Embedded Index Coding

Alexandra Porter and Mary Wootters, *Member, IEEE*

***Abstract*— Motivated by applications in distributed storage and distributed computation, we introduce *embedded index coding* (EIC). EIC is a type of distributed index coding in which nodes in a distributed system act as both broadcast senders and receivers of information. We show how linear embedded index coding is related to linear index coding in general, and give characterizations and bounds on the communication costs of optimal embedded index codes. We also define *task-based* EIC, in which there is only one sender node responsible for transmitting a block to a particular receiving node. Task-based EIC is more computationally tractable and has advantages in applications such as distributed storage, in which senders may complete their broadcasts at different times. Finally, we give heuristic algorithms for approximating optimal linear embedded index codes, and demonstrate empirically that these algorithms perform well.**

***Index Terms*— Index coding, distributed storage, coded computation.**

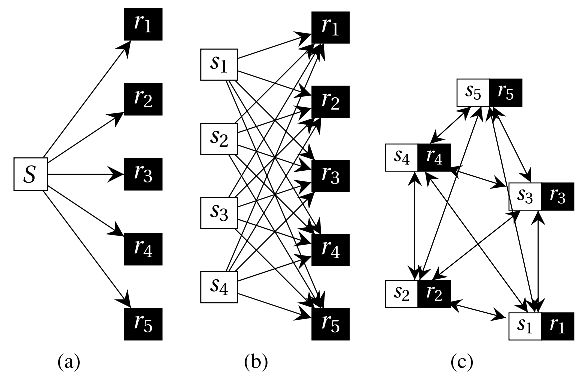


Fig. 1. Communication model for (a) centralized index coding with sender S, receivers $r_1, ..., r_5$; (b) general multi-sender index coding with senders $s_1, .., s_4$ and receivers $r_1, ..., r_5$; and (c) embedded index coding, a special case of (b) with joint sender and receiver nodes $r_1 = s_1, ..., r_5 = s_5$.

I. Introduction

IN *INDEX coding*, defined by Bar-Yossef, Birk, Jayram and Kol in [3], sender(s) encode data blocks into messages which are broadcast to receivers. The receivers already have some of the data blocks, and the goal is to minimize the number of messages broadcast by using this "side information." For example, if node r_1 knows a data block b_1 and node r_2 knows block b_2, a sender S can broadcast $b_1 \oplus b_2$. Then r_1 can cancel out b_1 and r_2 can cancel b_2 such that both nodes learn a distinct new block from a single broadcast message.

Index coding is typically studied in the models depicted in Figures 1a and 1b, where the senders are distinct from the receivers. In this paper, we consider a setting—depicted in Figure 1c—where the senders *are* the receivers. This is similar to a "peer-to-peer" network model, but in this setting nodes are always communicating by broadcasting to the full network, rather than communicating with each other directly. This model is motivated by applications in device-to-device multicast and distributed computation, as we discuss in Section I-B.

We call this model *embedding index coding* (EIC). In this paper, we study the EIC model, and establish characterizations of optimal EIC solutions as well as separations between EIC and other models. Moreover, we develop efficient heuristics for finding good EIC solutions, and demonstrate empirically that they perform well. We briefly summarize our contributions next in Section I-A, and then discuss some motivations for the EIC model more in Section I-B.

A. Contributions

Our contributions can be summarized as follows.

1) We define *embedded index coding* (EIC). As elaborated on in Section I-B below, EIC is motivated by applications in device-to-device multicast and in distributed computation. In this paper, we argue that it is worth studying EIC, both because of the many natural applications, and because—as we will demonstrate—focusing on EIC as a special case of the more general multi-sender model (that is, Figure 1c as a special case of Figure 1b) will allow us to obtain stronger results and faster algorithms.
2) We define the notion of a *task-based* solution to an EIC problem. In task-based solutions, the communication can be partitioned into independent tasks, so that each receiver is only reliant on a single sender to get a

Manuscript received September 24, 2019; revised July 20, 2020; accepted November 17, 2020. Date of publication December 10, 2020; date of current version February 17, 2021. This work was supported in part by the NSF under Grant CCF-1657049 and in part by the NSF CAREER under Grant CCF-1844628. The work of Alexandra Porter was supported in part by the National Science Foundation Graduate Research Fellowship under Grant DGE-1656518. This article was presented at the IEEE Information Theory Workshop (ITW) 2019. *(Corresponding author: Alexandra Porter.)*

Alexandra Porter is with the Department of Computer Science, Stanford University, Stanford, CA 94305 USA (e-mail: amporter@cs.stanford.edu).

Mary Wootters is with the Department of Computer Science, Stanford University, Stanford, CA 94305 USA, and also with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305 USA.

Communicated by B. Dey, Associate Editor for Coding Techniques.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIT.2020.3043767.

Digital Object Identifier 10.1109/TIT.2020.3043767

particular block.[1] Task-based solutions can be more robust to failures or delays: if a sender's messages are corrupted or lost, the messages from other senders can still be used to fully decode data blocks. Moreover, as noted below, there are efficient heuristics to find good task-based solutions to EIC problems.

3) We prove several results establishing relationships between centralized (single-sender) index coding, EIC, and task-based EIC, for linear schemes over $\mathbb{F}_2$. In particular, we show that the optimal communication for a general EIC problem is only a factor of two worse than the optimal communication in the centralized model; we give characterizations and bounds for the optimal communication cost of the best task-based solutions to an EIC problem; and we show separations between the three models.

4) Based on the (proofs of) the bounds described above, we design heuristics for designing general EICs and task-based EICs. We give empirical evidence that these approximation algorithms perform well, compared to existing algorithms for the (more general) multi-sender index coding problem (Figure 1b).

B. Motivation

In this section we give some motivation for the EIC model, which we argue may have many applications.

One natural application of EIC is file sharing, where several files are stored on several devices, and some devices would like files that they do not currently have. In this case, EIC might be relevant in a device-to-device multicast model, where network nodes communicate information among themselves via multicast, so each node is able to send the same message to multiple receivers simultaneously. A device-to-device multicast model has been studied, by e.g., [34], in which cellular phones share information with each other to reduce load on the base stations they would otherwise access for all data. In this setting, EIC could be used to reduce the communication cost of file sharing or similar tasks.

A second, less straightforward, application of EIC is *coded computation,* in particular the work of [24], [25]. Those works use coding theoretic techniques to improve the computation and communication costs of distributed computation in a MapReduce model. We describe the set-up in detail below; as we will see, there a "shuffle" phase which is precisely an instance of an EIC problem.

We first set up notation for the MapReduce model [9]. A distributed computation task structured using the MapReduce model consists of two computation phases: *map* and *reduce*, with a communication phase, *shuffle*, happening in between. Following the notation of [24], suppose we want to compute Q output functions, each of which are functions of N input files, using K servers, for some $Q, N, K \in \mathbb{N}$. Let $w_1, ..., w_N$ denote the input files, and let $\phi_1, ..., \phi_Q$ denote the output functions so at the end of the day we would like $\phi_q(w_1, \ldots, w_N)$ for $q = 1, \ldots, Q$. We also suppose that each function ϕ_q for $q = 1, ..., Q$ can be decomposed as $\phi_q(w_1, ..., w_N) = h_q(g_{q,1}(w_1), ..., g_{q,N}(w_N))$ for some other functions h_q and $g_{q,1}, ..., g_{q,N}$. The functions $g_{q,n}$ for $q = 1, ..., Q$ and $n = 1, ..., N$ are the *map* functions and the functions h_q for all $q = 1, ..., Q$ are the *reduce* functions. Each node x initially receives some subset W_x of the files $W := \{w_1, ..., w_N\}$. We note that the sets W_x may overlap, so the same file can be given to different nodes. Then node x computes the set of intermediate values $v_{q,n} := g_{q,n}(w_n)$ for all $q = 1, ..., Q$ and $w_n \in W_x$. Next, each node x is tasked with computing a set of reduce functions $Q_x \subset \{h_1, \ldots, h_Q\}$. For node x to compute a reduce function h_q, it needs $v_{q,1}, ..., v_{q,N}$. From the map phase, each node x only has $\{v_{q,n} : w_n \in W_x\}$. Thus the *shuffle* step, in which nodes communicate intermediate values with each other, is needed. Figure 2 shows an example of a computation task using the MapReduce.

The task faced by the nodes during the shuffle phase is precisely an instance of EIC. Each node has some side information—the intermediate values $v_{q,n}$ that it computed in the map phase—and each node wants some information—the intermediate values $v_{q,n}$ that it must compute in the reduce phase. The nodes can multicast information to each other and take advantage of side information in order to reduce communication. We note that this problem is only interesting if some intermediate values are held by multiple nodes; otherwise there is no opportunity for index coding. Thus, interesting instances of the EIC problem only arise when the files w_i are distributed with redundancy, which is precisely the case in the setting of coded computation [24], [25].

The works [24], [25] define the redundancy between the sets W_x in such a way that the they can analytically identify a good EIC solution to the specific EIC problem they (implicitly) define. Even though in those works one gets to design the EIC instance, there are two reasons in the context of coded computation that we might want to study a general EIC problem that we do not get to design.

The first reason is that it might be the case the redundancy pattern changes in an unforeseen way. For example, suppose that the map phase was disrupted so that the scheme of [24] was no longer possible, e.g. if a node failed (we note that the work [24] does not handle stragglers or failed nodes). This would give rise to a new EIC problem, which could be solved to determine a communication-efficient way to shuffle.

The second reason is that in some settings the redundancy pattern may not have been designed not to optimize the shuffle phase, but for some other reason. One example of this is the situation in [25], which handles straggling (that is, slow or unresponsive) nodes for the application of matrix multiplication. The scheme in [25] achieves a latency-load tradeoff which does not match the lower bound they give, and thus may not be optimal; viewing this as an EIC problem and solving it may yield shuffle schemes with less communication. In more detail, in the scheme of [25], only the fastest subset of q servers to complete the map operation move on to shuffle and reduce, thereby mitigating the effects of straggling servers.

[1] This can be seen as a generalization of *Instantly Decodable Network Codes* [19] which have been studied with similar motivation (see Remark 3). Task-based solutions are also related to *Locally Decodable Index Codes* [30] (see Remark 4).

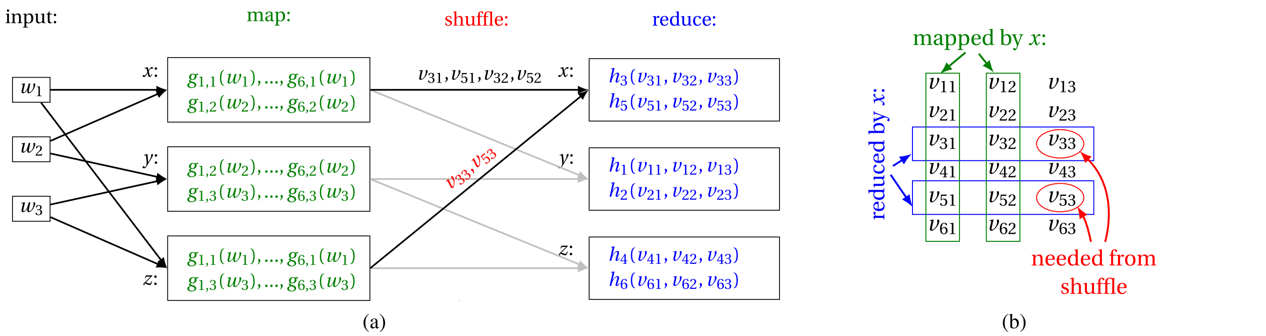


Fig. 2. Example of MapReduce computation with $N = 3$ and $Q = 6$: the system as a whole computes $\phi_1(w_1, w_2, w_3), \phi_2(w_1, w_2, w_3), ..., \phi_6(w_1, w_2, w_3)$. In (a), the boxes show a possible assignment of map and reduce functions. In particular, for node x: $W_x = \{1, 2\}$ meaning that x receives data blocks w_1 and w_2. Then x computes all $v_{11}, ..., v_{61}$ and $v_{12}, ..., v_{62}$ (green). In this example, $Q_x = \{3, 5\}$ meaning that x needs v_{31}, v_{32}, v_{33} and v_{51}, v_{52}, v_{53} in order to compute ϕ_3 and ϕ_5. Since x did not compute v_{33} and v_{53} it must acquire them via shuffling. Note that node x has $v_{31}, v_{51}, v_{32}, v_{52}$ from mapping so it reuses them locally to compute h_3 and h_5, as indicated by the arrow connecting the x's. In (b), the values computed in map and needed for reduce are visualized as a grid, with boxes indicating what x computes and then needs.

