Privacy in Index Coding: Improved Bounds and Coding Schemes

Mohammed Karmoose†, Linqi Song†, Martina Cardone‡, Christina Fragouli†
†University of California Los Angeles, Los Angeles, CA 90095 USA
‡University of Minnesota, Minneapolis, MN 55455 USA
Email: {mkarmoose, songlinqi}@ucla.edu, cardo089@umn.edu, christina.fragouli@ucla.edu

***Abstract*—It was recently observed in [1], that in index coding, learning the coding matrix used by the server can pose privacy concerns: curious clients can extract information about the requests and side information of other clients. One approach to mitigate such concerns is the use of k-limited-access schemes [1], that restrict each client to learn only part of the index coding matrix, and in particular, at most k rows. These schemes transform a linear index coding matrix of rank T to an alternate one, such that each client needs to learn at most k of the coding matrix rows to decode its requested message. This paper analyzes k-limited-access schemes. First, a worst-case scenario, where the total number of clients n is $2^T - 1$ is studied. For this case, a novel construction of the coding matrix is provided and shown to be order-optimal in the number of transmissions. Then, the case of a general n is considered and two different schemes are designed and analytically and numerically assessed in their performance. It is shown that these schemes perform better than the one designed for the case $n = 2^T - 1$.**

I. Introduction

It is well established that coding is necessary to optimally use wireless broadcasting for information transfer. The index coding framework, in particular, exemplifies the benefits of coding when using broadcast channels. In fact, by leveraging their side information, the requests of multiple clients can be simultaneously satisfied by a set of coded broadcast transmissions, the number of which could potentially be much smaller than uncoded information transfer [2].

However, as we observed in [1], [3], coding also poses privacy concerns: by learning the coding matrix, a curious client can infer information about the identities of the side information and request of other clients. In this paper, we build on the work in [1], [3] with the goal to offer improved constructions and bounds that enable to balance the trade-off between privacy and efficient broadcasting.

In an index coding setting, a server with m messages is connected to n clients via a lossless broadcast channel. Each client requests a specific message and may have a subset of the messages as side information. To satisfy all clients with the minimum number of transmissions T, the server can send coded broadcast transmissions; the clients then would use the coding matrix to decode their requests. In [1], we mitigated the aforementioned privacy risk by providing clients with access not to the entire coding matrix, but *only* to the rows required for them to decode their own requests. In fact, given a coding matrix that uses T transmissions to satisfy all clients, we can transform it into another coding matrix that uses $T_k \geq T$ transmissions to satisfy all clients, but where each client needs to learn only k rows of the coding matrix. In [1], we showed that the attained amount of privacy is dictated by k.

This formulation admits a geometric interpretation. In [2], it was shown that designing an index code is equivalent to the rank minimization of an $n \times m$ matrix $\mathbf{G}$, where the i-th row of $\mathbf{G}$ has certain properties which enable client i to recover its request. Assume that the rank of $\mathbf{G}$ is T; then, we can use as a coding matrix $\mathbf{A}$ any basis of this T-dimensional space. By doing so, client i can linearly combine some vectors of $\mathbf{A}$ to reconstruct the i-th row of $\mathbf{G}$. The geometric interpretation of our problem is therefore the following: given n distinct vectors in a T-dimensional space, represented by the rows of $\mathbf{G}$, we wish to find an overcomplete basis $\mathbf{A}_k$ of dimension $T_k \geq T$, such that each of the n vectors can be expressed as a linear combination of at most k of the $\mathbf{A}_k$ vectors.

In [1], we formalized the intuition that the achieved level of privacy can increase by decreasing the number k of rows of the coding matrix that a client learns. We also derived upper and lower bounds on T_k, with the former being independent of n. In this paper, our main contributions are as follows:

1) We derive an improved upper bound that again applies for all values of n, and show that, in contrast to the one in [1], it is order-optimal. Our upper bound is constructive, i.e., it provides a concrete construction of a coding matrix.
2) For general $n \leq 2^T - 1$, the previous construction does not always offer benefits over uncoded transmissions. For such cases, we propose two novel algorithms and assess their analytical and numerical performance. In particular, we show their superior performance over other schemes through numerical evaluations.

The paper is organized as follows. Section II formulates the problem and presents existing results. Section III provides a scheme for $n = 2^T - 1$. Section IV discusses special instances of the problem for a general n, while Section V presents upper bounds and algorithms. Section VI provides numerical evaluations, and finally Section VII discusses related work.

