Efficiently Decodable Error-Correcting List Disjunct Matrices and Applications
(Extended Abstract)

Hung Q. Ngo[1], Ely Porat[2], and Atri Rudra[1,⋆]

[1] Department of CSE, University at Buffalo, SUNY, Buffalo, NY, 14260, USA
[2] Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel

Abstract. A (d, ℓ)-list disjunct matrix is a non-adaptive group testing primitive which, given a set of items with at most d "defectives," outputs a superset of the defectives containing less than ℓ non-defective items. The primitive has found many applications as stand alone objects and as building blocks in the construction of other combinatorial objects.

This paper studies error-tolerant list disjunct matrices which can correct up to e_0 false positive and e_1 false negative tests in sub-linear time. We then use list-disjunct matrices to prove new results in three different applications.

Our major contributions are as follows. (1) We prove several (almost)-matching lower and upper bounds for the optimal number of tests, including the fact that $\Theta(d \log(n/d) + e_0 + de_1)$ tests is necessary and sufficient when $\ell = \Theta(d)$. Similar results are also derived for the disjunct matrix case (i.e. $\ell = 1$). (2) We present two methods that convert error-tolerant list disjunct matrices in a *black-box* manner into error-tolerant list disjunct matrices that are also *efficiently decodable*. The methods help us derive a family of (strongly) explicit constructions of list-disjunct matrices which are either optimal or near optimal, and which are also efficiently decodable. (3) We show how to use error-correcting efficiently decodable list-disjunct matrices in three different applications: (i) explicit constructions of d-disjunct matrices with $t = O(d^2 \log n + rd)$ tests which are decodable in poly(t) time, where r is the maximum number of test errors. This result is optimal for $r = \Omega(d \log n)$, and even for $r = 0$ this result improves upon known results; (ii) (explicit) constructions of (near)-optimal, error-correcting, and efficiently decodable monotone encodings; and (iii) (explicit) constructions of (near)-optimal, error-correcting, and efficiently decodable multiple user tracing families.

1 Introduction

The basic objective of *group testing* is to figure out a subset of "defective items" in a large item population by performing tests on subsets of items. The manifestation of "defective" and "tests" depends on the application. For most of this paper we will consider the basic interpretation where we have a universe $[n]$ of

⋆ Supported by NSF CAREER grant CCF-0844796.

L. Aceto, M. Henzinger, and J. Sgall (Eds.): ICALP 2011, Part I, LNCS 6755, pp. 557–568, 2011.

items and some subset $S \subset [n]$ of at most d *defectives* (also interchangeably called *positives*). Every (group) test is a subset $T \subseteq [n]$, which results in a *positive outcome* if some defective is in T and a *negative outcome* when T contains no defectives. In many applications, *non-adaptive* group testing is required, where one cannot use one test's outcome to design another test. Non-adaptive group testing ($NAGT$) has found applications in drug and DNA library screening [18], live baiting of DoS attackers [16], data forensics [12] and data streams [4], among others. See the standard monograph on group testing for more details [6].

The first objective in the design of such NAGT primitives is to minimize the number of tests necessary to identify (or *decode*) all the defectives. A NAGT strategy with t tests on n items can be represented by a $t \times n$ binary matrix $\mathbf{M}$ where each row is the incidence vector of the corresponding test. For unique decoding of up to d defectives, it is necessary that all the unions of up to d columns of $\mathbf{M}$ have to be distinct. Such a matrix is said to be *d-separable*. It has been known for a long time that the optimal number of rows of a d-separable matrix is between $\Omega(d^2 \log n / \log d)$ [8] and $O(d^2 \log(n/d))$ [6].

The second objective is to explicitly construct disjunct matrices with as few tests as possible. Recently, a $O(nt)$-time explicit construction attaining the $t = O(d^2 \log(n))$-bound has also been found [20]. No strongly explicit construction matching the bound is known.[1]

The third objective is to decode efficiently. The brute-force algorithm is too slow as it goes through all possible $\binom{n}{\leq d} = O(n^d)$ choices for the defective set. Some NAGT strategies, however, allow a very simple $O(nt)$-time decoding algorithm to work: the decoder simply eliminates items belonging to negative tests and returns the remaining items. We shall refer to this decoder as the *naive decoder*. A NAGT matrix is said to be *d-disjunct* iff the naive decoder works on all possible inputs of up to d defectives. While disjunct matrices are a stronger notion than separable matrices, they have asymptotically the same number of tests [6]. Thus, we went from $O(n^d)$ down to $O(nt)$-decoding time "for free."

