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 $\text{poly}(t, n)$.