Group Testing with General Correlation Using Hypergraphs

Hesam Nikpey
University of Pennsylvania
Philadelphia, PA, USA
Email: hesam@seas.upenn.edu

Saswati Sarkar
University of Pennsylvania
Philadelphia, PA, USA
Email: swati@seas.upenn.edu

Shirin Saeedi Bidokhti
University of Pennsylvania
Philadelphia, PA, USA
Email: saeedi@seas.upenn.edu

2024 IEEE International Symposium on Information Theory (ISIT) | 979-8-3503-8284-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISIT57864.2024.10619244

***Abstract*—Group testing, a problem with applications in various fields, traditionally assumes independent node states. Recent research, however, focuses on real-world scenarios that often involve correlations among nodes, challenging the simplifying assumptions made in existing models. In this work, we consider a comprehensive model for arbitrary statistical correlation among node states. To capture and leverage these correlations effectively, we model the problem by hypergraphs inspired by [1].**

We establish that arbitrary correlations among nodes can be represented as a hypergraph with a probability distribution over its edges, and design a novel greedy adaptive algorithm capable of conducting informative tests and dynamically updating the distribution. We analyze its performance and give theoretical guarantees on the number of tests that depend solely on the entropy of the underlying probability distribution and the average number of infections.

I. Introduction

In recent years, group testing has gained increasing interest across various disciplines, including computer science [2]–[5], statistics [6], [7], information theory [8]–[12], and biology [13]–[15]. This problem involves a population (set of nodes) where a specific subset is infected, and we can test any subpopulation. A positive test result indicates at least one infection within the subset, and the primary goal is to identify all infected individuals. Historically, group testing operated under the assumption of independent node states [16], [17] and almost matching lower and upper bounds are known on the minimum number of tests. In particular in [17], near-optimal bounds have been obtained for both adaptive and non-adaptive group testing, quantified as $O(H(X) + \mu)$, where $\mu = \sum_i p_i$ represents the average number of infections, with p_i being the probability that a node i is infected, and $H(X) = \sum_i -p_i \log p_i$ being the entropy.

However, states of nodes exhibit strong statistical *correlation* in most real-world settings. For example, in disease propagation scenarios, states of individuals within the same household exhibit correlation; if one member is infected, the likelihood of infection among other members increases [18]–[20]. Similarly, in network fault diagnosis, adjacent connections are more likely to be simultaneously faulty due to shared physical infrastructure or external factors [21]. Some recent papers have significantly enhanced the precision and efficiency of group testing algorithms by incorporating these correlations [20], [22]–[25].

Yet, much of the existing literature that consider group testing strategies for correlated nodes propose 1) models that are context-specific and oftentimes oversimplified, and 2) group testing strategies that are tied to specific correlation models. For disease propagation, the correlation models are built either on the notion of proximity (defined through graphs) [19], [20] or in conjunction with a disease spread model [3], [26]. For network fault diagnosis, the correlation is based on physical proximity between the nodes in a network, or frequency of communication between two nodes [21].

In this work, our objective is to devise group testing algorithms for a comprehensive model for correlation, ie., a general joint distribution denoted by $\mathcal{D}$ on the state of the nodes, where $\mathcal{D}(S)$ represents the probability that exactly a subset S of individuals are infected. Our aim is to devise group testing algorithms tailored to this expansive scope without substantial simplification. We describe our contributions below.

A. Contributions

We model arbitrary statistical correlation by employing hypergraphs, drawing inspiration from [1]. A hypergraph $G = (V, E)$ has node set V, $|V| = n$, and edge set E, where each hyperedge $e \in E$ is a subset of nodes: $e \subseteq V$. We consider an arbitrary probability distribution of infection of hyperedges, which is obtained from the general joint distribution $\mathcal{D}$ of infection of subsets of nodes which in turn captures arbitrary statistical correlation among infection of different nodes. The work in [1] considers a specific hypergraph model where all edges have equal cardinality d, where the correlations are combinatorial rather than statistical [1], and therefore differs in scope from our work.

We propose an adaptive group testing algorithm that is capable of exploiting correlation by the means of updating the posterior distribution on the hyperedges given the previous test results. The algorithm works in two stages. Stage 1 focuses on conducting "informative" tests, which significantly narrows down the search space. When these tests are no longer feasible, the algorithm transitions to a second stage where it tests the remaining uncertain nodes individually. We demonstrate that this algorithm successfully identifies the infected set and requires an expected number of tests that depends on the entropy of the prior distribution $H(X)$ and the mean number of infections μ. Furthermore, using methods

similar to those in [17], we establish entropy as a lower bound on the number of tests. This reveals that when the average number of infections is less than the entropy, our algorithm achieves order-wise optimality. We further provide an example in which the minimum number of tests is governed by μ which dominates $H(X)$ demonstrating the looseness of the most widely used entropy lower bound. The expected number of tests required by our algorithm is lower than or matches those required by algorithms in prior works that focus on specific correlation models.