The time complexity of $O(nt)$ is reasonable for most of the "traditional" algorithmic applications. However with the proliferation of massive data sets and their numerous applications, the decoding time of $O(nt)$ is no longer good enough because the number of items n is prohibitively large. For example, in typical data stream applications a running time of $\text{poly}(t)$ (with $t = O(d^2 \log n)$ tests in the best case) for moderate values of d would imply an exponential improvement in the running time. The question of constructing efficiently decodable disjunct matrices was first explicitly raised by Cormode and Muthukrishnan [4]. Recently, Indyk, Ngo and Rudra [14] presented a randomized construction of d-disjunct matrices with $t = O(d^2 \log(n))$ tests that could be decoded in time $\text{poly}(t)$. They also derandomized their construction for $d \leq O(\log n / \log \log n)$. Our construction in this paper removes the above constraint on d. We thus can get further

[1] Throughout this paper we will call a $t \times n$ matrix strongly explicit if any column of the matrix can be constructed in time $\text{poly}(t)$. A matrix will be called explicit if it can be constructed in time $\text{poly}(t, n)$.

down to poly(t) decoding time "for free." Henceforth, "efficient decoding" means decoding in poly(t)-time.

The fourth objective is to correct errors in test outcomes. In many applications such as drug discovery and DNA library screening, faulty test outcomes are unavoidable [15]. Or, when heavy-hitters in a data-stream are identified using group testing, non-heavy hitter elements might generate false positive tests if the small-tail property is not satisfied [4]. This paper essentially obtains the above "for free" result even *with* the additional error-correcting requirement.

Our NAGT results crucially use as a building block a weaker notion of disjunct matrices called list disjunct matrices, which turns out to have many other useful applications as well.

1.1 List Disjunct Matrices and Our Main Results

Similar to list-decoding, if we relax the exact decoding requirement and only require the NAGT primitive to output a bounded super-set of all the defectives, then separable/disjunct matrices become list-separable/disjunct matrices. Roughly, a (d, ℓ)-list disjunct matrix is one where the naive decoder always outputs a super-set of the defectives containing less than ℓ other non-defective items. The name "list disjunct" was coined in [14], though it was previously studied under the different names: (d, n, ℓ)-*super-imposed codes* in [7,5], *list-decoding super-imposed codes* of strength d and list-size ℓ in [21].[2] We stick with the term "list disjunct matrices" in this paper. Shortly prior to [14], Cheraghchi [3] studied the notion of *error-correcting measurement matrices* for d-sparse vectors, which are slightly more general than the notion of *list-separable matrices*. It can be shown that list-separable matrices are equivalent to list-disjunct matrices with a slight loss in parameters. Hence, the results in [3], though shown for list-separable matrices, apply for list-disjunct matrices as well. Conversely, our bounds also apply to error-correcting measurement matrices. Our lower bounds slightly improve lower bounds in [3].

List-disjunct matrices were used in [14] to construct efficiently decodable disjunct matrices. They also presented a "stand alone" application of list disjunct matrices in constructing sparsity separators, which were used by Ganguly [11] to design data stream algorithms for the sparsity problem. Rudra and Uurtamo [22] used these objects to design data stream algorithms for tolerant testing Reed-Solomon codes. As observed in De Bonis et. al. [5], these objects can also be used to construct optimal two stage group testing algorithms.

List disjunct matrices are also similar to other combinatorial objects such as selectors [13] and multi-user tracing families [2]. Selectors have found numerous applications such as broadcasting in unknown directed networks and designing tests for coin-weighting problems [13] as well as designing optimal two stage group testing schemes [5]. Multi-user tracing families were used to construct monotone encodings in [2]. Monotone encodings can be used for designing secure vote storage systems [17].

[2] The authors of [14] were not aware of these previous works.

Given the various applications of list disjunct matrices, it is very natural to ask if one could (explicitly) construct list disjunct matrices that are also efficiently decodable. Indyk, Ngo and Rudra [14] constructed efficiently decodable $(d, O(d^{1+\epsilon}))$-list-disjunct matrices with $O(d^{1+\epsilon} \log^{1+1/\epsilon} n)$ tests $(\forall \epsilon > 0)$. Cheraghchi [3] constructed efficiently decodable $(d, O(d))$-list-separable matrices (and thus, due to their almost-equivalence, $(d, O(d))$-list-disjunct matrices) with $O(d^{3+\alpha+1/\alpha} \log^{1+1/\alpha} n)$ tests $(\forall \alpha > 0)$.

This paper improves upon both [14] and [3] by presenting efficiently decodable $(d, O(d))$-list-disjunct matrices with $O(d^{1+o(1)} \log n \log \log_d n)$ tests as well as $(d, O(d^{1+\epsilon}))$-list disjunct matrices with $O(d^{1+\epsilon} \log n)$ tests (for any $\epsilon > 0$). In addition, our matrices are also error-correcting. To state the results more precisely we briefly discuss the error-correcting matrices next.

Error-correcting versions of disjunct matrices have been studied a fair bit [6] but to the best of our knowledge the only existing work that considers error tolerant list-separable matrices is [3]. This paper studies the general notion of (d, ℓ, e_0, e_1)-list disjunct/separable matrices which are (d, ℓ)-disjunct/separable matrices capable of correcting up to e_0 false positives and e_1 false negatives.

We prove upper and lower bounds on the optimal number of rows of a (d, ℓ, e_0, e_1)-list disjunct matrix. The bounds are tight for sufficiently large $e_0 + de_1$. (Our lower bounds are slightly better than those in [3].)