II. Problem Formulation and Previous Results

Notation. Calligraphic letters indicate sets; boldface lower case letters denote vectors and boldface upper case letters

The work of the authors was partially funded by NSF under Awards 1423271, 1527550 and 1740047.

indicate matrices; $|\mathcal{X}|$ is the cardinality of $\mathcal{X}$; $[n]$ is the set of integers $\{1, \cdots, n\}$; $\emptyset$ is the empty set; for all $x \in \mathbb{R}$, the floor and ceiling functions are denoted with $\lfloor x \rfloor$ and $\lceil x \rceil$, respectively; $\mathbf{0}_j$ is the all-zero row vector of dimension j; $\mathbf{0}_{i\times j}$ is the all-zero matrix of dimension $i \times j$; $\mathbf{1}_j$ denotes a row vector of dimension j of all ones.

Index Coding. We consider a setup similar to the one in [1]. We assume an index coding instance, where a server has a database $\mathcal{B} = \{\mathbf{b}_{\mathcal{M}}\}$ of m messages, with $\mathcal{M} = [m]$ being the set of message indices, and all messages $\mathbf{b}_j \in \mathbb{F}_2^F, j \in \mathcal{M}$, are F-long strings. The server is connected through a broadcast channel to a set of clients $\mathcal{C} = \{c_{\mathcal{N}}\}$, where $\mathcal{N} = [n]$ is the set of client indices, and $m \geq n$. Each client $c_i, i \in \mathcal{N}$, has a subset of the messages $\{\mathbf{b}_{\mathcal{S}_i}\}$, with $\mathcal{S}_i \subset \mathcal{M}$, as side information and requests a new message $\mathbf{b}_{q_i}$ with $q_i \in \mathcal{M} \setminus \mathcal{S}_i$ that it does not already have. A *linear index code* solution to the index coding instance is a designed set of broadcast transmissions that are linear combinations of the messages in $\mathcal{B}$. The linear index code can be represented as $\mathbf{AB} = \mathbf{Y}$, where $\mathbf{A} \in \mathbb{F}_2^{T\times m}$ is the coding matrix, $\mathbf{B} \in \mathbb{F}_2^{m\times F}$ is the matrix of all the messages and $\mathbf{Y} \in \mathbb{F}_2^{T\times F}$ is the resulting matrix of linear combinations. Upon receiving these transmissions, client $c_i, i \in \mathcal{N}$, employs linear decoding to retrieve $\mathbf{b}_{q_i}$. A linear index code with the minimum number of transmissions is called an *optimal* linear index code.

Problem Formulation. Designing the optimal linear index code is an NP-Hard problem, and therefore various algorithms exist for designing sub-optimal linear index codes (see Section VII). In this work, we are concerned with designing linear index codes that maintain higher privacy levels for the requests of clients. Our approach is based on using k-limited-access schemes [1]: given a coding matrix $\mathbf{A}$ of rank T, we wish to create an alternative index code $\mathbf{A}_k = \mathbf{PA}$, where $\mathbf{P} \in \mathbb{F}_2^{T_k\times T}$ has to be designed such that client $c_i, i \in \mathcal{N}$ can retrieve $\mathbf{b}_{q_i}$ using *at most* k vectors of $\mathbf{A}_k$, where $1 \leq k \leq T$. The value of T_k represents the number of transmissions associated with the alternative index code $\mathbf{A}_k$, and therefore our goal is to design $\mathbf{P}$ with minimum T_k. In order to create such a linear index code, we note that the coding matrix $\mathbf{A}$ allows client $c_i, i \in \mathcal{N}$ to retrieve $\mathbf{b}_{q_i}$ by a linear decoding operation expressed as $\mathbf{d}_i\mathbf{AB} = \mathbf{d}_i\mathbf{Y}$, where $\mathbf{d}_i \in \mathbb{F}_2^T$ is the decoding row vector of c_i. The resulting vector $\mathbf{g}_i = \mathbf{d}_i\mathbf{A}$ possesses certain properties which allows c_i to decode $\mathbf{b}_{q_i}$ using $\mathbf{b}_{\mathcal{S}_i}$ [2]. Therefore, an alternative index code $\mathbf{A}_k$ would still allow client c_i to decode $\mathbf{b}_{q_i}$ if it is able to reconstruct $\mathbf{g}_i$ using $\mathbf{A}_k$. Our problem can therefore be stated as follows: *Given $\mathbf{g}_i, i \in \mathcal{N}$, can we design a matrix $\mathbf{P}$, with T_k as small as possible, such that $\mathbf{g}_i, i \in \mathcal{N}$ can be reconstructed by adding at most k vectors out of $\mathbf{A}_k$?* Note that, by definition, $\mathbf{g}_i, i \in \mathcal{N}$ lie in the row span of $\mathbf{A}$. Since the rank of $\mathbf{A}$ is T, the maximum number of distinct $\mathbf{g}_i$ vectors is $2^T - 1$. Therefore, without loss of generality, we assume that $n \leq 2^T - 1$. We refer to the case where $n = 2^T - 1$ as *full-space covering*, and to the case where $n < 2^T - 1$ as *partial-space covering*.

