm. In the context of group testing, it is extremely difficult to analyze the performance of BP algorithms even for the asymptotic regime. To the best of our knowledge, no theoretical analysis has been provided for the BP-based group testing algorithms so far. However, we are hopeful that our work ignites interest in the researchers who are working in this area to further improve the performance of the group testing algorithms and to perform the theoretical analysis of their performance.

In Chapter 6, motivated by practical scenarios, such as testing for viral diseases, we studied a GT with side information problem. We focused on the following settings: (i) the GT procedure is noisy, i.e., the outcome of the GT procedure can be flipped with a certain probability; (ii) there is a certain amount of side information on the distribution of the infected individuals available to the GT algorithm. First, we proposed a probabilistic model, referred to as an interaction model, that captures the side information about the probability distribution of the infected individuals. Next, we presented a decoding scheme, based on belief propagation, that leverages the interaction model to improve the decoding accuracy. Our results indicated that the proposed algorithm achieves higher success probability and lower false-negative and false-positive rates when compared to the traditional belief propagation especially in the high noise regime. In our interaction model, we only considered one round of interactions between individuals. Extending the interaction model to capture interactions between individuals in more than one round is an interesting future direction.

In Chapter 7, we studied the problem of single-server online PIR with side information. In this problem, there is a user who wishes to download a sequence of messages $\mathcal{X}_{W}=\left\{X_{W_{1}}, X_{W_{2}}, \ldots, X_{W_{t}}\right\}$ from a database $\mathcal{X}$ of $K$ messages, stored on a single server. The communication is performed in rounds, such that at round $i$, the user wishes to retrieve
a message $X_{W_{i}}$ for some $W_{i} \in[K]$. We assume that the user decides on which message $W_{i}$ to request at round $i$ at the beginning of that round and that the identity of the future messages $W_{j}, j>i$ are not known at that time. We also assume that at the beginning of the first round the user has access to $M$ messages which are selected uniformly at random from the database. The identity of these $M$ messages are not known to the server. We focused on the scenario where at round $i$, the user wishes to protect the identity of all the requested messages individually up to round $i,\left\{W_{1}, \ldots, W_{i}\right\}$ for $1 \leq i \leq t$. That is, after the user makes a request to the server at round $i$, the server cannot decide which of the $K$ messages is more likely to get requested at that round and at the previous rounds. Focusing on scalar-linear settings, we characterized the per-round capacity, i.e., the maximum achievable download rate at each round. Note that the tightness of the scalar-linear capacity for general schemes is still open. We also presented a scalar-linear coding scheme that achieves this capacity. We showed that for the setting with $K$ messages stored at the server and a random subset of $M$ messages available to the user at the first round, the per-round capacity of the scalar-linear scheme is $C_{1}=(M+1) / K$ for the first round and $C_{i}=\left(2^{i-1}(M+1)\right) / K M$ for round $i \geq 2$, provided that $K /(M+1)=2^{l}$ for some $l \geq 1$. The generalization of these results for the cases in which $K /(M+1)$ is not a power of 2 is not straightforward, and remains an open problem.