II. Problem Formulation

We model a generalized group testing problem via hypergraphs [1]. Consider the hypergraph $G = (V, E)$ where V is the set of nodes and E the set of hyperedges. Each hyperedge $e \in E$ is a subset of the nodes, $e \subseteq V$. We assume $|V| = n$. In a population of size n, each individual is represented by a node in this hypergraph and the hyperedges capture possible dependencies between the nodes' states. A distribution $\mathcal{P}$ is assumed over the edges in E, and **one** edge $e \sim \mathcal{P}$ is sampled from this distribution to be infected. We use sampled edge, infected edge, or target edge interchangeably. The probability that hyperedge e is infected is denoted by p_e. When a hyperedge is infected, all nodes $v \in e$ are infected (positive) and all other nodes $u \notin e$ are not infected (negative). The goal is to sequentially (and adaptively) perform group tests, with minimum number of tests, until the infected edge is identified accurately.

Fig 1 shows an example of a hypergraph where $V = \{v_1, v_2, v_3, v_4, v_5\}$ is the set of nodes, $E = \{e_1 = \{v_1, v_2, v_3\}, e_2 = \{v_1, v_5\}, e_3 = \{v_4, v_5\}\}$ is the set of edges, and $p_{e_1} = 0.3$, $p_{e_2} = 0.2$, $p_{e_3} = 0.5$ define the distribution over the edges. Note that v_2 and v_3 have a complete correlation and are always in the same state because each edge contains either both of them or none of them. Also, v_1 and v_4 are also completely correlated as they are always in opposite states because each edge contains exactly one of them.

Let $X = (X_1, X_2, \ldots, X_n)$ be the vector of the nodes' states where $X_i = 1$ if the i'th node is positive and $X_i = 0$ otherwise. We denote the probability that node v is positive by p_v. When an edge e is sampled, we have $\forall v \in e : X_v = 1$ and $\forall v \notin e : X_v = 0$. Within this model, each node v may belong to multiple edges and therefore, we have $p_{v_i} = \mathbb{E}[X_i] = \sum_{e \in E:\ e \ni v_i} p_e$. The expected number of infections, μ, is thus

$$\mu := \mathbb{E}[X_1 + X_2 + \cdots + X_n] = \sum_{i \in V} p_i. \tag{1}$$

It is worthwhile to mention that in prior work such as [17], it is often assumed that $\mu \ll n$, based on the observation that otherwise the number of tests needed is $\Omega(n)$. Interestingly, this is not true in general when we have correlation, as captured in our model, and an example is given in [27, Appendix A]. We next define testing and recovery.

a) Testing.: A group test T is defined by the subset of nodes $T \subseteq V$ that take part in that test. We denote the i'th test by T_i, $T_i \subseteq V$, and its result by r_i. We say $r_i = 1$ (positive test result) iff there is at least one node in T_i that is infected (i.e., belongs to the sampled edge) and $r_i = 0$ otherwise. Testing is done *sequentially and adaptively*. More precisely, T_i is a function of the prior tests $T_1, \ldots, T_{i-1}$ and their respective results $r_1, \ldots, r_{i-1}$. We denote the total number of tests by L.

In order to design the tests sequentially, we utilize the posterior probability of nodes and edges of the hypergraph being infected given the previous tests and their results. In particular, suppose tests $T_1, T_2, \ldots, T_k$ are performed and the results are $r_1, r_2, \ldots, r_k$. We denote the posterior probability of $e \in E$ with $q_{e|\{(T_1,r_1),(T_2,r_2),\ldots,(T_k,r_k)\}}$. Note that $q_{e|\{\}} = p_e$. Similarly, the posterior probability of $v \in V$ being infected is

$$q_{v|\{(T_1,r_1),\ldots,(T_k,r_k)\}} = \sum_{e \in E:\ e \ni v} q_{e|\{(T_1,r_1),\ldots,(T_k,r_k)\}}. \tag{2}$$

When it is clear from the context, we drop $\{(T_1, r_1), (T_2, r_2), \ldots, (T_k, r_k)\}$ and use q_e and q_v.