Our previous work in [1] provided a lower bound on the minimum value of T_k, which we restate here for convenience.

Lemma II.1. *[1, Theorem III.1] Given an index coding matrix $\mathbf{A} \in \mathbb{F}_2^{T\times m}$ with $T \geq 2$, it is possible to transform it into $\mathbf{A}_k = \mathbf{PA}$ with $\mathbf{P} \in \mathbb{F}_2^{T_k\times T}$, such that each client can recover its request by combining at most k rows of it, if and only if*

$$T_k \geq T^\star = \min\left\{T_k : \sum_{i=1}^{k} \binom{T_k}{i} \geq n\right\} \overset{(a)}{\geq} T^{LB} = \frac{2^{\frac{T-1}{k}} k^{\frac{k-1}{k}}}{e}, \tag{1}$$

where (a) holds when $n = 2^T - 1$ and $k < \lceil T/2 \rceil$.

In addition, [1, Theorem III.1] provided a construction of a matrix $\mathbf{P}$ for which T_k is shown to have an exponent that is order-optimal for the full-space covering case and for some regimes of k. Differently, one contribution in this paper is a matrix construction that is order-optimal for any value of $1 \leq k < \lceil T/2 \rceil$[1]. This is described in the next section.

III. Improved Scheme for Full-Space Covering

Here we provide a novel scheme for the full-space covering case (i.e., $n = 2^T - 1$). This new scheme is order-optimal in the number of transmissions for the case when $1 \leq k < \lceil \frac{T}{2} \rceil$. This provides an improvement over the scheme presented in [1, Theorem III.1].

Theorem III.1. *For $n = 2^T - 1$ and $1 \leq k < \lceil \frac{T}{2} \rceil$ we have*

$$T_k \leq 2^{\lceil \frac{T}{k} \rceil} k. \tag{2}$$

Before providing the proof for Theorem III.1, which shows how the scheme is constructed, we analyze the performance of the scheme in comparison to the lower bound in (1). We do so in the next lemma (proof is in [4, Appendix 1]).

Lemma III.2. *For $1 \leq k < \lceil \frac{T}{2} \rceil$, we have $T_k = \Theta(2^{\frac{T}{k}} k)$.*

The main difference between Scheme-1 (in Theorem III.1) and Scheme-2 (in [1]) is as follows. Both schemes are designed by: (i) breaking the binary vector of length T into parts, (ii) providing all possible non-zero binary vectors that correspond to each part, and (iii) combining the solutions to reconstruct the original vector. However, the two schemes differ in the following: 1) Scheme-1 splits the vector into *larger* but *fewer* parts than Scheme-2, and 2) Scheme-1 aggregates the solutions *additively* while Scheme-2 aggregates them *multiplicatively*. While it is indeed true that providing all possible vectors for the parts in Scheme-1 would lead to larger partial solutions than those in Scheme-2, aggregating those solutions additively eventually leads to a smaller number of vectors than in Scheme-2. For example, assume $T = 20$ and $k = 5$; Scheme-1 consists of 75 vectors while Scheme-2 consists of 484 vectors.

The remainder of this section proves Theorem III.1 by showing how the scheme works (i.e., how $\mathbf{A}_k$ is constructed).

Example: We first show how the scheme is constructed via a small example, where $T = 8$ and $k = 3$. The idea is that, to

[1]The case $\lceil T/2 \rceil \leq k < T$ was solved in [1], where we showed that $T_k = \min\{T + 1, n\}$.

reconstruct a vector $\mathbf{vA}$, $\mathbf{v} \in \mathbb{F}_2^8$, we treat $\mathbf{v}$ as $k = 3$ disjoint parts; the first 2 are of length $\lceil \frac{T}{k} \rceil = 3$ and the remaining part is of length $T - (k-1)\lceil \frac{T}{k} \rceil = 2$. We then construct $\mathbf{P}$ as $k = 3$ disjoint sections, where each section allows us to reconstruct one part of the vector. Specifically, we construct

$$\mathbf{P} = \begin{bmatrix} \bar{\mathbf{B}}_1 & \mathbf{0}_{7\times 3} & \mathbf{0}_{7\times 2} \\ \mathbf{0}_{7\times 3} & \bar{\mathbf{B}}_2 & \mathbf{0}_{7\times 2} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & \bar{\mathbf{B}}_3 \end{bmatrix},$$

where $\bar{\mathbf{B}}_1 = \bar{\mathbf{B}}_2$ contain as rows all possible non-zero vectors of length 3, and $\bar{\mathbf{B}}_3$ contains as rows all possible vectors of length 2. Then, any vector $\mathbf{v}$ can be reconstructed by picking at most k vectors out of $\mathbf{P}$, one from each section. For example, let $\mathbf{v} = [0\,1\,0\,0\,1\,1\,1\,0]$. Then, this vector can be reconstructed by adding vectors number 2, 10 and 16 from $\mathbf{P}$.