## REFERENCES

[1] R. Dorfman, "The detection of defective members of large populations," The Annals of Mathematical Statistics, vol. 14, no. 4, pp. 436-440, 1943.
[2] A. Ganesan, S. Jaggi, and V. Saligrama, "Learning immune-defectives graph through group tests," IEEE Transactions on Information Theory, vol. 63, no. 5, pp. 30103028, 2017.
[3] A. Sharma and C. R. Murthy, "Group testing-based spectrum hole search for cognitive radios," IEEE Transactions on Vehicular Technology, vol. 63, no. 8, pp. 37943805, 2014.
[4] H. A. Inan, P. Kairouz, and A. Özgür, "Sparse combinatorial group testing," IEEE Transactions on Information Theory, vol. 66, no. 5, pp. 2729-2742, 2020.
[5] D. M. Malioutov, K. R. Varshney, A. Emad, and S. Dash, Learning Interpretable Classification Rules with Boolean Compressed Sensing, pp. 95-121. Cham: Springer International Publishing, 2017.
[6] K. R. Narayanan, A. Heidarzadeh, and R. Laxminarayan, "On accelerated testing for covid-19 using group testing," arXiv preprint arXiv:2004.04785, 2020.
[7] B. Abdalhamid, C. R. Bilder, E. L. McCutchen, S. H. Hinrichs, S. A. Koepsell, and P. C. Iwen, "Assessment of Specimen Pooling to Conserve SARS CoV-2 Testing Resources," American Journal of Clinical Pathology, vol. 153, pp. 715-718, 042020.
[8] M. Aldridge, "Conservative two-stage group testing," arXiv preprint arXiv:2005.06617, 2020.
[9] N. Shental, S. Levy, V. Wuvshet, S. Skorniakov, B. Shalem, A. Ottolenghi, Y. Greenshpan, R. Steinberg, A. Edri, R. Gillis, M. Goldhirsh, K. Moscovici, S. Sachren,
L. M. Friedman, L. Nesher, Y. Shemer-Avni, A. Porgador, and T. Hertz, "Efficient high-throughput sars-cov-2 testing to detect asymptomatic carriers," Science Advances, vol. 6, no. 37, 2020.
[10] E. Karimi, A. Heidarzadeh, K. R. Narayanan, and A. Sprintson, "Noisy group testing with side information," arXiv preprint arXiv:2202.12284, 2022.
[11] M. Sobel and P. A. Groll, "Binomial group-testing with an unknown proportion of defectives," Technometrics, vol. 8, no. 4, pp. 631-656, 1966.
[12] A. Mazumdar, "Nonadaptive group testing with random set of defectives," IEEE Trans. Inf. Theor., vol. 62, pp. 7522-7531, Dec. 2016.
[13] C. Wang, Q. Zhao, and C. N. Chuah, "Optimal nested test plan for combinatorial quantitative group testing," IEEE Transactions on Signal Processing, vol. PP, no. 99, 2017.
[14] E. Karimi, F. Kazemi, A. Heidarzadeh, and A. Sprintson, "A simple and efficient strategy for the coin weighing problem with a spring scale," in 2018 IEEE International Symposium on Information Theory (ISIT), pp. 1730-1734, June 2018.
[15] E. Karimi, A. Heidarzadeh, K. R. Narayanan, and A. Sprintson, "Scheduling improves the performance of belief propagation for noisy group testing," arXiv preprint arXiv:2110.10110, 2021.
[16] J. Scarlett and V. Cevher, "How little does non-exact recovery help in group testing?," in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6090-6094, March 2017.
[17] K. Lee, R. Pedarsani, and K. Ramchandran, "Saffron: A fast, efficient, and robust framework for group testing based on sparse-graph codes," in 2016 IEEE International Symposium on Information Theory (ISIT), pp. 2873-2877, 2016.
[18] M. Lotfi and M. Vidyasagar, "A fast noniterative algorithm for compressive sensing using binary measurement matrices," IEEE Trans. on Signal Processing, vol. 66, pp. 4079-4089, Aug 2018.
[19] M. Iwen, "Compressed sensing with sparse binary matrices: Instance optimal error guarantees in near-optimal time," Journal of Complexity, vol. 30, no. 1, pp. 1-15, 2014.
[20] X. Li, D. Yin, S. Pawar, R. Pedarsani, and K. Ramchandran, "Sub-linear time support recovery for compressed sensing using sparse-graph codes," IEEE Transactions on Information Theory, vol. 65, no. 10, pp. 6580-6619, 2019.
[21] J. Haupt and R. Baraniuk, "Robust support recovery using sparse compressive sensing matrices," in 2011 45th Annual Conference on Information Sciences and Systems, pp. 1-6, March 2011.
[22] D. Du and F. Hwang, Combinatorial Group Testing and Its Applications. Applied Mathematics, World Scientific, 2000.
[23] S. Keiper, G. Kutyniok, D. G. Lee, and G. E. Pfander, "Compressed sensing for finite-valued signals," Linear Algebra and its Applications, vol. 532, pp. 570-613, 2017.
[24] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in Proc. IEEE 36th Annual Foundations of Computer Science, pp. 41-50, 1995.
[25] S. Kadhe, B. Garcia, A. Heidarzadeh, S. E. Rouayheb, and A. Sprintson, "Private information retrieval with side information: The single server case," in 55th Annual Allerton Conf. on Commun., Control, and Computing, pp. 1099-1106, Oct 2017.
[26] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "Capacity of single-server singlemessage private information retrieval with coded side information," in Proc. IEEE