In this scheme, an MDS code is applied over the input values, such that each of the q servers doing the shuffle and reduce can collect *any* m intermediate values for each of its assigned reduce functions, and thus can get the required data regardless of which q servers completed the map phase. This differs from index coding, in which we have a fixed set of data that each node requires. However, finding the best shuffle scheme can be solved as a set of index coding problems, in which each corresponds to a particular outcome of which m values each node gets. Solving for the optimal solution to each and minimizing over all combinations would give an optimal shuffle solution. Thus EIC could be useful in closing the gap between the upper and lower bounds on communication established by [25].[2]

C. Outline

The rest of this paper is organized as follows. In Section II we review related work in more detail. In Section III we formally define the EIC problem and several notions of solution. In Section IV we show how EIC problems relate to more general index coding and we analyze how different notions of solutions are related. In Section V we provide algorithms for approximating optimal EIC solutions and demonstrate empirically that they perform well.

II. Related Work

In this section we briefly review related work. Index coding was first introduced by [3], based on the Informed-Source Coding on Demand (ISCOD) model proposed by [4], and many extensions and variations have been studied, including non-linear index coding [26] and multi-sender index coding [32]. We focus on *linear* index coding, where the messages broadcast are linear combinations of the original data.

The work of [3] characterized the number of broadcasts required to solve an index coding problem in terms of the *minrank* (c.f. Definition 6) of a relevant graph. The minrank is difficult to compute exactly, and a number of approximations and heuristics have been studied for computing optimal linear index codes [6], [7], [31], [36], [39], [40]. We will also use the minrank, and heuristics for computing it, in our approach.

Embedded index codes are a special case of the linear multi-sender index codes in [18] and [22], which both consist of multiple senders and multiple receivers, but as two distinct and non-overlapping sets of nodes; this is the setting depicted in Figure 1b. In [22], rank minimization is used in an approach similar to our method. The approaches of [18], [22] can also be applied to EIC, and we compare these approaches in more detail in Section V.

The embedded model in Figure 1c has been studied before in [12]. In that work, the authors study a special case of EIC, where each node wants all of the data blocks it does not already have. In this setting, they develop a greedy algorithm which uses a near-optimal number of broadcasts. However, their approach crucially uses the fact that every node wants every block, and does not seem to generalize to the general EIC setting that we study here.

While our coding scheme is deterministic, our multi-sender network model is similar to those studied with *composite coding,* an approach based on randomized coding [1]. Multi-sender models and achievable rate regions using composite coding are defined in [20], [21], [33]; to the best of our knowledge these results are not directly applicable to our scheme.

Index coding is a special instance of the network coding problem (e.g., [23]), in which source nodes send information over a network containing intermediate nodes, which may modify messages, in addition to receiver nodes. It has also been shown that network coding instances can be reduced to index coding instances [10], [11]. Real-Time Instantly Decodable Network Codes (IDNC's) [19] aim to minimize completion delay of the communication task, rather than the index coding goal of minimizing total number of messages. Our task-based solutions are a generalization of instant decodability in index codes (see Remark 3).

[2]We note that *pliable index coding* is a variant of index coding that allows for this flexibility, and has been used to improve shuffle [38] in a master-worker setting, but not the fully decentralized system we consider.

Task-based solutions are also related to the notion of *locally decodable index codes*. An index coding solution has *locality* r if each node uses at most r received symbols to decode any message symbol. There is tradeoff between optimal broadcast rate and locality of solutions for a given index coding problem [30]. When $r = 1$, locally decodable index codes are a special case of task-based schemes, although the notions diverge for more general r (see Remark 4).

Our construction is motivated by the problem of data shuffling for coded computation, such as in [24], [25]; this is described in Section I-B. Other connections between index coding and distributed storage have been established, but are not directly related to our work. These include the relationship between an optimal recoverable distributed storage code and a general optimal index code [29] and the duality of linear index codes and Generalized Locally Repairable codes [2], [35].

Finally, index coding techniques can also be applied to coded caching (e.g., [27], [13] and references therein), in which nodes may request and store data dynamically. Coded multicasting similar to index coding has been applied to decentralized coded caching [28], and our work could also be applied in coded caching.

a) Subsequent work: In our work, we introduce the notion of task-based schemes for EIC, and develop heuristics for these schemes. However, we left it as an open problem to understand the limitations of task-based schemes relative to other schemes. Since our work first appeared, Haviv has solved this problem by giving tight bounds on the gap between task-based schemes and centralized schemes for EIC [17]. Briefly, this work shows that for any graph G, the length of the best task-based scheme is at most quadradically worse than the best scheme without the task-based restriction, and also shows that there exist graphs where this gap is asymptotically tight.

III. Framework

In this section we formally describe our model for Embedded Index Coding.

We assume that there is a set of m data blocks, $\mathcal{D} \in (\mathbb{F}_2^\ell)^m$, where each data block is an element of $\mathbb{F}_2^\ell$; when convenient, we will view $\mathcal{D} \in \mathbb{F}_2^{m\times\ell}$ as an $m \times \ell$ boolean matrix with the m data blocks as rows. These m data blocks are stored on n storage nodes; each node i stores a subset of the data blocks, and some data blocks may be stored on multiple nodes. We assume that each node can perform local computations and can broadcast information over an error-free channel to all the other nodes. In this work, we focus on a *linear* model, where each node is restricted to computing $\mathbb{F}_2$-linear combinations of data blocks.

We note that all of our results generalize to arbitrary finite fields. However, for simplicity we will state all of our results over $\mathbb{F}_2$, the finite field of size two.

An Embedded Index Coding (EIC) problem is defined in terms of which data blocks each node *has* and *needs*. It will be convenient to represent these "has" and "needs" relationships in terms of binary matrices B and R respectively.

Definition 1: An *Embedded Index Coding (EIC) problem* is specified by a pair of matrices $R, B \in \mathbb{F}_2^{n\times m}$ s.t. *supp*$(R) \cap$ *supp*$(B) = \emptyset$.

Informally, the interpretation should be that in an EIC problem (R, B), a node u *needs* block a if $R_{ua} = 1$ and *has* block b if $B_{ub} = 1$.

Remark 1: The matrices (R, B) can be remembered as "R" for what nodes *request* and "B" for what nodes had *before*.

Each node u will broadcast a set of $b_u \in \mathbb{N}$ linear combinations of the blocks it has, and the goal is for each node to be able to recover all of the blocks that it needs. We formalize this in the following definition.

Definition 2: For an EIC problem (R, B) a *linear broadcast solution* that solves (R, B) is a collection of matrices $\beta^{(1)}, ..., \beta^{(n)}$ and integers $h_1, ..., h_n$ with $\beta^{(u)} \in \mathbb{F}_2^{h_u\times m}$ for each $u \in [n]$ so that:

- For each $u \in [n]$ and each $a \in [m]$ so that $B_{ua} = 0$, the a^{th} column of $\beta^{(u)}$ is zero.
- For each $u \in [n]$ and each $a \in [m]$ so that $R_{ua} = 1$, there is some vector $\boldsymbol{\alpha}^{(u,a)} \in \mathbb{F}_2^{\sum_\ell h_\ell+m}$ so that

$$\boldsymbol{e}_a = \boldsymbol{\alpha}^{(u,a)} \cdot \left[\begin{array}{c} \beta^{(1)} \\ \hline \beta^{(2)} \\ \hline \vdots \\ \hline \beta^{(n)} \\ \hline \mathrm{diag}(B_u) \end{array}\right],$$

  where B_u is the row of B indexed by u and $\mathrm{diag}(B_u)$ is the matrix with B_u on the diagonal. Above, $\boldsymbol{e}_j$ denotes the j^{th} standard basis vector.
- The *length* of an EIC solution is $\Sigma_a h_a$, the number of symbols broadcast. We also refer to this as the *communication cost* of the solution.

An Embedded Index Coding (EIC) problem (R, B) is *solvable* if a linear broadcast solution as defined above exists for (R, B). Note that as long as there is a nonzero entry in each column of B, meaning there is at least one node that has each block, a problem is solvable.

To use a linear broadcast solution, each node u computes and broadcasts $\beta^{(u)} \cdot \mathcal{D}$, where we view $\mathcal{D} \in \mathbb{F}_2^{m\times\ell}$ as a matrix whose rows are the data blocks. This can be computed locally because the only non-zero columns of $\beta^{(u)}$ correspond to non-zero entries of row B_u, i.e. blocks node u has.

In order to decode the blocks it wants, each node u uses the fact that

$$\text{block } a = \boldsymbol{e}_a \cdot \mathcal{D} = \boldsymbol{\alpha}^{(u,a)} \cdot \left[\begin{array}{c} \beta^{(1)} \\ \hline \beta^{(2)} \\ \hline \vdots \\ \hline \beta^{(n)} \\ \hline \mathrm{diag}(B_u) \end{array}\right] \cdot \mathcal{D},$$

and thus block a is a linear combination (given by $\boldsymbol{\alpha}^{(u,a)}$) of the broadcasts $\beta^{(1)}\mathcal{D}, \ldots, \beta^{(n)}\mathcal{D}$ that node u receives and the data blocks that u already has.

A. Problem Graph and Problem Matrix

We next define some representations of embedded index coding problems (extending the work of [3]) which will be useful in studying the length of solutions and the construction of algorithms.

We begin by defining a graph G which captures an EIC problem. The vertices of G will correspond to *requirement pairs* of the EIC problem, defined as follows.

Definition 3: Given an EIC problem (R, B), the set of *requirement pairs* for (R, B) is $P = \{(u, a) \in [n] \times [m] : R_{ua} = 1\}$.

Now we can formally define the problem graph G for an EIC problem (R, B).

Definition 4: Given an EIC problem (R, B), the *problem graph* $G = (V, E)$ corresponding to (R, B) is the graph G with vertices $V = \{v_{(u,a)} : (u, a) \in P\}$ and (directed) edges $E = \{(v_{(u,a)}, v_{(w,b)}) : B_{ub} = 1 \text{ or } a = b\}$.

That is, for (u, a) and (w, b) in P, there are two reasons that there could be an edge from the vertex $v_{(u,a)}$ to the vertex $v_{(w,b)}$: either the node u has the block b that the node w wants, or else the two blocks a and b are the same block. As we will see, these two types of edges play two different roles.