b) Recovery.: Given the test sequence $T_1, \ldots, T_L$ and the respective results $r_1, \ldots, r_L$, the nodes' states are estimated as $\hat{X} = (\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_n)$ where $\hat{X}_i$ is the estimated state of node i, $i = 1, \ldots, n$. The probability of error is then defined as $P_e = P(X \neq \hat{X})$, where the randomness is over $\mathcal{P}$ and possible randomness of the testing design.

c) Objective.: In this work, we aim to devise algorithms that, in expectation, use the least number of tests (minimize $\mathbb{E}[L]$) and recover the sampled edge with probability of error P_e that goes to zero as n goes to infinity. For some of our results, we have $P_e = 0$.

Remark 1. *This model is equivalent to arbitrary correlation among nodes. To see this consider an arbitrary distribution $\mathcal{D}$ over the power set of a population size n where $\mathcal{D}(S)$ is the probability that exactly subset S of the population is infected, ie, all nodes in S are infected and no node outside S is infected. Now in our model, we can consider a hypergraph where the edge set is the power set of the population, and $p_e = \mathcal{D}(S)$, for the hyperedge e that corresponds to S. Since $\sum_S \mathcal{D}(S) = 1$, $\{p_e\}_{e \in E}$ is a distribution. It is important to note that the set of hyperedges can also be a proper subset of the power set, depending on the distribution $\mathcal{D}$. From this, we can see that $|support(\mathcal{D})| = |E|$.*

In [27, Section 5.1], we provide details on how previous works can be modeled using hypergraphs and show how our testing algorithm would perform on them.

Before presenting our algorithm, we establish a lower bound on the number of tests needed under this model. Our result is inspired by [17] where almost matching lower and upper bounds are proved for adaptive and non-adaptive independent group testing. Using similar techniques, we find that entropy $H(X)$ is a lower bound on the number of tests.

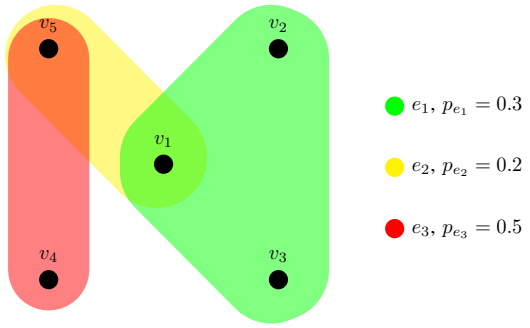

Fig. 1. A hypergraph with $V = \{v_1, v_2, v_3, v_4, v_5\}$, $E = \{\{1,2,3\}, \{1,5\}, \{4,5\}\}$ and $p_{\{1,2,3\}} = 0.3$, $p_{\{1,5\}} = 0.2$, $p_{\{4,5\}} = 0.5$.

Theorem 1. *For any algorithm that recovers the sampled edge e^* with probability at least $1-\epsilon$ using L tests, we have*

$$L \geq (1-\epsilon)H(X) \tag{3}$$

where $H(X) = \sum_{e \in E} -p_e \log p_e$.

This lower bound may not be tight as demonstrated next.

Example 1. *Consider a graph $G = (V, E)$ comprising n nodes where each subset of size $n-1$ is included in E as an hyperedge. Consider a uniform probability distributed on the edges, i.e. $p_e = \frac{1}{n}$ for all $e \in E$. In this example, the entropy $H(X)$ is $\log n$. However, it's notable that every test of size two or greater will yield positive results, rendering them redundant. Consequently, the problem is effectively simplified to individual testing when a single individual is found to be negative which takes $\Omega(n)$ tests.*

III. An Algorithmic Approach

A. Challenges

Most classic adaptive algorithms for independent group testing are, in essence, built on generalized binary-splitting which greedily chooses the test that most evenly splits the candidate nodes or equivalently chooses the test with the maximal information gain. In probabilistic group testing (with independent nodes) [17], this involves choosing subsets of nodes to test such that the probability of a subset containing a positive node is close to $1/2$, meaning that the test provides close to one bit of information. A negative test result allows a subset to be ruled out. Candidate subsets that test positive are, however, partitioned into two subsets (with roughly equal probability mass) for further testing. With such a design, the posterior probability of the sets testing positive, given the sequence of previous test results, is close to $1/2$. To follow this paradigm, we face challenges.

First and foremost, it is important to note that treating each node individually and disregarding the underlying correlation is not efficient in terms of the number of tests needed and we aim to propose testing strategies that exploit the correlation.