Information Theory Workshop (ITW'18), Nov 2018.
[27] R. Tandon, "The capacity of cache aided private information retrieval," in 55th Annual Allerton Conference on Commun., Control, and Computing, pp. 1078-1082, Oct 2017.
[28] Y.-P. Wei, K. Banawan, and S. Ulukus, "Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching," IEEE Transactions on Information Theory, 2018.
[29] Y.-P. Wei, K. Banawan, and S. Ulukus, "Cache-aided private information retrieval with partially known uncoded prefetching: Fundamental limits," IEEE Journal on Selected Areas in Communications, vol. 36, no. 6, pp. 1126-1139, 2018.
[30] Z. Chen, Z. Wang, and S. Jafar, "The capacity of private information retrieval with private side information," arXiv preprint arXiv:1709.03022, 2017.
[31] S. P. Shariatpanahi, M. J. Siavoshani, and M. A. Maddah-Ali, "Multi-message private information retrieval with private side information," in Proc. IEEE Information Theory Workshop (ITW), Nov 2018.
[32] A. Heidarzadeh, F. Kazemi, and A. Sprintson, "Capacity of single-server singlemessage private information retrieval with private coded side information," in 2019 IEEE International Symposium on Information Theory (ISIT), pp. 1662-1666, 2019.
[33] A. Heidarzadeh, S. Kadhe, S. El Rouayheb, and A. Sprintson, "Single-server multimessage individually-private information retrieval with side information," in 2019 IEEE International Symposium on Information Theory (ISIT), pp. 1042-1046, 2019.
[34] N. H. Bshouty, "Optimal algorithms for the coin weighing problem with a spring scale," in Conference on Learning Theory, 2009.
[35] H. S. Shapiro, "Problem E 1399," Amer. Math. Monthly, vol. 67, no. 82, pp. 697-697, 1960.
[36] B. Lindström, "Determining subsets by unramified experiments," in A survey of Statistical Design and Linear Models, 1975.
[37] M. Aigner and M. Schughart, "Determining defectives in a linear order," Journal of Statistical Planning and Inference, vol. 12, pp. 359-368, 1985.
[38] M. Aigner, "Search problems on graphs," Discrete Applied Mathematics, vol. 14, no. 3, pp. 215-230, 1986.
[39] S. Martirosyan and G. Khachatryan, "Construction of signature codes and the coin weighing problem," vol. 25, 041990.
[40] M. Aigner, Combinatorial Search. New York, NY, USA: John Wiley \& Sons, Inc., 1988.
[41] R. Uehara, K. Tsuchida, and I. Wegener, "Identification of partial disjunction, parity, and threshold functions," Theoretical Computer Science, vol. 230, no. 1, pp. 131147, 2000.
[42] A. Emad and O. Milenkovic, "Semiquantitative Group Testing," IEEE Transactions on Information Theory, vol. 60, pp. 4614-4636, Aug. 2014.
[43] E. Karimi, F. Kazemi, A. Heidarzadeh, K. R. Narayanan, and A. Sprintson, "Sparse graph codes for non-adaptive quantitative group testing," in 2019 IEEE Information Theory Workshop (ITW), pp. 1-5, 2019.
[44] B. Lindström, "On b2-sequences of vectors," Journal of number Theory, vol. 4, no. 3, pp. 261-265, 1972.
[45] P. Abdalla, A. Reisizadeh, and R. Pedarsani, "Multilevel group testing via sparsegraph codes," in 2017 51st Asilomar Conference on Signals, Systems, and Computers, pp. 895-899, Oct 2017.
[46] J. Scarlett and V. Cevher, "Limits on sparse support recovery via linear sketching with random expander matrices," in Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (A. Gretton and C. C. Robert, eds.), vol. 51 of Proceedings of Machine Learning Research, (Cadiz, Spain), pp. 149-158, PMLR, 09-11 May 2016.
[47] S. Lin and D. J. Costello, Error control coding. Pearson Education India, 2001.
[48] A. Vem, N. T. Janakiraman, and K. R. Narayanan, "Group testing using left-and-right-regular sparse-graph codes," arXiv preprint arXiv:1701.07477, 2017.
[49] C.-L. Chen, "Formulas for the solutions of quadratic equations over GF $\left(2^{m}\right)$ (corresp.)," IEEE Transactions on Information Theory, vol. 28, no. 5, pp. 792-794, 1982.
[50] E. Karimi, F. Kazemi, A. Heidarzadeh, K. R. Narayanan, and A. Sprintson, "Nonadaptive quantitative group testing using irregular sparse graph codes," in 2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 608-614, 2019.
[51] O. Gebhard, M. Hahn-Klimroth, D. Kaaser, and P. Loick, "Quantitative group testing in the sublinear regime," arXiv preprint arXiv:1905.01458, 2019.
[52] 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, pp. 163-170, 1975.
[53] J. Scarlett and V. Cevher, "Phase transitions in the pooled data problem," in Advances in Neural Information Processing Systems 30 (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 377-385, Curran Associates, Inc., 2017.
[54] A. El Alaoui, A. Ramdas, F. Krzakala, L. Zdeborová, and M. I. Jordan, "Decoding from pooled data: Phase transitions of message passing," IEEE Transactions on Information Theory, vol. 65, pp. 572-585, Jan 2019.
[55] J. Acharya and A. T. Suresh, "Optimal multiclass overfitting by sequence reconstruction from hamming queries," arXiv preprint arXiv:1908.03156, 2019.
[56] C.-C. Cao, C. Li, and X. Sun, "Quantitative group testing-based overlapping pool sequencing to identify rare variant carriers," BMC bioinformatics, vol. 15, no. 1, p. 195, 2014.
[57] G. De Marco, T. Jurdziński, and D. R. Kowalski, "Optimal channel utilization with limited feedback," in Fundamentals of Computation Theory (L. A. Gasieniec, J. Jansson, and C. Levcopoulos, eds.), (Cham), pp. 140-152, Springer International Publishing, 2019.
[58] C. Wang, Q. Zhao, and C.-N. Chuah, "Group testing under sum observations for heavy hitter detection," in 2015 Information Theory and Applications Workshop (ITA), pp. 149-153, IEEE, 2015.
[59] M. Cheraghchi, A. Karbasi, S. Mohajer, and V. Saligrama, "Graph-constrained group testing," IEEE Transactions on Information Theory, vol. 58, pp. 248-262, Jan 2012.
[60] H. A. Inan, P. Kairouz, M. Wootters, and A. Ozgur, "On the optimality of the kautzsingleton construction in probabilistic group testing," in 2018 56th Annual Allerton