Figure 3 shows two examples of problem graphs. In Figure 3a, all edges indicate where a node has a block that another is requesting, i.e. cases where $(v_{(u,a)}, v_{(w,b)}) \in E(G)$ because $B_{ub} = 1$. In Figure 3d, dashed edges indicate pairs of vertices which represent two requests for the same block, i.e. cases where $(v_{(u,a)}, v_{(w,b)}), (v_{(w,b)}, v_{(u,a)}) \in E(G)$ because $a = b$.

Definition 5: Given a graph $G = (V, E)$, we say that a matrix $A \in \mathbb{F}_2^{|V| \times |V|}$ *fits* G if:

1) $A_{kk} = 1$ for all $k \in [|V|]$ and
2) for any $k, \ell \in |V|$,$(k, \ell) \notin E$ implies that $A_{k\ell} = 0$.

Thus if M is the adjacency matrix of G and matrix A fits G, the non-zero entries of A (other than the diagonal) are a subset of the non-zero entries of M.

Definition 6: The *minrank* of a graph G in field $\mathbb{F}_2$, denoted $\mathrm{minrk}_2(G)$, is the rank of the lowest-rank matrix A over $\mathbb{F}_2$ which fits G:

$$\mathrm{minrk}_2(G) := \min\{\mathrm{rk}_2(A) : A \text{ fits } G\}$$

In Section IV-A, we will show how our definition of a problem graph generalizes the *side information graph* defined for index coding (that is, the centralized case of Figure 1(a), where each node requests a single unique block). In this setting, it was shown in [3] that $\mathrm{minrk}_2(G)$ is the length of the optimal index code. We will show later how the minrank can also be used in computing solutions for EIC problems.

B. Task-Based Solutions

We now define a *task-based solution,* which is a particular type of solution to an embedded index coding problem. As we will see, we can design efficient heuristics to find task-based solutions, and additionally task-based solutions may be more useful in settings with node failures.

Definition 7: A task $T = (k, M)$ is defined by a sender node k and a set of pairs

$$M \subseteq \{(u, a) \in P \,:\, B_{ka} = 1\}$$

Informally, if $T = (k, M)$ and $(u, a) \in M$, then this means that it is part of the node k's task to send the block a to the node u. Notice that this is not completely general: it rules out the possibility that the node u could recover the block a from two separate sender nodes.

A task-based solution is one built out of tasks. We formally define this as follows.

Definition 8: A *task-based solution* to an EIC problem (R, B) with requirement pairs P is a linear broadcast solution $\beta^{(1)}, ..., \beta^{(n)}$ so that $\beta^{(\ell)} \in \mathbb{F}_2^{h_\ell \times m}$, such that for each $(u, a) \in P$, there is an $\ell \in [n]$ and a coefficient vector $\boldsymbol{\alpha}_\ell^{(u,a)} \in \mathbb{F}_2^{h_\ell + m}$ such that

$$\boldsymbol{e}_a = \boldsymbol{\alpha}_\ell^{(u,a)} \cdot \left[\frac{\beta^{(\ell)}}{\mathrm{diag}(B_u)} \right].$$

Let $T = (\ell, M)$ be a task, for some $M \subseteq \{(v, b) \in P : B_{\ell b} = 1\}$. For any $(v, b) \in M$, we say that the node ℓ is *responsible* for (v, b) in the task T.

Informally, a task-based solution is a linear solution in which each node u decodes each requested block a using only messages from one sender node ℓ who is responsible for (u, a). That is, ℓ broadcasts a vector $\beta^{(\ell)} \cdot \mathcal{D}$, and u should be able to recover a from this vector and its local side information.

A task-based solution to (R, B) is related to the corresponding problem graph $G = (V, E)$ by specifying a partition of the vertices. Let $N^+(v_{(u,a)}) \subseteq V$ denote the out-edge neighborhood of a vertex $v_{(u,a)} \in V$: that is,

$$N^+(v_{(u,a)}) = \left\{ v_{(w,b)} \,:\, (v_{(u,a)}, v_{(w,b)}) \in E \right\}.$$

Definition 9: For an EIC problem (R, B) with problem graph G, define the *sender neighborhood* of node $u \in [n]$ as:

$$N_u = \{v_{(w,b)} \in V : \; B_{ub} = 1\}.$$

That is, the sender neighborhood N_u of a node u is the set of vertices in V corresponding to node-block pairs (w, b) so that the node u has the block b (and thus u could send b to w).

Remark 2: In terms of the problem graph G, we have $N_u \subseteq \cap_a N^+(v_{(u,a)})$. This is because every edge of the "first type" in G leaving $v_{(u,a)}$ goes to a vertex $v_{(w,c)}$ so that $B_{uc} = 1$ (that is, node u has block c), and in particular $v_{(w,c)} \in N_u$. However, it is possible that the containment above is strict. For example, suppose that u only wants one block, a, and furthermore that there is some other node w that also wants a. In this case, there is an edge of the "second type" from (u, a) to (w, a), and hence $(w, a) \in N^+(v_{(u,a)}) = \cap_a N^+(v_{(u,a)})$. However, $(w, a) \notin N_u$, because u does not have the block a. Thus $N_u \subsetneq \cap_a N^+(v_{(u,a)})$.

We note that this containment is an equality (that is, $N_u = \cap_a N^+(v_{(u,a)})$), if $|\{a : v_{(u,a)} \in V\}| > 1$, i.e., node u wants more than one block.

Figure 4 shows examples of sender neighborhoods for the problem graph examples shown in Figure 3a.

Remark 3: Finding tasks (k, M) which maximize $|M|$ while minimizing total broadcast messages is a generalization of Instantly Decodable Network Codes (IDNCs) [19]. More precisely, solving the IDNC problem on sender neighborhood N_k for some node k finds the task (k, M) with maximal $|M|$ such that only one message needs to be broadcast by sender k to satisfy all $(u, a) \in M$.

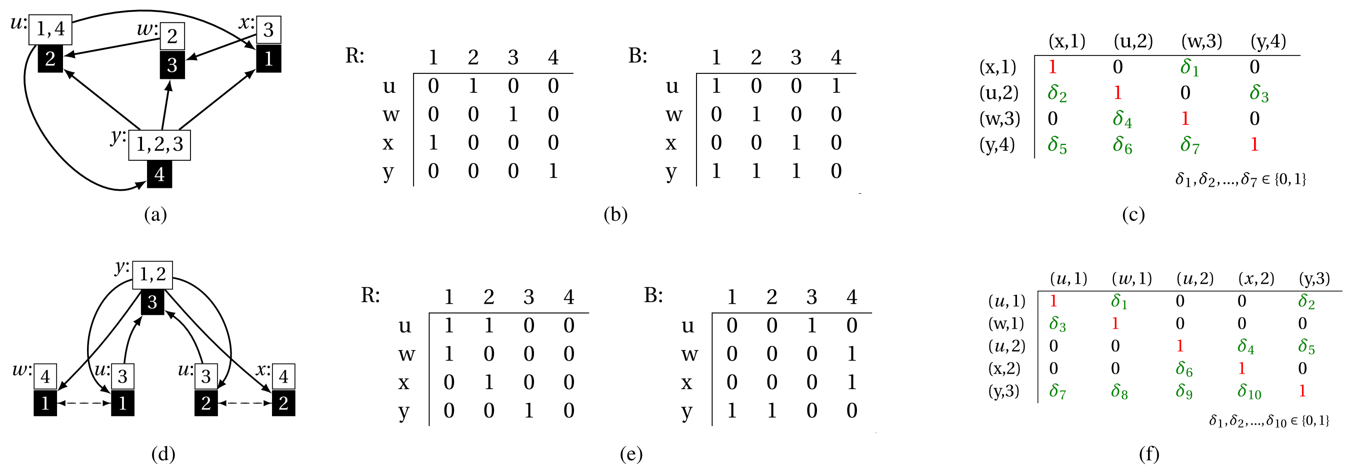


Fig. 3. Examples of a problem graph G for nodes u, w, x, y and data blocks $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4$. Each pair of boxes is a vertex in G, where the black boxes contain indices of requested data blocks and the white boxes contain indices of side information blocks; each pair of boxes is labeled with the relevant node. In (a), node u is requesting block $\mathcal{D}_2$ and has blocks $\mathcal{D}_1, \mathcal{D}_4$ in its side information. Part (b) shows the corresponding (R, B). Part (c) describes all matrices that fit the problem graph in (a). In (d), node u requests blocks 1 and 2, represented by two separate vertices. Since w also requests block 1 and x also requests block 2, we have a different type of edge (dashed) indicating vertices corresponding to the same requested block. Part (e) shows the corresponding (R, B) and (f) describes all matrices that fit the problem graph in (d). Note that all zeros in column 4 of R in (e) mean we do not have a row or column corresponding to $\mathcal{D}_4$ in (f).

Remark 4: Task-based solutions are also related to locally decodable index codes (LDICs) [30]. In an LDIC, a (centralized) index coding solution has locality r if each node uses at most r of the broadcast messages to decode any one block. In the case that $r = 1$, the natural generalization of LDICs to the decentralized setting is a special case of a task-based scheme. When $r > 1$, the two notions are different, but they have a similar flavor of restricting the information that can be used to reconstruct a single block.

Remark 5: Each node k and its sender neighborhood N_k (or any subset of N_k) together form an instance of an index coding problem with a single source: node k is a source which has all blocks requested by nodes in N_k. Thus the communication model is the same as in [3], but it is not necessarily a single unicast problem (see Definition 13); that is, it is not the case that each node wants a unique block.

Definition 10: Let G be a problem graph with sender neighborhoods $N_1, ..., N_n$. A *neighborhood partition* is a set $\{\tilde{N}_1, ..., \tilde{N}_n\}$ such that

1) $\tilde{N}_i \subseteq N_i$ for all $i = 1, ..., n$,
2) $\tilde{N}_i \cap \tilde{N}_j = \emptyset$ for any $i, j \in [n]$,
3) and $\bigcup_{i \in [n]} \tilde{N}_i = V(G)$.

We note that a neighborhood partition exists for a problem graph G of an EIC problem (R, B) as long as (R, B) is solvable. Indeed, in this case $\bigcup_{i \in [n]} N_i = V(G)$, which is all that is required.

Given an EIC problem with problem graph G and task-based solution T, there is a corresponding neighborhood partition $\{\tilde{N}_1, ..., \tilde{N}_n\}$: each vertex $v_{(u,a)}$ in G belongs to the $\tilde{N}_i$ such that $i \in [n]$ is responsible for (u, a) in T. Furthermore, any neighborhood partition trivially corresponds to at least one task-based solution, in which each sender $i \in [n]$ broadcasts each block requested by a node in $\tilde{N}_i$ as a separate message.

For example, there is a task-based solution for the EIC problem shown in Figure 3a, using sender neighborhoods $\tilde{N}_y = \{u, w, x\}$ and $\tilde{N}_u = \{y\}$. The messages for the task executed by node y are $\mathcal{D}_1 \oplus \mathcal{D}_2$ and $\mathcal{D}_2 \oplus \mathcal{D}_3$, and the message broadcast by node u for its task is $\mathcal{D}_4$. Then nodes u, w, and x each decode their requested block from the task executed by node y, and node y decodes its request from the task executed by node u. Task-based solutions like this can be easier to compute than a distributed solution in general, and they allow some independence between nodes: in the example, nodes u, w and x do not need to wait for any node other than node y to be able to decode their requested block.

Remark 6: While we only study task-based solutions on the EIC model, task-based solutions can also be defined for multi-sender index coding in general.