We then show how to construct error-tolerant and efficiently decodable list-disjunct matrices by presenting two general procedures which – in a *black-box* fashion – convert any (d, ℓ, e_0, e_1)-list disjunct matrices into (d, ℓ, e_0', e_1')-list disjunct matrices that are also efficiently decodable with a mild blow-up in the number of tests. Note that we essentially show how to convert a combinatorial guarantee into an algorithmic guarantee, which for combinatorial objects (e.g. codes) is generally not obvious.

One of our conversion procedures provides a tradeoff between the blow-up in the number of tests vs. the gain in decoding time (from the simple linear time algorithm). Unfortunately, this procedure can only give $e_0' = e_0$ and $e_1' = e_1$ (even though the number of tests has gone up). Our other conversion procedure does not provide such a nice tradeoff but it does lead to efficient decoding with a mild blow-up in the number of tests. More importantly, the quantities e_0'/e_0 and e_1'/e_1 scale up linearly with the blow-up in the number of tests. This allows us to design error-tolerant $(d, d^{1+\epsilon}, e_0, e_1)$-list disjunct matrices that for large values of $e_0 + de_1$ have essentially all the nice properties one can wish for: (i) Optimal number of tests; (ii) Strongly explicit construction and (iii) Efficiently decodable.

1.2 Applications and Other Results

(Near) Optimal, explicit, and error-correcting disjunct matrices. Constructions of efficiently decodable list disjunct matrices lead to constructions of efficiently decodable d-disjunct matrices with the best known $O(d^2 \log n)$ number of tests. This result settles an open question from [14]. The black-box conversion procedures also work for disjunct matrices. We prove a similar optimal result as in the

list-disjunct case where a disjunct matrix is explicit, error-correcting, efficiently decodable, and has the best known number of tests. In fact, when the number of errors is sufficiently large, our error-tolerant disjunct matrices also have the optimal number of tests. This result points out the following somewhat surprising fact: our understanding of error-tolerant (list) disjunct matrices is essentially limited by our understanding of the traditional no error case. In other words, adding more errors only makes the problem "easier," which is not the case for related combinatorial objects such as codes.

Near Optimal, Efficiently Computable, and Efficiently Decodable Monotone Encodings. With a construction similar to that in Alon-Hod [2], we show that $(d, d/2)$-list disjunct $t \times n$ matrices imply a (n, d)-monotone encoding of length t, i.e. a monotone injective functions from subsets of size up to d of $[n]$ to t bits. By contrast, the Alon-Hod's construction used multi-user tracing families which are, in a sense, duals of list-disjunct matrices. Alon and Hod showed that optimal (n, d)-monotone encodings have length $t = \Theta(d \log (n/d))$, and presented a probabilistic construction with encoding and decoding time of $O(nd \log n)$. From our list-disjunct matrix constructions, we construct (n, d)-monotone encodings with length $t = O((d \log n)^{1+o(1)})$ that are both explicitly computable and decodable in poly(t)-time. Our result also hold for the error-tolerant monotone encodings. We also prove an almost matching lower-bound for the length of error-correcting monotone-encoding.

Near-Optimal Efficiently Decodable Multiple User Tracing Families. Given positive integers $u \leq d$, a (d, u)-multiuser tracing (MUT) family is a NAGT strategy (or matrix) which, given the test outcomes imposed by an arbitrary set of $v \leq d$ defectives, there is a decoding algorithm that outputs at least $\min(u, v)$ out of the v defectives. Thus, in a sense MUT is the dual of list-disjunct matrices. Alon-Asodi [1] proved that, given n, the smallest t for which a $t \times n$ (d, u)-MUT matrix exists must satisfy $\Omega\left(\left(d + \frac{u^2}{\log u}\right) \cdot \log n\right) \leq t \leq O\left((d + u^2) \log n\right)$. From our results on list-disjunct matrices, we show how to construct (d, u)-MUT matrix of size $t \times n$ with $t = O((d^{1+o(1)} + u^2) \log n + (e_0 + e_1)d)$ which are explicit, efficiently decodable and error tolerant. By removing the explicitness requirement, we can retain the other two properties and reduce t to essentially the optimal $O((d + u^2) \log n + (e_0 + e_1)d \log \log n)$.

2 Error-Correcting List-Disjunct/Separable Matrices

Let $\mathbf{M} = (m_{ij})$ be any binary matrix with t rows and n columns. Let $\mathbf{M}^j$ denote the jth column of $\mathbf{M}$. We can also think of the columns $\mathbf{M}^j$ as characteristic vectors of subsets of $[t]$, i.e. $\mathbf{M}^j = \{i \mid m_{ij} = 1\}$. Thus, overloading notation we will apply set operations on the columns of $\mathbf{M}$.

The key combinatorial structure we study in this paper is the error-tolerant version of the notion of *list-disjunct matrix*. Using a (d, ℓ)-list-disjunct $t \times n$ matrix $\mathbf{M}$ we can design a NAGT procedure for n items with at most d defectives.

The decoding algorithm simply eliminates any item which is present in any negative test. When $\mathbf{M}$ is (d, ℓ)-list-disjunct, this "naive decoder" returns a set R of (remaining) items containing all defectives and less than ℓ extra (non-negative) items. In many applications, the test outcomes might have errors. The following combinatorial structure is a generalization of list-disjunct matrices which can correct up to e_0 *false positives* and e_1 *false negatives* in test outcomes.