Now consider the classical idea of greedily testing subsets and ruling out those subsets that test negative. If one wishes to utilize the correlation that is inherent in our model, it is not straightforward how the group tests should be designed: (i) Suppose that you have chosen a subset S of nodes to test. If this test is negative, not only all nodes in S are negative but also all those edges that contain a node in S are not the target infected edge and could be ruled out. (ii) If the test is positive, we cannot conclude that the nodes that belong to $V \backslash S$ are negative. As a matter of fact, all can still remain valid candidates if they share an edge with an element of S. In other words, we can not conclude that the target infected edge is a subset of S. The design of the tests is thus nontrivial.

B. Some Useful Definitions

Suppose group tests $T_1, T_2, \ldots, T_k$ are performed and the results are $r_1, r_2, \ldots, r_k$. Recall that the posterior probability of edge e (resp. node v) given the previous k test results, is $q_{e|\{(T_1,r_1),(T_2,r_2),\ldots,(T_k,r_k)\}}$ (resp. $q_{v|\{(T_1,r_1),(T_2,r_2),\ldots,(T_k,r_k)\}}$ defined in (2)). When it is clear from the context, we drop the conditions and show them with q_e and q_v. With a slight abuse of notation, we refer to $\{q_e\}_{e \in E}$ as the posterior distribution $\mathcal{D}$ which gets updated after each test.

To connect edge probabilities to the probability that a test becomes positive, we need the following definition.

Definition 1. *For a subset $S \subseteq V$, the edge set of S is defined as $E(S) = \{e \mid e \in E \wedge \forall v \in e : v \in S\}$. The weight of the set S after tests $\{(T_1, r_1), (T_2, r_2), \ldots, (T_k, r_k)\}$ is defined as*

$$w(S|\{(T_1,r_1),\ldots,(T_k,r_k)\}) = \sum_{e \in E(S)} q_{e|\{(T_1,r_1),\ldots,(T_k,r_k)\}}. \tag{4}$$

When clear from the context, we simply drop $\{(T_1, r_1), \ldots, (T_k, r_k)\}$ and write $w(S)$.

Here, $w(S)$ is the probability that $V \setminus S$ becomes negative, and as we will discuss later, we need to keep it close to half.

After performing k tests, the expected number of infections would be updated according to the posterior probabilities discussed above. We denote the expected number of infected nodes in the posterior probability space by:

$$\mu_{\{(T_1,r_1),\ldots,(T_k,r_K)\}} = \sum_{v} q_{v|\{(T_1,r_1),\ldots,(T_k,r_k)\}}. \tag{5}$$

We omit $(T_1, r_1), \ldots, (T_k, r_k)$ when contextually clear.

Throughout this work, when a new test (T_i, r_i) is done, we compute $q_{e|(T_1,r_1),\ldots,(T_i,r_i)}$ from the last posterior $q_{e|(T_1,r_1),\ldots,(T_{i-1},r_{i-1})}$. In other words, we initially start with $\mathcal{D}_0 = \mathcal{P}$ and after observing the results of the i'th test, compute $\mathcal{D}_i$ from $\mathcal{D}_{i-1}$ based on (T_i, r_i) where $\mathcal{D}_i = \{q_{e|(T_1,r_1),\ldots,(T_i,r_i)}\}_{e \in E}$. In Lemma 2, we prove that computing the posterior probability entails "removing some edges from the distribution" according to the following definition:

Definition 2. *Let $\mathcal{D}$ be a distribution over a set E. Removing $e \in E$ from $\mathcal{D}$ entails setting $\mathcal{D}(e) = 0$ and re-normalizing $\mathcal{D}(e)$ by a factor $c > 1$, $\mathcal{D}(e) \leftarrow c\mathcal{D}(e)$, such that $\sum_{e \in E} \mathcal{D}(e) = 1$.*

In the graph in Fig 1, we can see that $p_{v_1} = 0.5$, $p_{v_2} = p_{v_3} = 0.3$, $p_{v_4} = 0.5$, and $p_{v_5} = 0.7$. If $S = \{v_1, v_2, v_3, v_5\}$,

then $E(S) = \{e_1, e_2\}$ and $w(S) = p_{e_1} + p_{e_2} = 0.5$. The average number of infections is $\mu = 0.5 + 0.3 + 0.3 + 0.5 + 0.7 = 2.3$. If a test $\{v_2, v_4\}$ returns positive, $q_{e_2} = 0$ as it can not be the target edge, and the posterior is re-normalized by the scalar 0.8 so that $q_{e_1} = p_{e_1}/.8, q_{e_3} = p_{e_1}/.8$ sum up to 1.