Proof of Theorem III.1: Let $Q = \lfloor T / \lceil \frac{T}{k} \rceil \rfloor$ and $T_{\text{rem}} = T - Q\lceil \frac{T}{k} \rceil$. If k divides T, then $Q = k$, $T_{\text{rem}} = 0$, otherwise $Q \leq k-1$ and $T_{\text{rem}} \leq \lceil \frac{T}{k} \rceil$. Then, we can write $\mathbf{P} = \left[\mathbf{B}_1^T\, \mathbf{B}_2^T\, \cdots\, \mathbf{B}_Q^T\, \mathbf{B}_{Q+1}^T\right]^T$, where, for $i \in [Q]$, the matrix $\mathbf{B}_i$, of dimension $b_i \times T$, is constructed as follows

$$\mathbf{B}_i = \left[\mathbf{0}_{b_i \times (i-1)\lceil \frac{T}{k} \rceil} \quad \bar{\mathbf{B}}_i \quad \mathbf{0}_{b_i \times (Q-i)\lceil \frac{T}{k} \rceil} \quad \mathbf{0}_{b_i \times T_{\text{rem}}}\right],$$

where $\bar{\mathbf{B}}_i$, of dimension $b_i \times \lceil \frac{T}{k} \rceil$, has as rows all non-zero vectors of length $\lceil \frac{T}{k} \rceil$. Similarly, the matrix $\mathbf{B}_{Q+1}$, of dimension $b_{Q+1} \times T$, is constructed as follows

$$\mathbf{B}_{Q+1} = \left[\mathbf{0}_{b_{Q+1} \times Q\lceil \frac{T}{k} \rceil} \quad \bar{\mathbf{B}}_{Q+1}\right],$$

where $\bar{\mathbf{B}}_{Q+1}$, of dimension $b_{Q+1} \times T_{\text{rem}}$, has as rows all non-zero vectors of length T_{rem}.

In other words, the matrix $\mathbf{P}$ is constructed as a block-diagonal matrix, with the diagonal elements being $\bar{\mathbf{B}}_i$ for all $i \in [Q+1]$. Therefore, equation (2) holds by computing

$$T_k = \sum_{i=1}^{Q+1} b_i = Q\left(2^{\lceil \frac{T}{k} \rceil} - 1\right) + 2^{T_{\text{rem}}} - 1 \leq k 2^{\lceil \frac{T}{k} \rceil}.$$

What remains is to show that any vector $\mathbf{v} \in \mathbb{F}_2^T$ can be reconstructed by adding at most k vectors of $\mathbf{P}$. To show this, we can express it as $\mathbf{v} = [\mathbf{v}_1\, \cdots\, \mathbf{v}_{Q+1}]$ where $\mathbf{v}_i, i \in [Q]$ are parts of the vector $\mathbf{v}$ each of length $\lceil \frac{T}{k} \rceil$, while $\mathbf{v}_{Q+1}$ is the last part of $\mathbf{v}$ of length T_{rem}. Then, we can write $\mathbf{v} = \sum_{i \in \mathcal{K}(\mathbf{v})} \bar{\mathbf{v}}_i$, where $\bar{\mathbf{v}}_i = \left[\mathbf{0}_{(i-1)\lceil \frac{T}{k} \rceil} \quad \mathbf{v}_i \quad \mathbf{0}_{(Q-i)\lceil \frac{T}{k} \rceil}\, \mathbf{0}_{T_{\text{rem}}}\right]$ for $i \in [Q]$, $\bar{\mathbf{v}}_{Q+1} = \left[\mathbf{0}_{Q\lceil \frac{T}{k} \rceil} \quad \mathbf{v}_{Q+1}\right]$ and $\mathcal{K}(\mathbf{v}) \subseteq [Q+1]$ is the set of indices for which $\mathbf{v}_i$ is not all-zero. Then, according to the construction of $\mathbf{P}$, for all $i \in \mathcal{K}(\mathbf{v})$, the corresponding vector $\mathbf{v}_i$ is one of the rows in $\mathbf{B}_i$. The proof concludes by noting that $|\mathcal{K}(\mathbf{v})| \leq k$. ■