Definition 1. *Let $d \geq 1, \ell \geq 1, e_0 \geq 0, e_1 \geq 0$, and $n \geq d+\ell$ be given integers. A $t \times n$ binary matrix $\mathbf{M}$ is called a (d, ℓ, e_0, e_1)-list-disjunct matrix if $\mathbf{M}$ satisfies the following conditions. For any disjoint subsets $S, T \subset [n]$ such that $|S| = d, |T| = \ell$, and an arbitrary subset $X \subseteq \left(\bigcup_{j \in T} \mathbf{M}^j\right) \setminus \left(\bigcup_{j \in S} \mathbf{M}^j\right)$ of size $|X| \leq e_0$, there exists a column $\bar{j} \in T \setminus S$ such that $\left|\mathbf{M}^{\bar{j}} \setminus \left(X \cup \bigcup_{j \in S} \mathbf{M}^j\right)\right| \geq e_1 + 1$.*

Proposition 2. *Define the "naive decoder" be the algorithm which eliminates all items belonging to at least $e_1 + 1$ negative tests and returns the remaining items. If $\mathbf{M}$ is (d, ℓ, e_0, e_1)-list-disjunct, then the naive decoder returns a set R of items containing all the (at most d) defectives and at most $\ell - 1$ negative items, even if the test outcomes have up to e_0 false positives and e_1 false negatives.*

3 Intuition behind the Blackbox Conversion Procedures

The first conversion uses ideas similar to those used in [14]: given an error tolerant list disjunct matrix, we concatenate Parvaresh-Vardy codes [19] with it. PV codes have excellent list recoverability properties, which can be exploited to design efficient decoding procedure for the resulting matrix.

Our second conversion procedure is perhaps technically more interesting. The main idea behind the construction is as follows. Say we are trying to construct a (d, ℓ, e_0, e_1)-list-disjunct matrix $\mathbf{M}^*$ with n columns that is efficiently decodable from a family of matrices, where for any $i \geq d$, there is a (d, ℓ, e_0, e_1)-list-disjunct $t(i) \times i$ matrix (not necessarily efficiently decodable). Towards efficient decoding, say we somehow knew that all the positive items are contained in a subset $\mathcal{S} \subseteq [n]$. Then the naive decoder would run in time $O(t(n) \cdot |\mathcal{S}|)$, which would be sublinear if $|\mathcal{S}|$ and $t(n)$ are sufficiently small. The idea is to construct this small set $\mathcal{S}$ recursively.

Fix $n \geq d \geq 1$. Assume there exists a (d, ℓ, e_0, e_1)-list disjunct $t_1 \times \sqrt{n}$ matrix $\mathbf{M}^{(1)}$ that is efficiently decodable and let $\mathbf{M}^{(2)}$ be a (d, ℓ, e_0, e_1)-list disjunct $t_2 \times n$ matrix (that is not necessarily efficiently decodable). Let $\mathbf{M}^L$ be the $t_1 \times n$ matrix where the ith column (for $i \in [n]$) is identical to the jth column of $\mathbf{M}^{(1)}$ such that the *first* $\frac{1}{2} \log n$ bits of i is j (where we think of i and j as their respective binary representations). Similarly, let $\mathbf{M}^R$ be the $t_1 \times n$ matrix where the *last* $\frac{1}{2} \log n$ bits of i is j.

Let $S \subseteq [n], |S| \leq d$, be an arbitrary set of positives. Let the vector $\mathbf{r}^L$ ($\mathbf{r}^R$, resp.) be the vector that results from applying $\mathbf{M}^L$ ($\mathbf{M}^R$, resp.) on S. Note that $\mathbf{r}^L$ and $\mathbf{r}^R$ might be different from the unions $\cup_{j \in S} (\mathbf{M}^L)^j$ and $\cup_{j \in S} (\mathbf{M}^R)^j$,

respectively, due to the $(e_0, e_1$-bounded) errors in test outcomes. Apply the decoding algorithm for $\mathbf{M}^{(1)}$ to $\mathbf{r}^L$ ($\mathbf{r}^R$, resp.) and obtain the set S_L (S_R, resp.) of $\frac{1}{2}\log n$-bit vectors such that, for every $i \in S$, the first (last, resp.) $\frac{1}{2}\log n$ bits of i belongs to S_L (S_R, resp.). In other words, $\mathcal{S} = S_L \times S_R$ contains all the indices $i \in S$. Further, note that both $|S_L|$ and $|S_R|$ have less than $d + \ell$ elements.