C. Overview and Ideas

The ideas behind our algorithm can be summarized in three points: (1) devise a testing mechanism in which by testing a subset of the nodes, we can rule out edge sets that are not compatible with the result; (2) design the subset so that, independent of the test result, a constant (independent of parameters) fraction of the mass is ruled out. (3) Updating posterior probabilities of infections for the nodes and edges based on the prior test results. If at every step we can do this until we reach a single edge (with probability 1), then we would have done $O(H(X))$ in expectation (Lemma 3) which is optimal by the lower bound in Theorem 1. Example 1 shows that this might not be always possible. These types of tests are "informative", as they shrink the search space significantly.

For instance, suppose a set S with $w(S) = 1/2$ is given and we test $V \setminus S$. If the test is positive, it means that the sampled edge e^* is not in $E(S)$ which contains half the probability mass. If the test is negative, then e^* is in $E(S)$ which rules out $E \setminus E(S)$. So in either case, we gain one bit of information. Since finding a set that has $w(S) = 1/2$ might not be possible, we relax this condition. We aim to find a set S so that $w(S)$ is between two constants $c = 1/2 - \delta$ and $1 - c = 1/2 + \delta$, let's say between 0.05 and 0.95.

We next describe how to find $S \in V$ with $0.05 \leq w(S) \leq 0.95$. We design a greedy-like algorithm that finds S iteratively by removing the nodes from the working set. The algorithm starts with $S = V$. Initially, $w(S) = 1$. Then the algorithm drops the nodes in S one by one until it finds $0.05 \leq w(S) \leq 0.95$ (at which point the algorithm performs a test on $V \setminus S$ and based on the result updates the posterior distribution).

The challenge arises when the algorithm can not find such an S. In proposition 1, we argue that this can happen only when all the remaining nodes in S have at least probability $1 - 2c = .9$, to which we refer as high probability nodes, and the remaining nodes in $V \setminus S$ have a positive node with probability at most $c = 0.05$. What we propose in this stage is the following: In a single test, the algorithm can test all the low-probability nodes. If it is positive, we rule out 0.95 of the mass, which is an unexpected gain, and continue with the algorithm. If it is negative, we end up with the high probability nodes. At this point, the algorithm tests nodes individually. We refer to the first stage of the algorithm when an informative test can be found based on S as Stage 1 of the algorithm, and the second stage of the algorithm for which we test nodes individually as Stage 2 of the algorithm.

D. The Proposed Algorithm

The proposed algorithm is presented in Algorithm 1. In Stage 1 of the algorithm which corresponds to lines 3-14, the algorithm greedily finds a set S so that $c \leq w(S) \leq 1 - c$.

Algorithm 1: Adaptive Algorithm for General GT

1 **Input:** Graph $G = (V, E)$, distribution $\mathcal{D} = \mathcal{P}$ over E, and parameter $c < 1/2$.
2 **Output:** Target edge e^*.
3 Initialize set $S = V$.
4 Compute $w(S)$ by (4) using the (updated) posterior distribution $\mathcal{D} = \{q_e\}_{e \in E}$.
5 **If** $c \leq w(S) \leq 1 - c$, test $V \setminus S$:
6 If the test is positive, remove $E(S)$ from $\mathcal{D}$ by def 2 to update the posterior distribution.
7 If the test is negative, remove $E \setminus E(S)$ from $\mathcal{D}$ by def 2 to update the posterior distribution.
8 Return to line 3.
9 **Else**:
10 If there is an edge with $q_e = 1$, return e.
11 If $\exists v : c \leq w(S \setminus v) \leq 1 - c$, remove v from S and go to line 5.
12 If $\exists v : 1 - c < w(S \setminus v)$, remove v from S and go to line 11.
13 Otherwise ($w(S) > 1 - c$ and $\forall v : w(S \setminus v) < c$), test $V \setminus S$.
14 If the test is positive, remove $E(S)$ from $\mathcal{D}$ by def 2 to update the posterior and go to line 3.
15 If the test is negative, test every node $v \in S$ individually with $0 < q_v < 1$, update the posterior after every test and return the nodes with $q_v = 1$.

This is captured in lines 11 and 12. When the set S is found, it is tested in line 5 and the posterior is updated accordingly. The posterior probabilities are used in the calculation of $w(S)$ using (4). If the algorithm can not find a set $c \leq w(S) \leq 1 - c$ at this stage, it tests $V \setminus S$ in line 14 which has a high probability of being negative. If at Line 14, the test result is positive (unexpectedly), the algorithm rules out $1 - c$ fraction of the mass and gets back to line 5. Stage 2 of the algorithm is outlined in line 15, where it tests uncertain high probability nodes individually. Note that at this point, all the nodes outside of S are negative. The following example shows how Algorithm 1 is run on the graph of Fig 1.