IV. Partial-Space Covering

Here we study some specific instances of the problem, which we will later use in our algorithms. We first represent the problem through a bipartite graph as follows. We assume that the rank of the matrix $\mathbf{G}$ is T. Then, there exists a set of T linearly independent vectors in $\mathbf{G}$; without loss of generality, denote them as $\mathbf{g}_1$ to $\mathbf{g}_T$. We can then represent the problem as a bipartite graph $(\mathcal{U} \cup \mathcal{V}, \mathcal{E})$ with $|\mathcal{U}| = T$ and $|\mathcal{V}| = n - T$, where $u_i \in \mathcal{U}$ represents the vector $\mathbf{g}_i$ for $i \in [T]$, $v_i \in \mathcal{V}$ represents the vector $\mathbf{g}_{i+T}$ for $i \in [n-T]$, and an edge exists from node u_i to node v_j if $\mathbf{g}_i$ is one of the component vectors of $\mathbf{g}_{j+T}$. Figure 1 shows an example of such graph, where $n = 9$ and $T = 6$. For instance, v_1 (i.e., $\mathbf{g}_7$) can be reconstructed by adding $u_i, i \in [4]$. Given a node s in the graph, we refer to the sets $\mathcal{O}_s$ and $\mathcal{I}_s$ as the *outbound* and *inbound* sets of s, respectively: the inbound set contains the nodes which have edges outgoing to node s, and the outbound set contains the nodes to which node s has outgoing edges. For instance, with reference to Figure 1, $\mathcal{O}_{u_1} = \{v_1, v_2, v_3\}$ and $\mathcal{I}_{v_1} = \{u_1, u_2, u_3, u_4\}$. For this particular example, there exists a scheme with $T_2 = 6$ which can reconstruct any vector with at most $k = 2$ additions. The matrix $\mathbf{A}_2$ which corresponds to this solution consists of the following vectors: $\mathbf{g}_1$, $\mathbf{g}_1+\mathbf{g}_2$, $\mathbf{g}_1+\mathbf{g}_2+\mathbf{g}_3$, $\mathbf{g}_1+\mathbf{g}_2+\mathbf{g}_3+\mathbf{g}_4$, $\mathbf{g}_5$ and $\mathbf{g}_5+\mathbf{g}_6$. It is not hard to see that each vector in $\mathbf{G}$ can be reconstructed by adding at most 2 vectors in $\mathbf{A}_2$. The vectors in $\mathbf{A}_2$ that are not in $\mathbf{G}$ can be aptly represented as intermediate nodes on the previously described bipartite graph, which are shown in Figure 2 as highlighted nodes. Each added node represents a new vector, which is the sum of the vectors for the nodes in its inbound set. We refer to the process of adding these intermediate nodes as creating a *branch*, which is defined next.

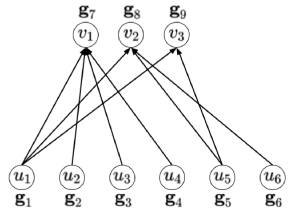

Figure 1: Bipartite graph representation.

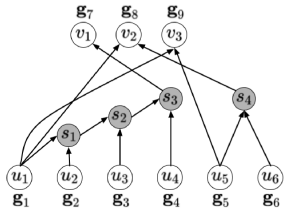

Figure 2: Optimal representation when $k = 2$.

Definition IV.1. Given an ordered set $\mathcal{S} = \{s_1, \cdots, s_S\}$ of nodes, where s_i preceeds s_{i+1} for $i \in [S-1]$, a *branch on* $\mathcal{S}$ is a set $\mathcal{S}' = \{s'_1, \cdots, s'_{S-1}\}$ of $S-1$ intermediate nodes added to the graph with the following connections: node s'_1 has two incoming edges from s_1 and s_2, and for $i \in [S-1] \setminus \{1\}$, s'_i has two incoming edges from nodes s'_{i-1} and s_{i+1}.

For the example in Figure 2, we created branches on two ordered sets, $\mathcal{S}_1 = \{u_1, u_2, u_3, u_4\}$ and $\mathcal{S}_2 = \{u_5, u_6\}$. Once the branch is added, we can change the connections of the nodes in $\mathcal{V}$ in accordance to the added vectors. For the example in Figure 2, we can replace $u_{[4]}$ in $\mathcal{I}_{v_1}$ with only s_3.

Using this representation, we have the following lemma.

Lemma IV.1. *If $\mathcal{O}_{u_{i_T}} \subseteq \mathcal{O}_{u_{i_{T-1}}} \subseteq \cdots \subseteq \mathcal{O}_{u_{i_1}}$ for some permutation $i_1, \cdots, i_T$ of $[T]$, then this instance can be solved by exactly T transmissions for any $k \geq 2$.*