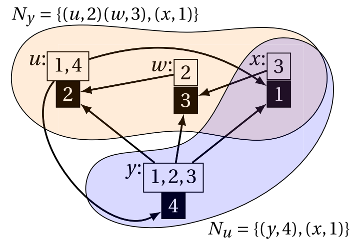
C. Centralized Solutions

We will later compare decentralized solutions to embedded index coding problems to an idealized centralized index coding solution. To that end, we define a solution to a an embedded index coding problem which assumes the existence of some oracle server with access to all of $\mathcal{D}$ (and has no requirements itself).

Definition 11: For an EIC problem defined by (R, B), a *centralized linear broadcast solution* which solves (R, B) is a matrix β and $h \in \mathbb{N}$ with $\beta \in \mathbb{F}_2^{h \times m}$ such that for each $u \in [n]$ and each $a \in [m]$ with $R_{ua} = 1$, there is some vector $\boldsymbol{\alpha}^{(u,a)} \in \mathbb{F}_2^{h+m}$ so that

$$\boldsymbol{e}_a = \boldsymbol{\alpha}^{(u,a)} \cdot \left[\frac{\beta}{\text{diag}(B_u)} \right].$$

Finally, we use the following symbols to denote the optimal lengths for each type of solution:



(a)

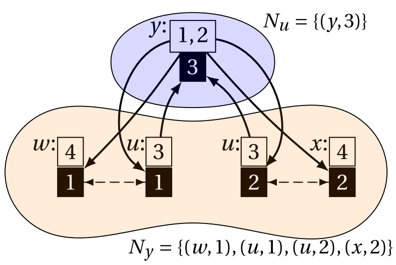
$\beta^{((y,4))}$:

(x,1)	(u,2)	(w,3)	(y,4)
1	1	0	0
0	1	1	0

$\beta^{((u,2))}$:

(x,1)	(u,2)	(w,3)	(y,4)
0	1	1	0

(b)



(c)

$\beta^{((y,3))}$:

(x,1)	(w,1)	(u,2)	(x,2)	(y,3)
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0

$\beta^{((u,1))}$:

(x,1)	(w,1)	(u,2)	(x,2)	(y,3)
0	0	0	0	1

(d)

Fig. 4. Examples from (a) Figure 3a and (c) Figure 3d, shown with sender neighborhoods of nodes u and y. Note that in (c), the out-neighborhood of the vertex $(u,1)$ is $\{(w,1),(y,3)\}$ and the out-neighborhood of $(u,2)$ is $\{(x,2),(y,3)\}$ but the sender neighborhood (Definition 9) of u is the intersection of these. In (b) and (d) the matrices that form a task-based solution for (a) and (c), respectively, are shown.

Definition 12: Let $(C)_{(R,B)}$ denote the minimum length of a centralized linear solution to the EIC problem (R,B) as defined in Definition 11.

Let $(D)_{(R,B)}$ denote the minimum length of a decentralized linear broadcast solution to the EIC problem (R,B) as defined in Definition 2.

Let $(T)_{(R,B)}$ denote the minimum length of a decentralized and task-based solution to the EIC problem (R,B) as defined in Definition 8.

IV. Minimum Code Lengths and Relationships

In this section, we analyze the values of $(C)_{(R,B)}$, $(D)_{(R,B)}$, and $(T)_{(R,B)}$ for a given (R,B). We drop (R,B) from the notation when comparing two of these under the same (R,B) in general. While it has been shown that graph-theoretic upper and lower bounds on minrank can have significant separation [37], they are still useful in comparing the achievable minimum lengths in different solution types for EIC problems.

A. $(C)_{(R,B)}$ and Minrank of the Problem Graph

First, we discuss an idealized centralized solution to an EIC problem, and introduce some useful machinery.

The work [3] defines the *side information graph* for an index coding problem. We show how our problem graph is an effective generalization of the side information graph such that the same technique of using minrank to find an optimal centralized solution applies. The side information graph as defined by [3] is equivalent to a Problem Graph (Definition 4) for any *single unicast* EIC problem (R,B) (defined below).

Definition 13: An EIC specified by (R,B) is a *single unicast* index coding problem if

1) every node requests exactly one data block and
2) each data block is requested by exactly one node.

Figure 3a shows the problem graph for a single unicast EIC; Figure 3d shows the problem graph for an EIC which is not single unicast.

We will generalize the following theorem, which restates Theorem 5 of [3] using our definitions:

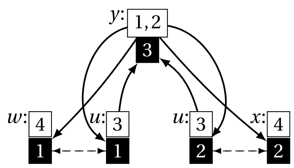
Theorem 1 (Theorem 5 of [3]): Given a single unicast EIC (R,B) and the corresponding problem graph G, $(C)_{(R,B)} = \mathrm{minrk}_2(G)$.

When a problem is not single unicast (in particular when the second condition of Definition 13 does not hold) we constrain the minrank function over a subset of possible matrices, constructed as follows:

Definition 14: Given an EIC problem (R,B) a problem graph $G=(V,E)$, we define the *column repetition function* $\phi_{(R,B)} : \mathbb{F}_2^{|V|\times m} \to \mathbb{F}_2^{|V|\times|V|}$ as follows. Given a matrix $A \in \mathbb{F}_2^{|V|\times m}$, construct a matrix $A' = \phi_{(R,B)}(A) \in \mathbb{F}_2^{|V|\times|V|}$ by copying columns of A as follows. We index the rows of A by pairs $(u,a)\in P$, and the columns of A by data blocks $b\in[m]$. We index the rows and columns of A' by pairs $(u,a)\in P$. Then, for all $(v,b)\in P$, the column of A' indexed by (v,b) is equal to the column of A indexed by b. We will denote the image of $\phi_{(R,B)}$ by $\mathcal{A}_{(R,B)} = \{A' : \exists A \text{ s.t. } \phi_{(R,B)}(A) = A'\}$.

For example, consider the problem graph in Figure 5a (which is the same as Figure 3d, reproduced for the reader's convenience). The matrices described by Figure 5c are those matrices A such that $A' = \phi_{(R,B)}(A)$ fits the problem graph in Figure 5a. Figure 5d describes the matrices A' that fit the graph in Figure 5a and are in $\mathcal{A}_{(R,B)}$. The set of matrices satisfying Figure 5d have the property that the columns indexed by $(u,1)$ and $(w,1)$ are equal, and the columns indexed by $(u,2)$ and $(x,2)$ are equal. Note that the matrices described by Figure 5d are a subset of those described by Figure 3c.

We subsequently assume that for each problem graph $G=(V,E)$, there is a fixed ordering function $f : V \to \{1,\ldots,|V|\}$ used to index matrices using the vertex set.



(a)

R:

	1	2	3	4
u	1	1	0	0
w	1	0	0	0
x	0	1	0	0
y	0	0	1	0

B:

	1	2	3	4
u	0	0	1	0
w	0	0	0	1
x	0	0	0	1
y	1	1	0	0

(b)

	1	2	3
(u,1)	1	0	δ_3
(w,1)	1	0	0
(u,2)	0	1	δ_4
(x,2)	0	1	0
(y,3)	δ_1	δ_2	1

$\delta_1, \delta_2, ..., \delta_4 \in \{0, 1\}$

(c)

	$(u,1)$	$(w,1)$	$(u,2)$	$(x,2)$	(y,3)
$(u,1)$	1	1	0	0	δ_3
(w,1)	1	1	0	0	0
$(u,2)$	0	0	1	1	δ_4
(x,2)	0	0	1	1	0
(y,3)	δ_1	δ_1	δ_2	δ_2	1

$\delta_1, \delta_2, ..., \delta_4 \in \{0, 1\}$

(d)

Fig. 5. Example of a problem graph G for nodes u, w, x, y and data blocks $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4$ ((a) is the same as Figure 3d and (b) is the same as Figure 3e). Part (c) describes all matrices $A \in \mathbb{F}_2^{|V| \times m}$ where $|V| = 5$ and $m = 3$, such that $A' = \phi_{(R,B)}(A)$ fits (a). Note that we omit $\mathcal{D}_4$ and use $m = 3$ instead of $m = 4$ because no node in this example requests $\mathcal{D}_4$. Part (d) describes all matrices A' that fit (a) and have some A such that $\phi_{(R,B)}(A) = A'$. Using the same assignment of $\delta_1, \delta_2, \delta_3, \delta_4 \in \{0, 1\}$ for (c) and (d) would produce a matrix A and the corresponding $A' = \phi_{(R,B)}(A)$.

Remark 7: The function $\phi_{(R,B)}$ preserves the rank of a matrix, since it just inserts duplicates of columns. That is, $\mathrm{rk}_2(\phi_{(R,B)}(A)) = \mathrm{rk}_2(A)$.

For an EIC problem (R, B), we will use the set $\mathcal{A}_{(R,B)}$ to restrict the domain of minrank, resulting in the *restricted-minrank:*

Definition 15: The *restricted minrank* of a graph $G = (V, E)$ in the field $\mathbb{F}_2$ over the set of matrices $\mathcal{A} \subseteq \mathbb{F}_2^{|V| \times |V|}$, denoted $\text{r-minrk}_2(G, \mathcal{A})$, is the rank of the lowest-rank matrix $A' \in \mathcal{A}$ which fits G:

$$\text{r-minrk}_2(G, \mathcal{A}) = \min\{\mathrm{rk}_2(A') : (A' \in \mathcal{A}) \wedge (A' \text{ fits } G)\}.$$

Lemma 1: Let G be the problem graph for an EIC problem defined by (R, B). Let A' be a matrix that fits G, and assume that $A' = \phi_{(R,B)}(A)$ for some matrix $A \in \mathbb{F}_2^{|V| \times m}$. Suppose that $A^{(r)} \in \mathbb{F}_2^{r \times m}$ is a matrix whose rows are r rows of A which span the rowspace of A; thus, the rowspace of $A^{(r)}$ is equal to that of A. Then $A^{(r)}$ is a centralized linear broadcast solution to (R, B).

Proof: Let $r := \mathrm{rk}_2(A) = \mathrm{rk}_2(A')$. Without loss of generality, suppose that the first r rows $A_1, ..., A_r$ span the rowspace A. Then for any $(u, a) \in P$, the corresponding row of A can be represented as $A_{(u,a)} = \sum_{i=1}^{r} \lambda_i^{(u,a)} A_i$ for some $\lambda_1^{(u,a)}, ..., \lambda_r^{(u,a)} \in \mathbb{F}_2$. For ease of notation let $\ell := (u, a)$, so $A_\ell = A_{(u,a)}$ denotes the row of A indexed by the request pair $(u, a) \in P$ (which corresponds to node u requesting block a) as in Definition 5.

Let β be the matrix

$$\beta = \begin{bmatrix} -- A_1 -- \\ \vdots \\ -- A_r -- \end{bmatrix},$$

so that rows of $\beta\mathcal{D}$ are the encoded messages which are broadcast by the centralized source:

$$\beta\mathcal{D} = \begin{bmatrix} A_1 \cdot \mathcal{D} \\ A_2 \cdot \mathcal{D} \\ \vdots \\ A_r \cdot \mathcal{D} \end{bmatrix}.$$

Then $[\lambda_1^{(\ell)}, ..., \lambda_r^{(\ell)}] \cdot (\beta\mathcal{D}) = A_\ell \cdot \mathcal{D}$. Thus if some node u has the set of encoded messages $\{A_1 \cdot \mathcal{D}, A_2 \cdot \mathcal{D}, ..., A_r \cdot \mathcal{D}\}$, it can compute $A_\ell \cdot \mathcal{D}$.