Now, our final matrix $\mathbf{M}^*$ is simple: just vertically stack $\mathbf{M}^L$, $\mathbf{M}^R$ and $\mathbf{M}^{(2)}$ together. Note that $\mathbf{M}^*$ is (d, ℓ, e_0, e_1)-list disjunct because $\mathbf{M}^{(2)}$ is (d, ℓ, e_0, e_1)-list disjunct. Finally, decoding $\mathbf{M}^*$ can be done efficiently: first decode the part of the result matrix corresponding to $\mathbf{M}^L$ and $\mathbf{M}^R$ to obtain S_L and S_R respectively – this is efficient as $\mathbf{M}^{(1)}$ is efficiently decodable. Finally computing the output item set (containing S) can be done with an additional $O(t_2 \cdot (d + \ell)^2)$-time as we only need to run the naive decoder for $\mathbf{M}^{(2)}$ over $\mathcal{S} = S_L \times S_R$. To achieve a tradeoff between the number of tests and the decoding time, we choose the parameters of the recursion more carefully.

Our conversion of a disjunct matrix into an efficiently decodable one is very simple: stack an efficiently decodable $(d, \text{poly}(d))$-list disjunct matrix on a d-disjunct matrix. Decoding the result vector from the list disjunct matrix gives a subset of size $\text{poly}(d)$ that contains all the defective items. We then run the naive decoder for the disjunct matrix on this set of possibilities leading to an overall efficient decoding algorithm. Since there is an $\Omega(d/\log d)$ gap in the number of tests needed in a list disjunct matrix and a disjunct matrix, as long as we have an efficiently decodable list disjunct matrix with $o(d^2 \log n/\log d)$ tests, we are fine. Indeed, we get such matrices from our construction mentioned earlier. To obtain the efficiently decodable matrix with $O(d^2 \log n)$ tests we use the d-disjunct matrix construction of Porat and Rothschild [20]. The same idea works for the error-correcting versions.

4 Bounds

Given d, ℓ, e_0, e_1, and n, let $t(d, \ell, e_0, e_1, n)$ denote the minimum number of rows t of a (d, ℓ, e_0, e_1)-list-disjunct matrix with t rows and n columns. It is not hard to see that every $(d, 1, e_0, e_1)$-list-disjunct matrix is the same as a $(d, 1, e_0 + e_1, 0)$-list-disjunct matrix, which is the same as a $(d, 1, 0, e_0 + e_1)$-list-disjunct matrix. Let $r = e_0 + e_1 + 1$. It is customary in the literature to call a $(d, 1, r - 1, 0)$-list-disjunct matrix a *d^r-disjunct matrix* [6]. To shorten the notations, let $t(d, r, n)$ denote $t(d, 1, e_0, e_1, n)$ for the disjunct case, where $r = e_0 + e_1 + 1$.

The following lower bound for (d, ℓ)-list-disjunct matrices is better than the similar bound proved in [5] in two ways: (1) the actual bounds are slightly better, and (2) the bound in [5] requires a precondition that $n > d^2/(4\ell)$ while ours does not. We make use of the argument from Erdős-Frankl-Füredi [9, 10], while [5] uses the argument from Ruszinkó [23]. The bound helps prove Theorem 4, which is tight when $\ell = \Theta(d)$.

Lemma 3. *For any n, d, ℓ with $n \geq d+\ell$, we have $t(d, \ell, 0, 0, n) > d \log\left(\frac{n}{d+\ell-1}\right)$. When $d \geq 2\ell$, we have $t(d, \ell, 0, 0, n) > \frac{\lfloor d/\ell \rfloor (d+2-\ell)}{2\log(e\lfloor d/\ell \rfloor (d+2-\ell)/2)} \log\left(\frac{n-d-2\ell+2}{\ell}\right)$.*

Theorem 4. *For any non-negative integers d, ℓ, e_0, e_1, n where $n \geq d + \ell$, we have $t(d, \ell, e_0, e_1, n) = \Omega\left(d \log \frac{n}{d+\ell-1} + e_0 + de_1\right)$. In particular, $t(d, \Theta(d), e_0, e_1, n) = \Omega\left(d \log(n/d) + e_0 + de_1\right)$. Furthermore, when $d \geq 2\ell$ we have $t(d, \ell, e_0, e_1, n) = \Omega\left(\frac{d^2/\ell}{\log(d^2/\ell)} \log \frac{n-d}{\ell} + e_0 + de_1\right)$.*

Theorem 5. *Let n, d, ℓ, e_0, e_1 be given non-negative integers. If $\ell = \Omega(d)$, then $t(d, \ell, e_0, e_1, n) = O\left(d \log(n/d) + e_0 + de_1\right)$. In particular, when $l = \Theta(d)$ we have a precise characterization $t(d, \ell, e_0, e_1, n) = \Theta\left(d \log(n/d) + e_0 + de_1\right)$.*

Theorem 6. *For the d^r-disjunct matrices, we have $t(d, r, n) = O\left(d^2 \log \frac{n}{d} + rd\right)$.*

5 Constructions

We will need the following existing results from the literature.

Theorem 7 ([3]). *Let $1 \leq d \leq n$ be integers. Then there exists a strongly-explicit $t \times n$ matrix that is $(d, O(d), \alpha t, \Omega(t/d))$-list disjunct (for any constant $\alpha \in (0, 1)$) with $t = O(d^{1+o(1)} \log n)$ rows.*

Theorem 8 ([14]). *Let $1 \leq d \leq n$ be integers. Then there exists a strongly-explicit $t \times n$ matrix that is $(d, \delta d, \Omega(t), \Omega(t/d))$-list disjunct (for any constant $\delta > 0$) with $t = O((d \log n)^{1+o(1)})$ rows.*

5.1 Black-Box Conversion Using List Recoverability

Our first black-box procedure converting any error tolerant list disjunct matrix into one that is also efficiently decodable (with a mild sacrifice in some parameters) is based on concatenating PV codes [19] with the given list disjunct matrix.