Proof: One solution of such instance would involve creating

a branch on the set $\mathcal{S} = \{u_{i_T}, u_{i_{T-1}}, \cdots, u_{i_1}\}$. The scheme used would have the matrix $\mathbf{A}_2$ with its t-th row $\mathbf{a}_t = \sum_{\ell=1}^{t} \mathbf{g}_{i_\ell}$ for $t \in [T]$. Note that $\mathbf{g}_{i_1} = \mathbf{a}_1$ and $\mathbf{a}_t + \mathbf{a}_{t-1} = \mathbf{g}_{i_t}$ for all $t \in [T] \setminus \{1\}$. Moreover, for $j \in [n] \setminus [T]$, if $v_{j-T} \in \mathcal{O}_{u_{i_t}}$ for some i_t, then $v_{j-T} \in \mathcal{O}_{u_{i_\ell}}$ for all $\ell \leq t$. If we let t be the maximum index for which $v_{j-T} \in \mathcal{O}_{u_{i_t}}$, then we have $\mathcal{I}_{v_{j-T}} = \{u_{i_1}, \cdots, u_{i_t}\}$, and so we get $\mathbf{g}_j = \sum_{\ell=1}^{t} \mathbf{g}_{i_\ell} = \mathbf{a}_t$. ■

Corollary IV.2. *For $\mathbf{G} \in \mathbb{F}_2^{n \times T}$ of rank T, if $n = T+1$, then this instance can be solved in T transmissions for any $k \geq 2$.*

Proof: Without loss of generality, let $\mathbf{g}_{[T]}$ be a set of linearly independent vectors of $\mathbf{G}$. Then, we have $\mathcal{O}_{u_i} = \{v_1\}$ for $i \in \mathcal{I}_{v_1}$ and $\mathcal{O}_{u_j} = \emptyset$ for $j \in [T] \setminus \mathcal{I}_{v_1}$. Thus, from Lemma IV.1, this instance can be solved in T transmissions. ■

V. Algorithms for General Instances

A. Successive Circuit Removing (SCR) algorithm

Our first proposed algorithm is based on Corollary IV.2, which can be interpreted as follows: any matrix $\mathbf{G}$ of $r+1$ vectors and rank r can be reconstructed by a corresponding $\mathbf{A}_2$ matrix with r rows. We denote this collection of vectors as a *circuit*[2]. Our algorithm works for the case $k = 2^q$, for some integer q. We first describe SCR for the case where $q = 1$. The algorithm works as follows:

1) *Circuit Finding:* find a set of vectors of $\mathbf{G}$ that form a circuit of small size. Denote the size of this circuit as $r+1$.
2) *Matrix Update:* apply Corollary IV.2 to find a set of r vectors that can optimally reconstruct the circuit by adding at most $k = 2$ of them, and add this set to $\mathbf{A}_2$.
3) *Circuit Removing:* update $\mathbf{G}$ by removing the circuit. Repeat the first two steps until the matrix $\mathbf{G}$ is of size $T' \times T$ and of rank T', where $T' \leq T$. Then, add these vectors to $\mathbf{A}_2$.

Once SCR is executed, the output is a matrix $\mathbf{A}_2$ such that any vector in $\mathbf{G}$ can be reconstructed by adding at most $k = 2$ vectors of $\mathbf{A}_2$. Consider now the case where $q = 2$ (i.e., $k = 4$) for example. In this case, a second application of SCR on the matrix $\mathbf{A}_2$ would yield another matrix, denoted as $\mathbf{A}_4$, such that any row in $\mathbf{A}_2$ can be reconstructed by adding at most 2 vectors of $\mathbf{A}_4$. Therefore, any vector in $\mathbf{G}$ can now be reconstructed by adding at most 4 vectors of $\mathbf{A}_4$. We can therefore extrapolate this idea for a general q by successively applying SCR q times on $\mathbf{G}$ to obtain $\mathbf{A}_k$, with $k = 2^q$.

The following theorem gives a closed form characterization of the best and worst case performance of SCR.

Theorem V.1. *Let T_q^{SRC} be the number of vectors in $\mathbf{A}_k$ obtained via SCR. Then, for $k = 2^q$ and integer q, we have*

$$\underbrace{f^{Best}(f^{Best}(\cdots f^{Best}(n)))}_{q \text{ times}} \leq T_q^{SRC} \leq \underbrace{f^{Worst}(f^{Worst}(\cdots f^{Worst}(n)))}_{q \text{ times}}, \tag{3}$$

where $f^{Best}(n) = 2\left\lfloor \frac{n}{3} \right\rfloor$ and $f^{Worst}(n) = T\left(\left\lfloor \frac{n}{T+1} \right\rfloor + 1\right)$.

[2]This is in accordance to the definition of a circuit for a matroid [5].