We next define the vector $\mu \in \mathbb{F}_2^m$: let $\mu_b = A_{\ell b}$ if $B_{ub} = 1$, otherwise let $\mu_b = 0$. Equivalently, $\mu := B_u \odot A_\ell$, where we use $\odot$ to denote Hadamard (entry-wise) product. By definition of B, node u has $\boldsymbol{e}_b\mathcal{D}$ for any $b \in [m]$ such that $B_{ub} = 1$. We claim that the only $b \in [m]$ so that $A_{\ell b} = 1$ and $B_{ub} \neq 1$ is $b = a$. Indeed, if $A_{\ell b} = A_{(u,a),b} = 1$, then $A'_{(u,a),(v,b)} = 1$ for all v so that $(v, b) \in P$, by the definition of $\phi_{(R,B)}$. Since A' fits G, this implies that either $a = b$ or that node u has the block b. But if $B_{ub} \neq 1$, then u does not have b, leaving only the possibility that $a = b$.

As a result, all non-zero entries of $A_\ell - \boldsymbol{e}_a$ are indexed by some b such that $B_{ub} = 1$, so

$$\mu = B_u \odot A_\ell = A_\ell - \boldsymbol{e}_a.$$

Then we construct the decoding vector $\boldsymbol{\alpha}^{(u,a)}$ as

$$\boldsymbol{\alpha}^{(u,a)} := [\lambda_1^{(\ell)} \ldots \lambda_r^{(\ell)} \;\; -\mu_1 \cdots -\mu_m].$$

so that decoding is done by computing

$$\begin{aligned} &\boldsymbol{\alpha}^{(u,a)} \cdot \left[\frac{\beta}{\mathrm{diag}(B_u)}\right] \cdot \mathcal{D} \\ &= [\lambda_1^{(\ell)}, ..., \lambda_r^{(\ell)}] \cdot (\beta\mathcal{D}) - (\mu \odot B_u) \cdot \mathcal{D} \\ &= [\lambda_1^{(\ell)}, ..., \lambda_r^{(\ell)}] \cdot (\beta\mathcal{D}) - (A_\ell - \boldsymbol{e}_a) \cdot \mathcal{D} \\ &= A_\ell \cdot \mathcal{D} - (A_\ell - \boldsymbol{e}_a) \cdot \mathcal{D} \\ &= \boldsymbol{e}_a\mathcal{D}. \end{aligned}$$

This shows that $A^{(r)}$ is a centralized linear broadcast solution to (R, B), as per Definition 11. □

We next generalize Theorem 1 to EIC problems which are not single unicast.

Theorem 2: Given a EIC problem (R, B), corresponding problem graph G, and column repetition function $\phi_{(R,B)}$ with range $\mathcal{A}_{(R,B)}$,

$$(C)_{(R,B)} = \text{r-minrk}_2(G, \mathcal{A}_{(R,B)}).$$

Proof: Let $A' \in \mathcal{A}_{(R,B)}$ be the matrix of lowest rank in $\mathcal{A}_{(R,B)}$ that fits G and let $r := \text{rk}_2(A') = \text{r-minrk}_2(G, \mathcal{A}_{(R,B)})$. Let $A \in \mathbb{F}_2^{|V|\times m}$ such that $\phi_{(R,B)}(A) = A'$. By Lemma 1, a matrix $A^{(r)}$ composed of r linearly independent rows of A is a centralized linear solution to (R, B) of length r (by the choice of r, the rowspan of $A^{(r)}$ equals the rowspan of A). Since a centralized source is able to construct each of these messages for this solution (that is, the rows of matrix $A^{(r)}\mathcal{D}$) we conclude that

$$(C)_{(R,B)} \le r = \text{r-minrk}_2(G, \mathcal{A}_{(R,B)}).$$

For the other direction, suppose that $Z \in \mathbb{F}_2^{s\times m}$ is a linear solution to (R, B) for some $s \in \mathbb{Z}^+$. Let $\mathbf{z}_i \in \mathbb{F}_2^m$ denote the i^{th} row of Z. We will show that the row span of Z contains the row span of some matrix A such that $A' := \phi_{(R,B)}(A)$ fits G. Consider some $(u, a) \in P$. By the definition of a linear solution, there exists some vector $\boldsymbol{\alpha}^{(u,a)}$ such that

$$\boldsymbol{e}_a = \boldsymbol{\alpha}^{(u,a)} \cdot \left[\frac{Z}{\text{diag}(B_u)}\right].$$

Write

$$\boldsymbol{\alpha}^{(u,a)} = [\lambda_1^{(u,a)} \dots \lambda_s^{(u,a)}\ \mu_1 \dots \mu_m]$$

for some $\lambda_i^{(u,a)}, \mu_j \in \mathbb{F}_2$, so that

$$\boldsymbol{e}_a = \sum_{i=1}^{s} \lambda_i^{(u,a)} \mathbf{z}_i + \sum_{j=1}^{m} \mu_j B_{uj} \boldsymbol{e}_j.$$

Let $A_{(u,a)} \in \mathbb{F}_2^m$ be the vector

$$A_{(u,a)} = \boldsymbol{e}_a - \sum_{j=1}^{m} \mu_j B_{uj} \boldsymbol{e}_j = \sum_{j=1}^{s} \lambda_j^{(u,a)} \mathbf{z}_j.$$

Then $A_{(u,a)}$ is in the row span of Z, and moreover the a'th entry of $A_{(u,a)}$ satisfies

$$A_{(u,a),a} = 1.$$

Additionally, for any block $b \in [m]$ with $b \neq a$ such that $B_{ub} = 0$,

$$A_{(u,a),b} = (\boldsymbol{e}_a)_b - \mu_b B_{ub} = 0.$$

Let $A \in \mathbb{F}_2^{|P|\times m}$ be the matrix whose rows are given by $A_{(u,a)}$ for $(u, a) \in P$. Let $A' := \phi_{(R,B)}(A)$. We claim that A' fits G. Indeed, we have for all $(u, a) \in P$ that

$$A'_{(u,a),(u,a)} = A_{(u,a),a} = 1$$

by the above, and so the first requirement of Definition 5 is met.

To see the second requirement of Definition 5, first note that for all $b \neq a$, we have

$$A'_{(u,a),(w,b)} = A_{(u,a),b}$$

which by the above is non-zero only if $B_{ub} = 1$; that is, only if there is an edge (of the "first type") from $v_{(u,a)}$ to $v_{(w,b)}$ in G. Second, there is always an edge (of the "second type") from $v_{(u,a)}$ to $v_{(w,a)}$. Thus, the only non-zero off-diagonal entries of A' correspond to edges in G, and the second requirement of Definition 5 is satisfied.

Thus, for any linear solution Z of length s, there is a matrix A so that the row span of Z contains the row span of A and so that $A' = \phi_{(R,B)}(A)$ fits G. Thus,

$$\begin{aligned}(C)_{(R,B)} &\ge s \ge \text{rk}_2(A) = \text{rk}_2(A') \\ &\ge \text{r-minrk}_2(G, \mathcal{A}_{(R,B)}).\end{aligned}$$

This completes the proof. □

Because the minrank gives the optimal linear solution for a centralized sender with all data blocks, our definition of the problem graph is a natural extension of index coding and the side information graph to the embedded index coding model. In the following it will be helpful to use the following theorem from [3] relating minrank to some other standard graph properties. For an undirected graph G, the *chromatic number* $\chi(G)$ is the minimum number of colors required to color the vertices of G so that no neighboring vertices have the same color. The *clique number* $\omega(G)$ is the size of the largest clique in G. The independence number, denoted $\alpha(G)$, is the set of the largest independent set in G, so $\alpha(G) = \omega(\overline{G})$.

Throughout the paper, we will apply these notions to directed graphs, rather than undirected graphs. To avoid confusion, we will adopt the following notation.

Definition 16: Let $G = (V, E)$ be a directed graph. The undirected graph $G_{either} = (V, E_{either})$ is the graph so that $\{u, v\} \in E_{either}$ whenever either $(u, v) \in E$ or $(v, u) \in E$. The undirected graph $G_{both} = (V, E_{both})$ is the graph so that $\{u, v\} \in E_{both}$ whenever both $(u, v) \in E$ and $(v, u) \in E$.

We note that for any directed graph G, $\overline{G_{both}} = (\overline{G})_{either}$ and $(\overline{G})_{both} = \overline{G_{either}}$. For undirected graphs G, the following relationships are known [3].

Theorem 3 ([3]): Let G be an undirected graph. Then $\omega(\overline{G}) \le \text{minrk}_2(G) \le \chi(\overline{G})$.

A similar bound applies to our restricted version of minrank:

Corollary 1: Given a EIC problem (R, B), a corresponding problem graph $G = (V, E)$, and the column repetition function $\phi_{(R,B)}$ with range $\mathcal{A}_{(R,B)}$, we have:

$$\begin{aligned}&\omega(\overline{G_{either}}) \le \text{minrk}_2(G) \le \\ &\text{r-minrk}_2(G, \mathcal{A}_{(R,B)}) \le \chi(\overline{G_{both}}).\end{aligned}$$

Proof: Since r-minrk$_2$ is a minimization over a smaller set of matrices than minrk$_2$, clearly $\text{minrk}_2(G) \le \text{r-minrk}_2(G, \mathcal{A})$. We also have

$$\omega(\overline{G_{either}}) \le \text{minrk}_2(G_{either}) \le \text{minrk}_2(G),$$

where the first inequality follows from Theorem 3 and the second inequality follows from the fact that any matrix that fits G also fits G_{either}. Thus it remains the show the final inequality in the statement of the Corollary.

Using a similar approach to [3], we show the final inequality by describing a matrix $A' \in \mathcal{A}$ that fits G such that $\text{rk}_2(A') \le \chi(\overline{G_{both}})$. This is enough to establish the final inequality, since we will have

$$\text{r-minrk}_2(G, \mathcal{A}) \le \text{rk}_2(A') \le \chi(\overline{G_{both}}).$$

By the definition of chromatic number, there is a partition of V into sets $C_1, \ldots, C_{\chi(\overline{G_{both}})}$ so that each C_i forms a clique in G_{both}. Let $C \subseteq V$ be a clique from such a partition. Define a vector $\mathbf{c}^{(C)} \in \mathbb{F}_2^m$ so that the a'th entry of $\mathbf{c}^{(C)}$ is given by

$$c_a^{(C)} = \begin{cases} 1 & \exists u \in [n] \text{ so that } v_{(u,a)} \in C \\ 0 & \text{else} \end{cases}$$

Now, define a matrix $A \in \mathbb{F}_2^{|P| \times m}$ with rows indexed by elements of P, so that if $v_{(u,a)}$ is in the clique C, then

$$A_{(u,a)} = \mathbf{c}^{(C)}.$$

Let $A' = \phi_{(R,B)}(A)$. Thus by definition of $\phi_{(R,B)}(A)$ (see Remark 7):

$$\mathrm{rk}_2(A') = \mathrm{rk}_2(A) \leq \chi(\overline{G_{both}})$$

since there are only $\chi(\overline{G_{both}})$ distinct rows of A.