Theorem 9. *Let $\ell, d \geq 1$, $e_0, e_1 \geq 0$ be integers. Assume that for every $Q \geq d$, there exists a (d, ℓ, e_0, e_1)-list-disjunct $\bar{t}(d, \ell, e_0, e_1, Q) \times Q$ matrix. For every $s \geq 1$ and every $n \geq d$, define $A(d, l, s) = 3(d+l)^{1/s}(s+1)^2$. Let k be minimum integer such that $2k \log(kA(d, l, s)) \geq \log n$, and q be the minimum power of 2 such that $q \geq kA(d, l, s)$. Then, there exists a $(d-1, L, e_0', e_1')$-list disjunct $t' \times n$ matrix with the following properties:*

(i) $t' = O\left(s^2 \cdot (d+\ell)^{1/s} \cdot \left(\frac{\log n}{\log q}\right) \cdot \bar{t}(s)\right)$, *where $\bar{t}(s)$ is shorthand for $\bar{t}(d, \ell, e_0, e_1, q^s)$*
(ii) $e_0' = \gamma e_0$ *and* $e_1' = \gamma e_1$ *where* $\gamma = \Theta(t'/\bar{t}(s))$.
(iii) $L = s^{O(s)} \cdot (d+\ell)^{1+1/s}$.
(iv) It is decodable in time $(t')^{O(s)}$.

Combining Theorem 9 and Theorem 7, we get the following result.

Corollary 10. *Let $\epsilon > 0$ be a real number and let $1 \leq d \leq n$ be integers. Then there exists a strongly-explicit $t \times n$ matrix that is $(d, (1/\epsilon)^{O(1/\epsilon)} \cdot d^{1+\epsilon}, \Omega(t), \Omega(t/d))$-list disjunct with $t = (1/\epsilon)^{O(1/\epsilon)} \cdot d^{1+\epsilon} \cdot \log n$ rows that can be decoded in time $t^{O(1/\epsilon)}$.*

5.2 Black-Box Conversion Using Recursion

Unlike the previous construction, the second procedure gives a more general tradeoff between the blow-up in the number of tests and the resulting decoding time. On the other hand, this conversion uses multiple matrices from a given family of error-tolerant matrices unlike the previous procedure, which only used one error-tolerant list disjunct matrix.

Theorem 11. *Let $n \geq d \geq 1$ be integers. Assume for every $i \geq d$, there is a (d, ℓ, e_0, e_1)-list disjunct $t(i) \times i$ matrix $\mathbf{M}_i$ for integers $1 \leq \ell \leq n - d$ and $e_0, e_1 \geq 0$. Let $1 \leq a \leq \log n$ and $1 \leq b \leq \log n/a$ be integers. Then there exists a $t_{a,b} \times n$ matrix $\mathbf{M}_{a,b}$ that is (d, ℓ, e_0, e_1)-list disjunct that can be decoded in time $D_{a,b}$ where*

$$t_{a,b} = \sum_{j=0}^{\lceil \log_b(\frac{\log n}{a}) \rceil - 1} b^j \cdot t\left(\sqrt[b^j]{n}\right) \tag{1}$$

and

$$D_{a,b} = O\left(t_{a,b} \cdot \left(\frac{\log n \cdot 2^a}{a} + (d + \ell)^b\right)\right). \tag{2}$$

Finally, if the family of matrices $\{\mathbf{M}_i\}_{i \geq d}$ is (strongly) explicit then so is $\mathbf{M}_{a,b}$.

The bound in (1) is somewhat unwieldy. We note in Corollary 12 that when $t(i) = d^x \log^y i$ for some reals $x, y \geq 1$, we can achieve efficient decoding with only a log-log factor increase in number of tests. Theorem 11 with $b = 2$ and $a = \log d$ implies the following:

Corollary 12. *Let $n \geq d \geq 1$ be integers and $x, y \geq 1$ be reals. Assume for every $i \geq d$, there is a (d, ℓ, e_0, e_1)-list disjunct $O(d^x \log^y i) \times i$ matrix for integers $1 \leq \ell \leq n - d$ and $e_0, e_1 \geq 0$. Then there exists a $t \times n$ matrix that is (d, ℓ, e_0, e_1)-list disjunct that can be decoded in $\mathrm{poly}(t, \ell)$ time, where*

$$t \leq O\left(d^x \cdot \log^y n \cdot \log \log_d n\right).$$

Finally, if the original matrices are (strongly) explicit then so is the new one.