Example 2. *Let's run Algorithm 1 on the graph in Figure 1 with $c = 0.1$. Initially, $S = \{v_1, v_2, v_3, v_4, v_5\}$ and $w(S) = 1$. By removing v_1 from S, $E(S \setminus v_1)$ contains only one edge $e_3 = \{4, 5\}$, hence $c < w(S \setminus v_1) = 0.5 < 1 - c$ and the algorithm tests v_1. If it is negative, then it rules out e_1 and e_2 and after scaling the edges, $q_{e_3} = 1$ and the algorithm returns it. If it is positive, it removes e_3 and scales the other edges by 2 so $q_{e_1} = 0.6$ and $q_{e_2} = 0.4$. Again S would be initialized to contain* all *the nodes, and after removing v_2, $w(S \setminus v_2) = 0.4$ so the algorithm test v_2 to eliminate one of e_1 or e_2. The expected number of tests here is 1.5.*

IV. Analysis

In this section, we bound the expected number of tests performed by Algorithm 1. The main result is built on the

following key lemma. Recall that $H(X)$ is the entropy of the edge distribution and $\tilde{\mu}$ is the expected number of infections in line 15 (computed with the posterior probabilities). The proofs are provided in the full version [27].

Lemma 1. *Algorithm 1 finds e^* with $\mathbb{E}[L_1 + L_2] \leq \frac{1}{\log \frac{1}{(1-c)}} H(X) + \frac{\mathbb{E}[\tilde{\mu}]}{1-2c}$ tests in expectation.*

The roadmap of our proof for Lemma 1 is as follows. We first prove that removing $E(S)$ (resp. $E \backslash E(S)$) from $\mathcal{D}$ (see Definition 2) upon obtaining a positive (resp. negative) test result for $V \backslash S$ provides the true posterior distribution. This is established in Lemma 2. We then bound the expected number of tests done in Stage 1 of the algorithm by $\frac{1}{\log \frac{1}{(1-c)}} H(X)$ (see Lemma 3) and the number of individual tests done in Stage 2 by $\frac{\tilde{\mu}}{1-2c}$ (see Lemma 4) to conclude the proof of Lemma 1.

a) Posterior Update: We first need the following lemma to justify the correct operation of Algorithm 1 in removing edges after each test. In particular, The target edge e^* is never removed and each test indicates whether the target edge is in $E(S)$ or $E \setminus E(S)$. Building on this, we further prove that the update rule utilized in Algorithm 1 computes the posterior probabilities given all the previous tests.

Lemma 2. *Consider a distribution $\mathcal{D}$ over the edge set E and a test $T = V \setminus S$ with result r. If $r = 1$, the posterior probability is obtained by removing $E(S)$ according to Definition 2. If $r = 0$, the posterior probability is obtained by removing $E \setminus E(S)$ according to Definition 2.*

b) Bounding Tests in Stage 1 and Stage 2: The above lemma implies that if both $E(S)$ and $E \backslash E(S)$ have a moderate mass (i.e., $w(S) \geq c$ and $1 - w(S) \geq c$), regardless of the result, the test is informative. This is quantified to limit the number of tests for Stage 1 of the algorithm:

Lemma 3. *Before line 15, the expected number of tests that Algorithm 1 performs is bounded by $\mathbb{E}[L_1] \leq \frac{1}{\log \frac{1}{(1-c)}} H(X)$.*

Algorithm 1 stops performing in Stage 1 when the greedy process of finding S, $c \leq w(S) \leq 1-c$, fails in line 12, and the nodes out of S tests negative in line 14. The following proposition shows that the remaining set of nodes are partitioned into "high probability nodes" and "low probability nodes".

Proposition 1. *When Algorithm 1 reaches 13, we have (i) $w(S) > 1 - c$, (ii) $Pr(\exists v \in V \setminus S : v \in e^*) < c$ where e^* is the target edge, and (iii) $\forall v \in S : q_v > 1 - 2c$.*

At this point, in Stage 2, we resort to individual testing in Line 15. We next bound $\mathbb{E}[L_2]$ based on the expected number of infected nodes $\tilde{\mu}$ at the "stopping time".