Now we just need to show that A' fits G. Consider some row $A'_{(u,a)}$, where $A_{(u,a)} = \mathbf{c}^{(C)}$ for some clique C, and choose some $(w,b) \neq (u,a)$ such that $A'_{(u,a)(w,b)} = 1$. If $a = b$, then $(v_{(u,a)}, v_{(w,b)}) \in E$ is an edge of the "second type." On the other hand, if $a \neq b$, then by the definition of $\phi_{(R,B)}$, $c_b^{(C)} = 1$, so there exists some $x \in [n]$ so that $v_{(x,b)} \in C$. Since C is a clique in G_{both}, $(v_{(u,a)}, v_{(x,b)}) \in E$. By the definition of problem graph this is an edge of the "first type," so $B_{ub} = 1$, so we also have $(v_{(u,a)}, v_{(w,b)}) \in E$. Thus any non-zero off-diagonal entry of A corresponds to an edge in G. Moreovoer, the diagonal entries of A' are

$$A'_{(u,a),(u,a)} = A_{(u,a),a} = c_a^{(C)}$$

where $v_{(u,a)} \in C$, so this is 1 from the definition of $c_a^{(C)}$. Thus, A' fits G. □

B. Cost of Decentralization: $(D) \leq 2(C)$

It can easily be seen that $(C) \leq (D)$, that is, that the minimum length of a decentralized embedded index code is at least the minimum length of a centralized solution. Indeed, the (D) messages transmitted in the decentralized solution can all be constructed by a centralized source which has access to all data blocks. Thus we are interested in how much larger (D) can be than (C). Intuitively, one way we can construct a decentralized solution is to create a centralized solution, and split the messages into parts that can be constructed by nodes using their side information. In fact, we show that each message of a centralized solution only needs to be split into two messages to make a decentralized solution, so (D) is no more than a factor of 2 worse than (C):

Theorem 4: Given a solvable EIC problem (R, B), $(D)_{(R,B)} \leq 2 \cdot (C)_{(R,B)}$.

Proof: Let P be the set of requirement pairs for (R, B). Let $G = (V, E)$ be the problem graph for (R, B) and let $\phi_{(R,B)}$ be the corresponding column expansion function, with image $\mathcal{A}_{(R,B)}$. By Theorem 2, $(C)_{(R,B)} = \text{r-minrk}_2(G, \mathcal{A}_{(R,B)})$. Let $A' \in \mathcal{A}_{(R,B)}$ be a matrix with $A' = \phi_{(R,B)}(A)$ for some $A \in \mathbb{F}_2^{|V| \times m}$ so that

$$\mathrm{rk}_2(A') = \mathrm{rk}_2(A) = (C)_{(R,B)} =: r$$

and so that A' fits G. By Lemma 1, there is a matrix $A^{(r)}$ with rows $A_1, \ldots, A_r$ that is a centralized linear broadcast solution to (R, B). We will show how to simulate this centralized solution using only $2r$ messages.

Since $A_1, \ldots, A_r$ are rows of A, they correspond to requirement pairs in P. Fix $\ell \in [r]$ and suppose that A_ℓ corresponds to $(u, a) \in P$. Since A' fits G, the diagonal entries of A' are non-zero. This means that $A_{\ell,a} \neq 0$. Further, for $b \neq a$, if $A_{\ell,b} \neq 0$ then there is an edge of the "first type" in G: that is, $B_{ub} = 1$, which means that node u has block b. Thus, node u is able to compute

$$\sum_{b: B_{ub}=1} A_{\ell b} \boldsymbol{e}_b \cdot \mathcal{D} = A_\ell \cdot \mathcal{D} - \boldsymbol{e}_a \cdot \mathcal{D}$$

using the information it had before and the information it receives.

The decentralized scheme is then as follows. For each $\ell \in [r]$ corresponding to (u, a), we have two broadcasts:

1) Node u broadcasts $\sum_{b: B_{ub}=1} A_{\ell b} \boldsymbol{e}_b \cdot \mathcal{D}$. That is, A_ℓ is a row of $\beta^{(u)}$.
2) Fix any other node w so that $B_{wa} = 1$. We note that such a node exists because (R, B) is solvable. Then node w broadcasts $\boldsymbol{e}_a \cdot \mathcal{D}$. That is, $\boldsymbol{e}_a$ is a row of $\beta^{(w)}$.

Now every node can add together the two broadcasts corresponding to $\ell \in [r]$ to obtain $A_\ell \cdot \mathcal{D}$. Since $A^{(r)}$ is a linear centralized solution to (R, B), this scheme is a linear centralized solution to (R, B). □

We note that the proof of Theorem 4 crucially uses the EIC formulation; this shows why considering EIC separately as a special case of multi-sender index coding can be valuable.

C. Cost of Task-Based Solutions: Upper Bound for (T)

We first show how the minrank can be used to re-formulate the length (T) of the optimal task-based solution. Let (R, B) be an EIC problem, with problem graph $G = (V, E)$. Recall from Definition 10 that, given a task-based solution T, the neighborhood partition $\{\tilde{N}_1, \ldots, \tilde{N}_n\}$ is a partition of V so that $\tilde{N}_w \subseteq N_w$ is the set of vertices $v_{(u,a)}$ so that w is responsible for (u, a) in T.

For $\tilde{N}_w \subseteq N_w$ corresponding to a task-based solution T, let $G|_{\tilde{N}_w}$ denote the induced subgraph of G on the vertices $\tilde{N}_w$. As per Remark 5, each $\tilde{N}_w$ corresponds to an EIC problem $(R^{(w)}, B^{(w)})$, over the set of blocks $\{a \in [m] : \exists u \in [n] \text{ s.t. } v_{(u,a)} \in \tilde{N}_w\}$. Thus by definition of $\tilde{N}_w$, node w has all blocks used in problem $(R^{(w)}, B^{(w)})$ and any centralized solution to $(R^{(w)}, B^{(w)})$ can be broadcast by w. Note that such a solution can easily be used as a self-contained part of a solution to the problem (R, B) with the full set of m blocks. To do so, we just insert zeros in encoding and decoding vectors for blocks in $[m]$ not used in $(R^{(w)}, B^{(w)})$. Then the messages of the solution to $(R^{(w)}, B^{(w)})$ can be used by vertices of $\tilde{N}_w$ as in the subproblem.

We first show how solutions to these subproblems can be used as building blocks for task-based solutions.

Lemma 2: Let G be a problem graph for a solvable EIC problem (R, B). Let $\{\tilde{N}_1, ..., \tilde{N}_n\}$ be a neighborhood partition. Let $G|_{\tilde{N}_w}$ be the subgraph of G induced by $\tilde{N}_w$ and let $(R^{(w)}, B^{(w)})$ be the problem with problem graph $G|_{\tilde{N}_w}$ for all $w \in [n]$. Then any set of solutions $\{\beta^{(w)} \in \mathbb{F}^{h_w \times m} : w \in [n], h_w \in \mathbb{Z}^+\}$ for problems $\{(R^{(w)}, B^{(w)}) : w \in [n]\}$ forms a task-based solution to (R, B) with length $\sum_{i=1}^n h_i$.

Proof: For each vertex $v_{(u,a)}$ of G there is some $\tilde{N}_w$ such that $v_{(u,a)} \in \tilde{N}_w$. Let $\beta^{(w)}$ be the centralized linear broadcast solution to EIC problem $(R^{(w)}, B^{(w)})$, where $\beta^{(w)} \in \mathbb{F}^{h_w \times m}$ for some $h_w \in \mathbb{Z}^+$. Then there exists some $\boldsymbol{\alpha}^{(u,a)}$ such that $\boldsymbol{e}_a = \boldsymbol{\alpha}^{(u,a)} \cdot \left[\frac{\beta^{(w)}}{\mathrm{diag}(B_u)}\right]$. Since there is such a vertex $v_{(u,a)}$ for each $(u, a) \in P$, all requests in P are satisfied in this way by some $\beta^{(w)} \cdot \mathcal{D}$. By definition of $(R^{(w)}, B^{(w)})$, each $\beta^{(w)} \cdot \mathcal{D}$ for $w \in [n]$ can be broadcast by node w. Thus $\beta^{(1)}, ..., \beta^{(n)}$ forms a task-based solution to (R, B) with length $\sum_{i=1}^n h_i$. $\square$

We can then compute the length of an optimal task-based solution, (T), in terms of neighborhood partitions.

Lemma 3: Given a solvable EIC problem (R, B), let $\mathscr{N}$ be the set of all possible neighborhood partitions (as in Definition 10). For $\{\tilde{N}_1, ..., \tilde{N}_n\} \in \mathscr{N}$, let $(R^{(w)}, B^{(w)})$ be the EIC problem induced by $\tilde{N}_w$. Then

$$(T)_{(R,B)} = \min_{\{\tilde{N}_1,...,\tilde{N}_n\} \in \mathscr{N}} \sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})}).$$

Proof: Observe that since (R, B) is solvable, $\mathscr{N}$ is non-empty. We first show that

$$(T)_{(R,B)} \leq \min_{\{\tilde{N}_1,...,\tilde{N}_n\} \in \mathscr{N}} \sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})}).$$

Consider the neighborhood partition $\{\tilde{N}_1, ..., \tilde{N}_n\}$ which minimizes $\sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})})$. A possible task-based solution T can be constructed by optimally solving the centralized index coding problem $(R^{(w)}, B^{(w)})$ defined by each $G|_{\tilde{N}_w}$ with sending node w, as shown in Lemma 2. By Theorem 2, each centralized subproblem optimal solution has length $\text{r-minrk}_2(G|_{\tilde{N}_w}, \mathcal{A}_{(R^{(w)},B^{(w)})})$, so the total length of T is $\sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})})$.

Next we show

$$\min_{\{\tilde{N}_1,...,\tilde{N}_n\} \in \mathscr{N}} \sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})}) \leq (T)_{(R,B)}.$$

Let T be the optimal task-based solution, with length $(T)_{(R,B)}$. Construct $\{\tilde{N}_1, ..., \tilde{N}_n\}$ so that

$$\tilde{N}_w = \{v_{(u,a)} : w \text{ is responsible for } (u, a) \text{ in } T\}.$$

By the definition of a task-based solution, each vertex is assigned to exactly one such $\tilde{N}_w$, so we have a neighborhood partition $\{\tilde{N}_1, ..., \tilde{N}_n\} \in \mathscr{N}$. The centralized index coding problems $(R^{(w)}, B^{(w)})$ for each $w \in [n]$ have problem graphs $G|_{\tilde{N}_w}$ and optimal solutions of length $\text{r-minrk}_2(G|_{\tilde{N}_w}, \mathcal{A}_{(R^{(w)},B^{(w)})})$ (Theorem 2). If $(T)_{(R,B)} < \sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})})$, then the solution to $(R^{(w)}, B^{(w)})$ for some $w \in [n]$ must have length strictly less than $\text{r-minrk}_2(G|_{\tilde{N}_w}, \mathcal{A}_{(R^{(w)},B^{(w)})})$. This contradicts Theorem 2. Thus $(T)_{(R,B)} \geq \min_{\{\tilde{N}_1,...,\tilde{N}_n\} \in \mathscr{N}} \sum_{i=1}^n \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)},B^{(i)})})$. $\square$

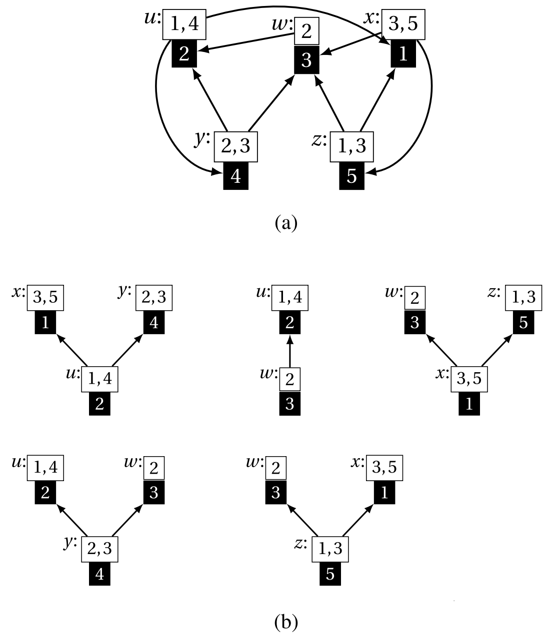


Fig. 6. (a) Example of a problem graph for a single unicast EIC problem that has $(C) < (D) < (T)$. Each pair of boxes is a vertex in the problem graph, where black boxes contain indices of requested data blocks and the white boxes contain indices of side information blocks; the label u, w, x, y, z indicates which node the vertex corresponds to. (b) The subgraphs induced by each node along with its sender neighborhood.