Conference on Communication, Control, and Computing (Allerton), pp. 188-195, 2018.
[61] P. Nikolopoulos, T. Guo, S. R. Srinivasavaradhan, C. Fragouli, and S. Diggavi, "Community aware group testing," arXiv preprint arXiv:2007.08111, 2020.
[62] S. Ahn, W.-N. Chen, and A. Özgür, "Adaptive group testing on networks with community structure," in 2021 IEEE International Symposium on Information Theory (ISIT), pp. 1242-1247, IEEE, 2021.
[63] M. Aldridge, O. Johnson, and J. Scarlett, "Group testing: An information theory perspective," Foundations and Trends $\circledR$ in Communications and Information Theory, vol. 15, no. 3-4, pp. 196-392, 2019.
[64] A. I. V. Casado, M. Griot, and R. D. Wesel, "Informed dynamic scheduling for beliefpropagation decoding of ldpc codes," in 2007 IEEE International Conference on Communications, pp. 932-937, 2007.
[65] P. Radosavljevic, A. de Baynast, and J. Cavallaro, "Optimized message passing schedules for ldpc decoding," in Conference Record of the Thirty-Ninth Asilomar Conference onSignals, Systems and Computers, 2005., pp. 591-595, 2005.
[66] J. H. McDermott, D. Stoddard, P. J. Woolf, J. M. Ellingford, D. Gokhale, A. Taylor, L. A. Demain, W. G. Newman, and G. Black, "A nonadaptive combinatorial group testing strategy to facilitate health care worker screening during the severe acute respiratory syndrome coronavirus-2 (sars-cov-2) outbreak," The Journal of Molecular Diagnostics, vol. 23, no. 5, pp. 532-540, 2021.
[67] X. Xia, Y. Liu, Y. Xiao, J. Cui, B. Yang, and Y. Peng, "Adagt: An adaptive group testing method for improving efficiency and sensitivity of large-scale screening against
covid-19," IEEE Transactions on Automation Science and Engineering, pp. 1-17, 2021.
[68] J. Zhu, K. Rivera, and D. Baron, "Noisy pooled pcr for virus testing," arXiv preprint arXiv:2004.02689, 2020.
[69] P. Nikolopoulos, S. R. Srinivasavaradhan, T. Guo, C. Fragouli, and S. Diggavi, "Group testing for overlapping communities," in ICC 2021-IEEE International Conference on Communications, pp. 1-7, IEEE, 2021.
[70] R. Goenka, S.-J. Cao, C.-W. Wong, A. Rajwade, and D. Baron, "Contact tracing information improves the performance of group testing algorithms," arXiv preprint arXiv:2106.02699, 2021.
[71] R. A. Kleinman and C. Merkel, "Digital contact tracing for covid-19," CMAJ, vol. 192, no. 24, pp. E653-E656, 2020.
[72] A. M. Ross, L. D. S. Zerden, B. J. Ruth, J. Zelnick, and J. Cederbaum, "Contact tracing: An opportunity for social work to lead," Social Work in Public Health, vol. 35, no. 7, pp. 533-545, 2020. PMID: 32781912.
[73] S. Munzert, P. Selb, A. Gohdes, L. F. Stoetzer, and W. Lowe, "Tracking and promoting the usage of a covid-19 contact tracing app," Nature Human Behaviour, vol. 5, no. 2, pp. 247-255, 2021.
[74] D. J. MacKay, Information theory, inference and learning algorithms. Cambridge university press, 2003.
[75] S. Russell and P. Norvig, "Artificial intelligence: A modern approach," 2003.
[76] J. Barbier and D. Panchenko, "Strong replica symmetry in high-dimensional optimal bayesian inference," arXiv preprint arXiv:2005.03115, 2020.
[77] A. Coja-Oghlan, C. Efthymiou, N. Jaafari, M. Kang, and T. Kapetanopoulos, "Charting the replica symmetric phase," Communications in Mathematical Physics, vol. 359, p. 603-698, Feb 2018.
[78] A. Coja-Oghlan and W. Perkins, "Belief propagation on replica symmetric random factor graph models," Annales de l'institut Henri Poincare D, vol. 5, no. 2, pp. 211249, 2018.
[79] L. Zdeborová and F. Krzakala, "Statistical physics of inference: thresholds and algorithms," Advances in Physics, vol. 65, no. 5, pp. 453-552, 2016.
[80] F. Kazemi, E. Karimi, A. Heidarzadeh, and A. Sprintson, "Single-server singlemessage online private information retrieval with side information," in 2019 IEEE International Symposium on Information Theory (ISIT), pp. 350-354, 2019.
[81] H. Sun and S. A. Jafar, "The capacity of robust private information retrieval with colluding databases," IEEE Transactions on Information Theory, vol. 64, no. 4, pp. 2361-2370, 2018.
[82] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," IEEE Transactions on Information Theory, vol. 64, no. 3, pp. 1945-1956, 2018.
[83] K. Banawan and S. Ulukus, "Multi-message private information retrieval: Capacity results and near-optimal schemes," IEEE Transactions on Information Theory, vol. 64, no. 10, pp. 6842-6862, 2018.
[84] A. Heidarzadeh, B. Garcia, S. Kadhe, S. El Rouayheb, and A. Sprintson, "On the capacity of single-server multi-message private information retrieval with side information," in Proc. 56th Annual Allerton Conference on Commun., Control, and Computing, pp. 180-187, Oct 2018.