Corollary 12 along with Theorems 8 and 7 imply the followings:

Corollary 13. *Let $1 \leq d \leq n$ be integers. For any constant $\delta > 0$ there exists a strongly-explicit $t \times n$ matrix that is $(d, \delta d, \Omega(t/\log\log n), \Omega(t/(d \log\log n)))$-list-disjunct with $t = O((d \log n)^{1+o(1)})$ rows and can be decoded in $\mathrm{poly}(t)$ time.*

Corollary 14. *Let $1 \leq d \leq n$ be integers. For any constant $\alpha \in (0, 1)$ there exists a strongly-explicit $t \times n$ matrix that is $(d, O(d), \alpha t/\log\log n, \Omega(t/(d \log\log n)))$-list disjunct with $t = O(d^{1+o(1)} \log n \log\log n)$ rows and can be decoded in $\mathrm{poly}(t)$ time.*

6 Applications

Efficiently decodable disjunct matrices. The following proposition along with Corollary 10 (with say $\epsilon = 1/2$) lead to the next theorem, a significant improvement over the similar result obtained in [14] that only worked for $d \leq O(\log n / \log\log n)$.

Proposition 15. *Let $n \geq d \geq 1$ be integers. Let M_1 be a (d, ℓ)-list disjunct $t_1 \times n$ matrix that can be decoded in time D. Let M_2 be a d-disjunct $t_2 \times n$ matrix. Then there exists a $(t_1 + t_2) \times n$ matrix that is d-disjunct and can be decoded in time $D + O((d+\ell) \cdot t_2)$. If both M_1 and M_2 are polynomial time constructible then so is the final matrix.*

Theorem 16. *Let $1 \leq d \leq n$. Then there exists a $t \times n$ d-disjunct matrix with $t = O(d^2 \log n)$ that can be decoded in $\mathrm{poly}(t)$ time. Further, the matrix can be computed in time $\tilde{O}(nt)$.*

(Near) Optimal Error-Tolerant, Efficiently Constructible, and Efficiently Decodable (List) Disjunct Matrices. We begin with simple observation that stacking i copies of a list disjunct matrix increases the error-tolerance parameters by a factor of i.

Proposition 17. *If there exists a (d, ℓ, e_0, e_1)-list disjunct matrix with t rows then there exists another $(d, \ell, \alpha \cdot e_0, \alpha \cdot e_1)$-list disjunct matrix with αt rows for any integer $\alpha \geq 1$.*

The surprising thing is that the result above leads to optimal construction of (list) disjunct matrices. In particular, applying Proposition 17 with Corollary 10 implies the following:

Corollary 18. *For every $\epsilon > 0$, there exists a strongly explicit $(d, (1/\epsilon)^{O(1/\epsilon)} \cdot d^{1+\epsilon}, r, r/d)$-list disjunct matrix with $O(t + r)$ rows, where $t = O((1/\epsilon)^{O(1/\epsilon)} \cdot d^{1+\epsilon} \cdot \log n)$ that can be decoded in time $r \cdot t^{O(1/\epsilon)}$.*

Note that Theorem 4 shows that the number of tests in the result above is optimal for $r \geq \Omega(t)$. We also get the following result with the construction in [20], Corollary 18, and Proposition 17. The result is optimal for $r = \Omega(d \log n)$, by Theorem 4.

Corollary 19. *Let $1 \leq d \leq n$ and $r \geq 1$ be integers. Then there exists a $t \times n$ d^r-disjunct matrix with $t = O(d^2 \log n + rd)$ decodable in $r \cdot \mathrm{poly}(t)$ time. Further, the matrix can be computed in time $\tilde{O}(nt)$.*

Error tolerant monotone encodings. A (one-sided) *r-error-correcting (n, d)-monotone encoding* is a monotone injective mapping from subsets of size up to d of $[n]$ to t bits, such that we can recover the correct subset even when up to r bits are flipped from 0 to 1. The number t is called the *length* of the encoding. This type of one-sided error holds true for the read-once memory environment where monotone encoding is applicable [17, 2].

Theorem 20. *Let $n \geq d$ be given positive integers. For each integer i, $0 \leq i \leq \log_2 d - 1$, let $\mathbf{A}_i$ be a $(d/2^i, d/2^{i+1}, r, 0)$-list-disjunct $t_i \times n$ matrix. From the matrices $\mathbf{A}_i$, we can construct a r-error-correcting (n, d)-monotone encoding with length $t = \sum_{i=0}^{\log_2 d - 1} t_i$ which can be encoded and decoded in time $O(nt)$. Furthermore, if the matrices $\mathbf{A}_i$ are strongly explicit then the monotone encoding is strongly explicit. If the list-disjunct matrices $\mathbf{A}_i$ can be decoded in time $T_i(n, d)$, then the monotone encoding can be computed in time $\sum_i T_i(n, d)$.*

Corollary 21. *There exist r-error-correcting (n, d)-monotone-encodings of length $t = O\left(d \log(n/d) + r \log d\right)$. Note that, this bound is best possible when $r = 0$ because the information theoretic bound is $\Omega(d \log(n/d))$.*

Finally, apply the list-disjunct matrices in Corollary 13 to Theorem 20, we obtain the following.