The following upper bound on $(T)_{(R,B)}$ follows from Lemma 3 and Corollary 1.

Lemma 4: Given a solvable EIC problem defined by (R, B), let $\mathscr{N}$ be the set of all possible neighborhood partitions (Definition 10). Then

$$(T)_{(R,B)} \leq \min_{\{\tilde{N}_1,...,\tilde{N}_n\} \in \mathscr{N}} \sum_{i=1}^n \chi\left(\overline{(G|_{\tilde{N}_i})_{both}}\right). \quad (1)$$

In Section V-B, we will use Lemma 4 to develop algorithms to approximate the optimal neighborhood partition (in the sense that the right hand side of (1) is minimized), by reducing the problem to the minimum cover problem.

D. An Example Where $(C) \neq (D) \neq (T)$

Figure 6a is an example of a single unicast EIC problem (R, B) for which $(C) < (D) < (T)$. First consider (C): by inspection, a central source with all data blocks could send messages $\mathcal{D}_2 \oplus \mathcal{D}_4$, $\mathcal{D}_3$, and $\mathcal{D}_1 \oplus \mathcal{D}_5$ so that all five nodes can decode their requested block, but no combination of fewer messages suffices. No solution of length at most 2 is possible (see Appendix), so, $(C) = 3$.

Next consider (D). A solution of minimum-length is: node y broadcasts $\mathcal{D}_2 \oplus \mathcal{D}_3$, node z broadcasts $\mathcal{D}_3 \oplus \mathcal{D}_1$, node u broadcasts $\mathcal{D}_4$, and node x broadcasts $\mathcal{D}_5$. It can be checked that this is indeed a minimum-length solution (see Appendix). Thus, $(D) = 4$.

Finally, consider (T). Then out-neighborhoods of each node, as shown in Figure 6, are the subgraphs over which we can apply index coding (Remark 5). In particular, we construct the neighborhood partition from these subgraphs (Definition 10). Since the graph induced by each neighborhood is acyclic, as shown in [39] there is no way to do any non-trivial coding in any subgraph to a code shorter than the uncoded solution. Thus any task-based solution requires that all blocks be broadcast uncoded. Since there are five blocks that need to be sent, we have $(T) = 5$.

E. Separations Between (T) and (C)

We next give a condition on the problem graph G which guarantees that (T) is strictly larger than (C), with a gap as big as the gaps from the graph-theoretic minrank bounds.

Lemma 5: Given a solvable EIC problem defined by (R, B) and corresponding problem graph G, if

$$(\chi(G_{either}) - 1)\chi(\overline{G_{both}}) < |V|,$$

then there is an optimal task-based solution with neighborhood partition $\{\tilde{N}_1, ..., \tilde{N}_n\}$ so that

$$(C)_{(R,B)} \leq \chi(\overline{G_{both}}) < \sum_{i=1}^{n} \omega\left(\overline{(G|_{\tilde{N}_i})_{either}}\right) \leq (T)_{(R,B)}$$

Proof: Let $V := V(G)$. First note that for any undirected graph H, $\alpha(H) = \omega(\overline{H})$ and $\frac{|V|}{\chi(H)} \leq \alpha(H)$ (see, e.g., [41]). Consider coloring each graph $(G|_{\tilde{N}_i})_{either}$ (induced by an element of the neighborhood partition) individually, compared to coloring all of G_{either} at once. Since every node in $\tilde{N}_i$ shares a neighbor in G_{either} (i.e. any of the vertices $v_{(i,a)} \in V$ for some $a \in [m]$) there is a color in the minimum coloring of G_{either} not necessary to color $(G|_{\tilde{N}_i})_{either}$. Thus $\chi((\tilde{N}_i)_{either}) \leq \chi(G_{either}) - 1$. Putting these steps together, we have

$$\begin{aligned}\sum_{i=1}^{n} \omega(\overline{(G|_{\tilde{N}_i})_{either}}) &= \sum_{i=1}^{n} \alpha((G|_{\tilde{N}_i})_{either}) \\ &\geq \sum_{i=1}^{n} \frac{|V(G|_{\tilde{N}_i})|}{\chi((G|_{\tilde{N}_i})_{either})} \\ &\geq \sum_{i=1}^{n} \frac{|V(G|_{\tilde{N}_i})|}{\chi(G_{either}) - 1} \\ &= \frac{|V|}{\chi(G_{either}) - 1}.\end{aligned}$$

Since by assumption we have $\frac{|V|}{\chi(G_{either})-1} > \chi(\overline{G}_{either}) = \chi(\overline{G_{both}})$, this proves the strict inequality in the claim. Above, the first inequality follows from applying $\omega(\overline{H}) \geq \frac{|V|}{\chi(H)}$ to each induced graph $(G|_{\tilde{N}_i})_{either}$. The final equality follows from the fact that $\sum_{i=1}^{n} |V(G|_{\tilde{N}_i})| = |V|$, which is true because the graphs in the set $\{G|_{\tilde{N}_i} : i \in [n]\}$ are induced by corresponding elements of the vertex partition $\{\tilde{N}_1, ..., \tilde{N}_n\}$.

Finally, the remaining two inequalities in the statement of the lemma follow from Corollary 1. □

Lemma 5 establishes a gap between $(C)_{(R,B)}$ and $(T)_{(R,B)}$ whenever $|V| > (\chi(G_{either}) - 1)(\chi(\overline{G_{both}}))$, so we note here a few simple graphs which illustrate how this may or may not be the case.

Suppose that G is a directed clique, which for example would be the case if every node wanted a different block and had every block they did not want. Then $\chi(G_{either}) = n$, while $\chi(\overline{G_{both}}) = 1$, so

$$(\chi(G_{either}) - 1)(\chi(\overline{G_{both}})) = n - 1 < n = |V|.$$

In this case, Lemma 5 establishes a gap. Similarly, Lemma 5 establishes a gap for graphs consisting of multiple disconnected directed cliques.

On the other hand, consider a directed cycle, which for example would be the case if every node i wanted a different block, which was held only by node $i - 1 \pmod n$. Then Lemma 5 does not establish a gap: $\chi(G_{either}) = 2$ or 3 and $\chi(\overline{G_{both}}) = n$, so $(\chi(G_{either}) - 1)(\chi(\overline{G_{both}})) \not< n$.

V. ALGORITHMS

In this section, we use results from the previous section to design two heuristics for finding good EIC solutions. We also demonstrate empirically that our algorithms perform well.

First, we use Theorem 4 to give an algorithm which produces an EIC solution that is optimal within a factor of two. We show empirically that our algorithm is faster (more precisely, has a smaller search space) than the algorithm of [22]. We note that our algorithm is tailored for EIC while the approach of [22] works more generally in the multi-sender model. This demonstrates the value of focusing on EIC as a special case.

Second, we use Lemma 4 to give a heuristic algorithm to design a task-based scheme for an EIC problem. We show empirically that the quality of solution returned by our algorithm is within a small constant factor (at most 1.4 in our experiments) of the optimal *centralized* scheme.

We describe both of these in more detail below.

A. Approximating (D)

The proof of Theorem 4 gives an algorithm to approximate the optimal decentralized solution to an EIC problem, which we detail in Algorithm 1. Algorithm 1 first computes the exact optimal centralized solution with length $(C)_{(R,B)}$ and then uses the transformation outlined in the proof of Theorem 4 to arrive at a decentralized solution with length at most $2(C)_{(R,B)}$. We note that in practice the optimal centralized solution could also be approximated, leading to a decentralized solution of length at most twice the cost of the approximation.

Algorithm 1: Given an EIC problem (R, B):

1) Construct the problem graph G
2) Find $A \in \mathbb{F}_2^{|V| \times m}$ such that $\mathrm{rk}_2(A) = \mathrm{rk}_2(\phi_{(R,B)}(A)) = \text{r-minrk}_2(G, \mathcal{A}_{(R,B)})$
   a) Let $r := rk_2(A)$
   b) Let $A_1, ..., A_r$ be linearly independent rows of A
3) For each $A_\ell \in \{A_1 ... A_r\}$:
   a) Let $\ell = (u, a)$
   b) Node u: broadcast $\sum_{b:B_{ub}=1} A_{\ell b} \boldsymbol{e}_b \cdot \mathcal{D}$
   c) For an arbitrary node w s.t. $B_{wa} = 1$: compute and broadcast $\boldsymbol{e}_a \cdot \mathcal{D} = \mathcal{D}_a$

We compare Algorithm 1 to the algorithms in previous work [18], [22], which apply more generally to any multi-sender index coding problem but can in particular be applied to EIC. In the multi-sender index coding problem solved by [18], [22], there is a set of sender nodes and a set of receiver nodes, which do not overlap. Each sender node has some subset of the data that it can use in constructing messages to send to the receivers, and each receiver requests one of the data items and has a subset of the others as side-information. Since the algorithms of [18], [22] only apply to single unicast EIC problems (Definition 13), we restrict our analysis to that case. To use the algorithms of [18], [22], for an EIC problem (R, B), we construct a sender and a receiver for each of the original nodes. Thus the k^{th} sender has the data blocks indicated by B_k and the k^{th} receiver has as side-information the data indicated by B_k. Let n_k be the number of data blocks that sender k has, so $n_k = |B_k|_1$.

In our method and in those of [18], [22], the strategy is to search over a set of matrices and find the one of minimum rank. Thus, to compare the complexity of our approach to previous work, we can compare the size of the state spaces.[3] As we describe below, the search space for Algorithm 1 is much smaller than that for the other algorithms. More precisely, we can see analytically that

$$|\text{Search space for [18]}| = |\text{Search space for Algorithm 1}|^{\Theta(n)},$$

and we observe empirically that something similar may be true for the work of [22]. We elaborate below.

1) Size of Search Spaces for [18] [22], and Algorithm 1: We begin by describing two previous approaches [18], [22] and stating the size of their search spaces, as well as the search space in Algorithm 1; we will compare these expressions in Section V-A2.

In the work of Kim and No [18], the authors show (Theorem 1 in that work) that an optimal linear strategy can be obtained by finding a matrix F of lowest rank that "fits" (see Definition 3 and Corollary 1 of [18]) the given problem. To use our notation, this matrix F is created by essentially stacking the $\beta^{(i)}$ matrices for each sender i on top of each other. The precise size of the search space is complicated and depends on the problem (see Remark 2 of [18]), but it could be[4] as large as

$$2^{\left(n \sum_{k=1}^{n} n_k\right) - \left(\sum_{k=1}^{n} (m - n_k)\right)}. \tag{2}$$

[3]We note that if the minrank is computed exactly, then Combined LT-CMAR approach of previous work becomes an exact algorithm, while our algorithm is a two-approximation.