Lemma 4. *Let $\tilde{\mu} = \mu_{(\mathbf{T_1},\mathbf{r_1}),\ldots,(\mathbf{T_{L_1}},\mathbf{r_{L_1}})}$ be the expected number of infected nodes at the beginning of line 15 where $(\mathbf{T_i}, \mathbf{r_i}), 1 \leq i \leq L_1$ are random variables indicating the prior tests and results. Then the algorithm performs $L_2 \leq \frac{\tilde{\mu}}{1-2c}$ tests in line 15.*

A direct implication of Lemma 1 is that when the size of the edges is constrained so that $\mu \approx \tilde{\mu}$, then $O(H(X) + \mu)$ tests are achievable in expectation. This is quantified next.

Theorem 2. *If $\forall e \in E : f_1(n) \leq |e| \leq f_2(n)$, Algorithm 1 finds e^* with $\frac{1}{\log \frac{1}{(1-c)}} H(X) + \frac{f_2(n)}{f_1(n)(1-2c)} \mu$ tests in expectation.*

While Theorem 2 imposes a constraint on the edge sizes and finds the infected set with probability 1, the constraint can be relaxed if a small probability of error (ϵ, fixed or approaching zero) can be tolerated. We use Markov Lemma to bound the probability that $|e^*|$ exceeds μ/ϵ. By removing those edges for which $|e| > \mu/\epsilon$, we ensure $\tilde{\mu} < \mu/\epsilon$ and get the following.

Theorem 3. *There is an algorithm that performs $\mathbb{E}[L_1 + L_2] \leq \frac{1}{\log \frac{1}{(1-c)}} H(X) + \frac{\mu}{\epsilon(1-2c)}$ tests in expectation and returns the correct edge e^* with probability $1 - \epsilon$.*

In the above Theorem, there is a trade-off between the probability of error and the expected number of tests. With ϵ error, the second term in expectation increases with $1/\epsilon$. So if we set $\epsilon \to 0$ when $n \to \infty$, for example $\epsilon = 1/\log n$, we incur a multiplicative factor of $\log n$ on the second term. The above theorem can be further improved if $|e^*|$ is concentrated around μ. In this case, the dependency on $1/\epsilon$ becomes slower. For example, if the concentration is exponential, for instance, we have $Pr(|e^*| > 2\mu) < 2^{-\Omega(n)}$, then we can show that with probability $1 - 2^{-\Omega(n)}$ the algorithm recovers e^* and the expected number of tests is $\mathbb{E}[L_1 + L_2] \leq \frac{1}{\log \frac{1}{(1-c)}} H(X) + \frac{2\mu}{(1-2c)}$. More details are provided in [27, Section 4].

V. Discussion

In this work, we studied the problem of group testing with arbitrary correlation using hypergraphs. We proposed an adaptive algorithm and provided theoretical guarantees on its performance as a function of the entropy of the model as well as the expected number of infected nodes.

Our correlation model contains prior models in [1], [17], [19], [23], [24] as a special case and the proposed algorithm can recover/enhance some of their results. For instance, in [17], independent GT with individual infection probabilities p_i is considered. Applying Theorem 3 yields the guarantee of $O(H(X) + \frac{\mu}{\epsilon})$ tests in expectation, recovering infected nodes with probability $1-\epsilon$. For $\mu \gg 1$, this becomes $O(H(X)+\mu)$, because the number of infections is concentrated around μ, aligning with [17], refer to [27, Section 5.1] for more details.

While we discussed conditions under which our algorithm and its corresponding performance upper bound are near-optimal, it remains open whether our algorithm has an optimal performance. Interestingly, in contrast to the classical (independent) group testing, entropy is not a tight lower bound as we discuss in Example 1. Future works include providing tighter lower/upper bounds for the adaptive case. More generally, devising non-adaptive algorithms with near-optimal performance guarantees is another topic of interest. Additionally, while our current algorithm's complexity is polynomial in the number of edges, future efforts could explore more efficient polynomial algorithms in the number of nodes.

REFERENCES

[1] M. Gonen, M. Langberg, and A. Sprintson, "Group testing on general set-systems," in *2022 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2022, pp. 874–879.

[2] A. Ibarrondo, H. Chabanne, V. Despiegel, and M. Önen, "Grote: Group testing for privacy-preserving face identification," in *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy*, 2023, pp. 117–128.

[3] J. Zhang and L. S. Heath, "Adaptive group testing strategy for infectious diseases using social contact graph partitions," *Scientific Reports*, vol. 13, no. 1, p. 12102, 2023.