Corollary 22. *Given integers $n \geq d \geq 1$ and r, there exists a $\mathrm{poly}(t)$-time algorithm computing an r-error-correcting (n, d)-monotone-encoding with length $t = O\left((d \log n)^{1+o(1)} + r \log d \cdot \log\log n\right)$, which can be decoded in $\mathrm{poly}(t)$-time also.*

The following lower bound is off from the upper bound by a factor of about $\tilde{O}(\log d)$.

Proposition 23. *Suppose there exists a r-error-correcting (n, d)-monotone encoding, then the code length t has to satisfy $t = \Omega\left(d \log(n/d) + r \log\left(\frac{d \log(n/d)}{r}\right)\right)$.*

Efficiently Decodable Multiple User Tracing Family. The following results answer two open questions left in [1], concerning the explicit constructions of MUT families and the fast-decodability of such families. Furthermore, our results are error-correcting.

Theorem 24. *Given non-negative integers $e_0, e_1, u \leq d < n$, there is a randomized construction of $t \times n$ (d, u)-MUT matrix, which can correct e_0 false positives and e_1 false negatives, where $t = O((d + u^2) \log n + (e_0 + e_1)d)$. If we also want explicit construction along with efficient decoding, then t is slightly increased to $t = O((d^{1+o(1)} + u^2) \log n + (e_0 + e_1)d \log\log n)$.*

References

1. Alon, N., Asodi, V.: Tracing many users with almost no rate penalty. IEEE Trans. Inform. Theory 53(1), 437–439 (2007)
2. Alon, N., Hod, R.: Optimal monotone encodings. IEEE Transactions on Information Theory 55(3), 1343–1353 (2009)
3. Cheraghchi, M.: Noise-resilient group testing: Limitations and constructions. In: Kutyłowski, M., Charatonik, W., Gębala, M. (eds.) FCT 2009. LNCS, vol. 5699, pp. 62–73. Springer, Heidelberg (2009)

4. Cormode, G., Muthukrishnan, S.: What's hot and what's not: tracking most frequent items dynamically. ACM Trans. Database Syst. 30(1), 249–278 (2005)
5. De Bonis, A., Gąsieniec, L., Vaccaro, U.: Optimal two-stage algorithms for group testing problems. SIAM J. Comput. 34(5), 1253–1270 (electronic) (2005)
6. Du, D.Z., Hwang, F.K.: Combinatorial group testing and its applications, 2nd edn. Series on Applied Mathematics, vol. 12. World Scientific Publishing Co. Inc., River Edge (2000)
7. D'yachkov, A.G., Rykov, V.V.: A survey of superimposed code theory. Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform. 12(4), 229–242 (1983)
8. D'yachkov, A.G., Rykov, V.V., Rashad, A.M.: Superimposed distance codes. Problems Control Inform. Theory 18(4), 237–250 (1989)
9. Erdős, P., Frankl, P., Füredi, Z.: Families of finite sets in which no set is covered by the union of r others. Israel J. Math. 51(1-2), 79–89 (1985)
10. Füredi, Z.: On r-cover-free families. J. Combin. Theory Ser. A 73(1), 172–173 (1996)
11. Ganguly, S.: Data stream algorithms via expander graphs. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 52–63. Springer, Heidelberg (2008)
12. Goodrich, M.T., Atallah, M.J., Tamassia, R.: Indexing information for data forensics. In: Third International Conference on Applied Cryptography and Network Security (ANCS), pp. 206–221 (2005)
13. Indyk, P.: Explicit constructions of selectors and related combinatorial structures, with applications. In: SODA, pp. 697–704 (2002)
14. Indyk, P., Ngo, H.Q., Rudra, A.: Efficiently decodable non-adaptive group testing. In: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1126–1142 (2010)
15. Kainkaryam: Pooling in high-throughput drug screening. Current Opinion in Drug Discovery & Development 12(3), 339–350 (2009)
16. Khattab, S.M., Gobriel, S., Melhem, R.G., Mossé, D.: Live baiting for service-level dos attackers. In: INFOCOM, pp. 171–175 (2008)
17. Moran, T., Naor, M., Segev, G.: Deterministic history-independent strategies for storing information on write-once memories. Theory of Computing 5(1), 43–67 (2009)
18. Ngo, H.Q., Du, D.Z.: A survey on combinatorial group testing algorithms with applications to DNA library screening. In: Discrete Mathematical Problems with Medical Applications. DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 55, pp. 171–182. Amer. Math. Soc., Providence (2000)
19. Parvaresh, F., Vardy, A.: Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 285–294 (2005)
20. Porat, E., Rothschild, A.: Explicit non-adaptive combinatorial group testing schemes. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 748–759. Springer, Heidelberg (2008)
21. Rashad, A.M.: Random coding bounds on the rate for list-decoding superimposed codes. Problems Control Inform. Theory/Problemy Upravlen. Teor. Inform. 19(2), 141–149 (1990)
22. Rudra, A., Uurtamo, S.: Data stream algorithms for codeword testing. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6198, pp. 629–640. Springer, Heidelberg (2010)
23. Ruszinkó, M.: On the upper bound of the size of the r-cover-free families. J. Combin. Theory Ser. A 66(2), 302–310 (1994)