[^0]:    *Reprinted with permission from [14] "A Simple and Efficient Strategy for the Coin Weighing Problem with a Spring Scale," by E. Karimi, F. Kazemi, A. Heidarzadeh and A. Sprintson, 2018. In Proceedings of 2018 IEEE International Symposium on Information Theory (ISIT), 2018, pp. 1730-1734. Copyright © by IEEE.

[^1]:    *Reprinted with permission from [43] "Sparse Graph Codes for Non-adaptive Quantitative Group Testing," by E. Karimi, F. Kazemi, A. Heidarzadeh, K. R. Narayanan and A. Sprintson, 2019. In Proceedings of 2019 IEEE Information Theory Workshop (ITW), 2019, pp. 1-5. Copyright © by IEEE.

[^2]:    *Reprinted with permission from [50] "Non-adaptive Quantitative Group Testing Using Irregular Sparse Graph Codes," by E. Karimi, F. Kazemi, A. Heidarzadeh and A. Sprintson, 2019. In Proceedings of 2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 2019, pp. 608614. Copyright © by IEEE.

[^3]:    *Throughout the chapter the base of $\log$ is 2 , unless explicitly noted otherwise.

[^4]:    *Reprinted with permission from [80] "Single-Server Single-Message Online Private Information Retrieval with Side Information," by F. Kazemi, E. Karimi, A. Heidarzadeh and A. Sprintson, 2019. In Proceedings of 2019 IEEE International Symposium on Information Theory (ISIT), 2019, pp. 350-354. Copyright © by IEEE.