[4] M. Cheraghchi, R. Gabrys, and O. Milenkovic, "Semiquantitative group testing in at most two rounds," in *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021, pp. 1973–1978.

[5] M. Cheraghchi, A. Karbasi, S. Mohajer, and V. Saligrama, "Graph-constrained group testing," *IEEE Transactions on Information Theory*, vol. 58, no. 1, pp. 248–262, 2012.

[6] F. Iliopoulos and I. Zadik, "Group testing and local search: is there a computational-statistical gap?" in *Conference on Learning Theory*. PMLR, 2021, pp. 2499–2551.

[7] A. Coja-Oghlan, O. Gebhard, M. Hahn-Klimroth, A. S. Wein, and I. Zadik, "Statistical and computational phase transitions in group testing," in *Conference on Learning Theory*. PMLR, 2022, pp. 4764–4781.

[8] H. Wang, R. Gabrys, and V. Guruswami, "Quickly-decodable group testing with fewer tests: Price-ett's nonadaptive splitting with explicit scalars," in *IEEE International Symposium on Information Theory, ISIT 2023, Taipei, Taiwan, June 25-30, 2023*. IEEE, 2023, pp. 1609–1614. [Online]. Available: https://doi.org/10.1109/ISIT54713.2023.10206843

[9] S. Ahn, W.-N. Chen, and A. Özgür, "Noisy adaptive group testing for community-oriented models," in *2023 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2023, pp. 1621–1626.

[10] C. Yao, P. Nikolopoulos, and C. Fragouli, "A diagonal splitting algorithm for adaptive group testing," *arXiv preprint arXiv:2305.06737*, 2023.

[11] M. Aldridge, O. Johnson, and J. Scarlett, "Group testing: An information theory perspective," 2019.

[12] M. Aldridge, "Rates of adaptive group testing in the linear regime," in *2019 IEEE international symposium on information theory (ISIT)*. IEEE, 2019, pp. 236–240.

[13] H.-Y. Li and K. Guo, "Blood group testing," *Frontiers in Medicine*, vol. 9, p. 827619, 2022.

[14] G. Daniels, "An overview of blood group genotyping," *Annals of Blood*, vol. 8, 2023.

[15] D. Hong, R. Dey, X. Lin, B. Cleary, and E. Dobriban, "Group testing via hypergraph factorization applied to covid-19," *Nature communications*, vol. 13, no. 1, p. 1837, 2022.

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

[17] T. Li, C. L. Chan, W. Huang, T. Kaced, and S. Jaggi, "Group testing with prior statistics," in *2014 IEEE International Symposium on Information Theory*. IEEE, 2014, pp. 2346–2350.

[18] P. Nikolopoulos, S. R. Srinivasavaradhan, T. Guo, C. Fragouli, and S. Diggavi, "Group testing for connected communities," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2341–2349.

[19] H. Nikpey, J. Kim, X. Chen, S. Sarkar, and S. S. Bidokhti, "Group testing with correlation under edge-faulty graphs," *arXiv preprint arXiv:2202.02467*, 2022.

[20] B. Arasli and S. Ulukus, "Group testing with a graph infection spread model," *Information*, vol. 14, no. 1, p. 48, 2023.

[21] F. Xu, S.-I. Azuma, R. Ariizumi, and T. Asai, "Performance limitation of group testing in network failure detection," *IEEE Access*, 2023.

[22] B. Arasli and S. Ulukus, "Group testing with a graph infection spread model," *arXiv preprint arXiv:2101.05792*, 2021.

[23] P. Nikolopoulos, S. R. Srinivasavaradhan, T. Guo, C. Fragouli, and S. Diggavi, "Community-aware group testing," *IEEE Transactions on Information Theory*, 2023.

[24] S. Ahn, W.-N. Chen, and A. Özgür, "Adaptive group testing on networks with community structure: The stochastic block model," *IEEE Transactions on Information Theory*, 2023.

[25] E. Karimi, A. Heidarzadeh, K. R. Narayanan, and A. Sprintson, "Noisy group testing with side information," in *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2022, pp. 867–871.

[26] V. Brault, B. Mallein, and J.-F. Rupprecht, "Group testing as a strategy for covid-19 epidemiological monitoring and community surveillance," *PLoS computational biology*, vol. 17, no. 3, p. e1008726, 2021.

[27] H. Nikpey, S. Sarkar, and S. Saeedi Bidokht, "Generalized group testing using hypergraphs," *arXiv preprint*, 2024, https://www.seas.upenn.edu/~saeedi/GTcorrelation.pdf.