Proof: First we focus on the case $q = 1$. The lower bound in (3) corresponds to the best case when the matrix $\mathbf{G}$ can be partitioned into disjoint circuits of size 3. In this case, if SRC finds one such circuit in each iteration, then each circuit is replaced with 2 vectors in $\mathbf{A}_2$ according to Corollary IV.2. To obtain the upper bound, note that any collection of $T+1$ has at most T independent vectors, and therefore contains a circuit of at most size $T+1$. Therefore, the upper bound corresponds to the case where the matrix $\mathbf{G}$ can be partitioned into circuits of size $T+1$ and an extra T linearly independent vectors. In that case, the algorithm can go through each of these circuits, adding T vectors to $\mathbf{A}_2$ for each of these circuits, and then add the last T vectors in the last step of the algorithm. Finally, the bounds in (3) for a general q can be proven by a successive repetition of the above arguments. ■

B. Branch-Search heuristic

A naive approach to determining the optimal matrix $\mathbf{A}_k$ is to consider the whole space $\mathbb{F}_2^T$, loop over all possible subsets of vectors of $\mathbb{F}_2^T$ and, for every subset, check if it can be used as a matrix $\mathbf{A}_k$. The minimum-size subset which can be used as $\mathbf{A}_k$ is indeed the optimal matrix. However, such algorithm requires in the worst case $O\left(2^{2^T}\right)$ number of operations, which makes it prohibitively slow even for very small values of T. Instead, the heuristic that we here propose finds a matrix $\mathbf{A}_k$ more efficiently than the naive search scheme. The main idea behind the heuristic is based on providing a subset $\mathcal{R} \subset \mathbb{F}_2^T$ which is much smaller than 2^T and is guaranteed to have at least one solution. The heuristic then searches for a matrix $\mathbf{A}_k$ by looping over all possible subsets of $\mathcal{R}$. Our heuristic therefore consists of two sub-algorithms, namely Branch and Search. Branch takes as input $\mathbf{G}$, and produces as output a set of vectors $\mathcal{R}$ which contains at least one solution $\mathbf{A}_k$. The algorithm works as follows:

1) Find a set of T vectors of $\mathbf{G}$ that are linearly independent. Denote this set as $\mathcal{B}$.
2) Create a bipartite graph representation of $\mathbf{G}$ as discussed in Section IV, using $\mathcal{B}$ as the independent vectors for $\mathcal{U}$.
3) Pick the dependent node v_i with the highest degree, and split ties arbitrarily. Denote by $\deg(v_i)$ the degree of node v_i.
4) Consider the inbound set $\mathcal{I}_{v_i}$, and sort its elements in a descending order according to their degrees. Without loss of generality, assume that this set of ordered independent nodes is $\mathcal{I}_{v_i} = \{u_1, u_2, \cdots, u_{\deg(v_i)}\}$.
5) Create a branch on $\mathcal{I}_{v_i}$. Denote the new branch nodes as $\{u_1^\star, u_2^\star, \cdots, u_{\deg(v_i)}^\star\}$.
6) Update the connections of all dependent nodes in accordance with the constructed branch. This is done as follows: for each node $v_j \in \mathcal{V}$ with $\deg(v_j) \geq k$, if $\mathcal{I}_{v_j} \cap \mathcal{I}_{v_i}$ is of the form $\{u_1, u_2, \cdots, u_\ell\}$ for some $\ell \leq \deg(v_i)$, then replace $\{u_1, u_2, \cdots, u_\ell\}$ in $\mathcal{I}_{v_j}$ with the single node $u_\ell^\star$.
7) Repeat 3) to 6) until all nodes in $\mathcal{V}$ have degree at most k.

The output $\mathcal{R}$ is the set of vectors corresponding to all nodes in the graph. The next theorem shows that $\mathcal{R}$ in fact contains one possible $\mathbf{A}_k$, and characterizes the performance of Branch.

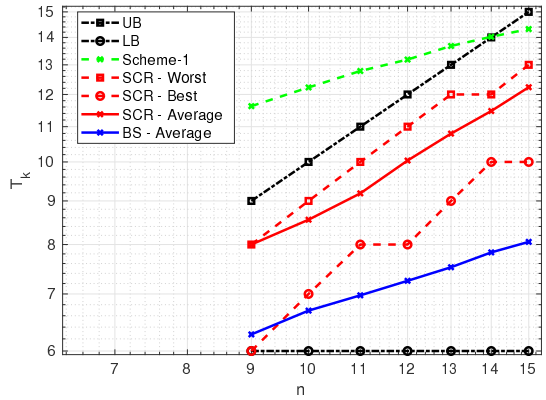

Figure 3: Numerical evaluation - $T = 6$, $k = 2$.

Theorem V.2. *(Proof in [4, Appendix 2]) For a matrix* $\mathbf{G}$ *of dimension* $n \times T$*, (a) Branch produces a set* $\mathcal{R}$ *which contains at least one possible* $\mathbf{A}_k$*, (b) the worst-case time complexity* t_{Branch} *of Branch is* $O(n^2)$*, and (c)* $|\mathcal{R}| \leq (n-T)T$*.*

Let t_{Search} be the worst-time complexity of the Search step in Branch-Search. Then the worst-case time complexity of Branch-Search is equal to $t_{\text{BS}} = t_{\text{Branch}} + t_{\text{Search}} \leq O(n^2) + 2^{|\mathcal{R}|} = O(n^2) + O(2^{nT}) = O(2^{nT})$, which is exponentially better than the complexity of the naive search. Although our heuristic is still of exponential runtime complexity, we observe from numerical simulations that $|\mathcal{R}|$ is usually much less than $(n-T)T$. Finding more efficient ways of searching through the set $\mathcal{R}$ to find a better solution $\mathbf{A}_k$ is an open question.

VI. Numerical Evaluation

Here we evaluate the performance of our proposed schemes through numerical evaluations. Specifically, we assess the performance in terms of T_k of the scheme in Theorem III.1 (which we here refer to as Scheme-1), SCR and Branch-Search (labeled BS). We compare their performance against the lower bound in Lemma II.1 (denoted by LB), and the upper bound of sending uncoded transmissions (denoted by UB). For the case of partial-space covering, we adapt Scheme-1 in the following way: we first sort the columns of $\mathbf{G}$ in a decreasing order according to their weights (i.e., number of non-zero elements), then for the i-th part of length $\lceil T/k \rceil$, we fill $\mathbf{B}_i$, not with all non-zero vectors of length $\lceil T/k \rceil$ (as described in the proof of Theorem III.1), but only with all the vectors that appear for that part across all the n vectors of $\mathbf{G}$. This modification removes vectors from the matrix $\mathbf{A}_k$ that are not used by any vector in $\mathbf{G}$. For SCR, we evaluate its average performance as well as its upper and lower bound performance established in Theorem V.1. For Branch-Search, we evaluate its average performance. Figure 3 shows the performance of all the aforementioned schemes for $T = 6$ and $k = 2$. As can be seen, Scheme-1 does not perform well for small values of n. SCR consistently performs better than uncoded transmissions. In addition, although the current implementation of SCR greedy searches for a small circuit to remove, more sophisticated algorithms for small circuit finding could potentially improve its performance. However, the bounds in (3) suggest that the performance of SCR is asymptotically $O(n)$. Branch-Search appears to perform better than other schemes in the average sense. Our current investigation includes understanding its asymptotic behavior in the worst-case.

VII. Related Work

The problem of protecting privacy was initially proposed to enable the disclosure of databases for public access, while maintaining the anonymity of the users [6]. In *Private Information Retrieval* (PIR) [7], [8], clients ensure that no information about their requests is revealed to a set of malicious databases when they retrieve information from them. Similarly, the problem of *Oblivious Transfer* (OT) [9] establishes, by means of cryptographic techniques, two-way private connections between the clients and the server.

We were here interested in addressing privacy concerns within the framework of index coding. This problem differs from secure index coding [10], where the goal is to guarantee that each client does not learn any information about the *content* of the messages other than its request. Differently, our goal was to limit the information that a client can learn about the *identities* of the requests of other clients. Our initial work in [3] addressed the possibility of designing coding matrices that provide privacy guarantees for clients. The solutions based on k-limited-access schemes proposed in [1] can be interpreted as finding overcomplete bases that allow *sparse* representation of vectors, which is closely related to dictionary learning [11]. However, finding lossless representation of vectors forbids us from using the efficient dictionary learning algorithms.

References

[1] M. Karmoose, L. Song, M. Cardone, and C. Fragouli, "Preserving privacy while broadcasting: K-limited-access schemes," in *IEEE Information Theory Workshop (ITW)*, November 2017, pp. 514–518.

[2] Z. Bar-Yossef, Y. Birk, T. Jayram, and T. Kol, "Index coding with side information," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1479–1494, February 2011.

[3] M. Karmoose, L. Song, M. Cardone, and C. Fragouli, "Private broadcasting: an index coding approach," in *IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2548–2552.

[4] ——, "Privacy in index coding: Improved bounds and coding schemes," *arXiv preprint arXiv:1801.03892*, 2018.

[5] J. G. Oxley, *Matroid theory*. Oxford University Press, USA, 2006, vol. 3.

[6] C. C. Aggarwal and S. Y. Philip, "A general survey of privacy-preserving data mining models and algorithms," in *Privacy-preserving data mining*. Springer, 2008, pp. 11–52.

[7] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, November 1998.

[8] 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, March 2018.

[9] M. Mishra, B. K. Dey, V. M. Prabhakaran, and S. Diggavi, "The oblivious transfer capacity of the wiretapped binary erasure channel," in *IEEE International Symposium on Information Theory*, June 2014, pp. 1539–1543.

[10] S. H. Dau, V. Skachek, and Y. M. Chee, "On the security of index coding with side information," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3975–3988, June 2012.

[11] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, June 2010.