[4]In slightly more detail, the dimension of the search space in [18] might be counted as follows. Essentially, that work seeks to find the matrix of smallest rank by filling in the matrices $\beta^{(1)}, \ldots, \beta^{(n)} \in \mathbb{F}^{m \times n}$ stacked next to each other to make a large $m \times n^2$ matrix. For each sender $k \in [n]$, there are n_k rows of $\beta^{(k)}$ that may be nonzero, leading to $n \sum_k n_k$ free variables. However, there are additional linear constraints: [18] shows that one may assume that for any data block $i \in [m]$ and node $j \in [n]$ so that node j wants block i, $\sum_{k \in [n]} \beta_{i,j}^{(k)} = 1$, while for any $i \in [m]$ and $j \in [n]$ so that node j neither wants nor has block i, $\sum_{k \in [n]} \beta_{i,j}^{(k)} = 0$. This imposes an additional $m - n_j$ linear constraints for each node j: one constraint for every block $i \in [m]$ that bide j does not have. Thus the dimension is at least the number of variables minus the number of constraints, which gives the expression in (2).

We note that, in [18], m refers to the number of receivers and n is the number of messages, while for us m and n are reversed; we have used our notation above.

The work of Li et al. [22], introduces an approach called Combined Lower-Triangularization and Common-Message-Aware Reduction (Combined LT-CMAR). The idea is to reduce the search space by both enforcing some structure and by removing redundancies between senders. We give a high-level overview of the approach here, and refer the reader to [22], Section IIIB, for details. Given a set S of senders and letting $K := |S|$, all "weak" senders[5] are removed, to leave a sender set S^* of size K^*. That work considers a set $\overline{\mathcal{C}}''(S^*)$ of possible encoding matrices (that is, matrices $\beta^{(1)}, ..., \beta^{(K^*)}$, using our notation in Definition 2) that exhibit particular structure. In particular, they show that they can make these matrices lower triangular, and they show how to take into account overlapping messages between senders. (See Section IIIB in [22] for more details). This results in an optimization problem that we describe below. Suppose that $\mathbf{M}$ is any matrix that fits G. For each node $j \in [n]$, $\mathcal{B}_j \subseteq [m]$ is the set of indices of blocks that j does *not* have as side information. In particular, $[\mathbf{M}]_{j,\mathcal{B}_j}$ does not depend on the choice of $\mathbf{M}$, so long as it fits G. With this notation, Proposition 3 in [22] shows that the optimal broadcast rate is the optimal value of the following optimization problem:

$$\begin{aligned}
&\text{minimize} \\
&\qquad \sum_{k=1}^{K^*} \mathrm{rk}_2(\overline{\mathbf{C}}''_k) \\
&\text{subject to} \\
&\qquad (i)(\overline{\mathbf{C}}''_1, ..., \overline{\mathbf{C}}''_{K^*}) \in \overline{\mathcal{C}}''(S^*) \\
&\qquad (ii)[\mathbf{M}]_{j,\mathcal{B}_j} \in \mathrm{rowspan}_2([\overline{\mathbf{C}}''_k]_{:,\mathcal{B}_j}), \forall j \in [1:N]
\end{aligned}$$

Above, the matrices $\overline{\mathbf{C}}''_k \in \mathbb{F}_2^{n_k \times n}$ play the role of our repair matrices $\beta^{(k)}$. The search space size for Combined LT-CMAR (that is, the size of $\overline{\mathcal{C}}''(S^*)$) is bounded by

$$2^{\sum_{k=1}^{n}(n_k^2+n_k)/2 - \sum_{k=1}^{n}((n_k^{(c)})^2 + n_k^{(c)})/2}, \tag{3}$$

where $n_k^{(c)}$ is the maximum number of blocks that node k has in common with any other node. See the discussion before Lemma 1 and after Proposition 3 of [22] for comments on the search space size. We do not know of a better way to exactly solve the optimization problem above than by exhausting over all matrices that satisfy (i) and checking that they satisfy (ii). Thus, rather than pruning the search space, it seems that item (ii) adds computational complexity.

Finally, we consider the complexity of our approach, in terms of the search space size. Note that for single unicast EIC problems, applying $\phi_{(R,B)}$ has no effect (that is, $A = \phi_{(R,B)}(A)$), so the computation of the restricted min-rank $\text{r-minrk}_2(G, \mathcal{A}_{(R,B)})$ in step 2 of Algorithm 1 is equivalent to computing the (non-restricted) minrank of G, $\mathrm{minrk}_2(G)$. In this single unicast case, our method solves for $\mathrm{minrk}_2(G)$

[5]A *weak* sender is one whose available information is a subset of the information available to another sender.

which can be expressed as:

$$\begin{aligned} &\text{minimize} \quad \mathrm{rk}_2(A) \\ &\text{subject to} \quad A \text{ fits } G \end{aligned} \tag{4}$$

It is not hard to see that the search space of (4), solved in step 2 of Algorithm 1, has a search space of size (in number of matrices)

$$2^{\sum_{k=1}^{n} n_k}. \tag{5}$$

2) Comparison of the Size of Search Spaces: We can compare the search space size of Algorithm 1 to that of [18] analytically. Let $\bar{d} = \frac{1}{m}\sum_{k=1}^{n} n_k$ be the average number of times that each block is replicated in the system. Then the state space of Algorithm 1 has size (as in (5)) given by

$$2^{m\bar{d}},$$

while the state space in [18] has size (as in (2)) given by

$$2^{m\bar{d}\left(n(1-1/\bar{d})+1\right)}.$$

Thus, our approach has complexity a power of $n(1-1/\bar{d})$ smaller than that of [18].

It is difficult to compare the size (3) of the search space in [22] to the size (5) of our search space analytically, so we do this empirically. Figure 7 shows how the base-2 logarithm of the search space of Algorithm 1 compares to that of the Combined LT-CMAR procedure of [22]. The larger the ratio, the more costly the Combined LT-CMAR algorithm is relative to our EIC heuristic. Values are computed on Erdős-Renyi graphs randomly generated with various values of n, the number of vertices, and p, the probability of each directed edge existing. Note that graphs are re-sampled for each trial until one is generated such that every node has an out-degree of at least one. This is done because a node without an out-edge cannot satisfy any requirements with messages from the other nodes, so it has to be dropped from the problem, reducing n. Except for the smallest values of n and p, S_{EIC} is smaller than $S_{LT-CMAR}$, meaning that Algorithm 1 has a smaller search space than the combined LT-CMAR algorithm.

As shown in Figure 7b, the ratios go down in some cases as the edge probability approaches 1, because denser graphs create more similarities in the neighborhoods of nodes for the Combined LT-CMAR procedure to leverage into search space reduction. However, as shown in Figure 7a, for fixed edge probabilities not close to 1, the quantity $S_{LT-CMAR}$ (that is, the logarithm of the search space size for Combined LT-CMAR) grows relative to S_{EIC} (the logarithm of the search space size for Algorithm 1) as n grows. The fact that the plots in Figure 7a seem to be growing linearly suggests that

$$\begin{aligned} &|\text{Search space for Combined LT-CMAR}| = \\ &|\text{Search space for Algorithm 1}|^{\Theta(n)}, \end{aligned}$$

matching the analytical conclusion we obtained comparing Algorithm 1 to the work of [18] above.

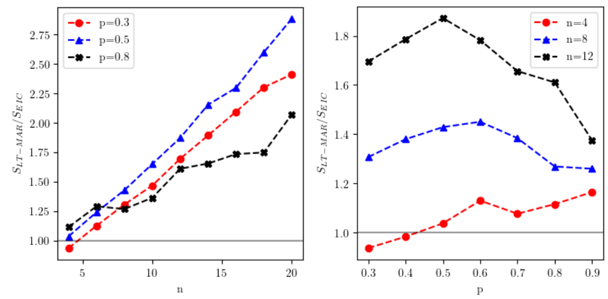


Fig. 7. Comparison of the search space size of our method vs. that of [22]. Both (a) and (b) plot $S_{LT-CMAR}/S_{EIC}$. Here, $S_{LT-CMAR}$ is the logarithm, base 2, of the search space size for a given (R, B) using the method of [22]. S_{EIC} is the logarithm, base 2, of the search space size using our method of approximating with the corresponding centralized solution. Ratios $S_{LT-CMAR}/S_{EIC}$ are plotted for the averages over sets of 20 Erdős-Renyi graphs with the given number of vertices n and probability p for each directed edge, as n varies (a) or p varies (b). When $S_{LT-CMAR}/S_{EIC} > 1$ our algorithm has a strictly smaller search space.

B. Approximating (T)

Computing a task-based solution consists of two main steps: finding a neighborhood partition (Definition 10) and finding an index coding solution to the task defined by each $\tilde{N}_i$ for sender node i. Our heuristic uses Lemma 4 to approximate an optimal choice for a neighborhood partition. In order to see how, we define the *neighborhood-cliques* associated with an EIC problem:

Definition 17: Given a problem graph G for some EIC problem (R, B) with sender node neighborhoods $N_1, ..., N_n$, let the set of *neighborhood-cliques* be $\mathscr{C} := \{V(C) : C$ is a maximal clique in $G|_{N_i}$ for some $N_i\}$

We first define the min-cover and min-clique-cover problems. Let $U = \{u_1, ..., u_n\}$ be a set of n elements. Let $S = \{X_1, ..., X_m\}$ be subsets of U, i.e. $X_j \subseteq U$ for each $j \in [m]$, such that for all $i \in [n]$, u_i is in some X_j. The min-cover problem over (U, S) is to find the smallest $\{X_{j_1}, ..., X_{j_k}\} \subseteq S$, such that $\bigcup_{\ell=1}^{k} X_{j_\ell} = U$. The min-clique-cover problem over a graph G is an instance (U, S) of min-cover in which $U = V(G)$ and S is the set of all cliques in G, including non-maximal cliques.

Using neighborhood-cliques, solving for the chromatic numbers used to upper bound the length of a task-based solution reduces to min-cover:

Theorem 5: Given an EIC problem (R, B) and the corresponding problem graph G, solving for the neighborhood partition $\tilde{N}_1, ...\tilde{N}_n$ to minimize

$$\sum_{i=1}^{n} \chi\left(\overline{(G|_{\tilde{N}_i})_{both}}\right) \tag{6}$$

is equivalent to the min-cover problem over vertices of G with sets

$$\mathscr{C} = \{V(C_i) : C_i \text{ is a maximal clique in } (G|_{N_i})_{both}\}.$$

In particular, given a solution to the min-cover problem for $\mathcal{C}$, we can, in polynomial time, find a neighborhood partition to minimize (6).

Theorem 5 will follow from the following two lemmas.

Lemma 6: x Given a neighborhood partition $\{\tilde{N}_i : i \in [n]\}$ for some (R, B) with problem graph G, there exists a cover $\mathcal{C}$ of $V(G)$ chosen from elements of $\mathscr{C}$ (the set of neighborhood-cliques) such that

$$\sum_{C_j \in \mathcal{C}} \chi\left(\overline{C_j}\right) = \sum_{i=1}^{n} \chi\left(\overline{(G|_{\tilde{N}_i})_{both}}\right).$$

Proof: Take some $\tilde{N}_i$ and consider a minimum coloring using $\chi(\overline{(G|_{\tilde{N}_i})_{both}})$ colors. Each set of vertices with a shared color is by definition a clique in $(G|_{\tilde{N}_i})_{both}$, call such a clique C_j. Since $\tilde{N}_i \subseteq N_i$, C_j (or a larger clique containing C_j if C_j is not maximal) is in $\mathscr{C}$. We can apply this to all $\tilde{N}_i$ to get a cover of $V(G)$. Since we create such a C_j for each color used for $\tilde{N}_i$, and the complement of a clique is 1-colorable,

$$\sum_{j: C_j \text{used for } \tilde{N}_i} \chi(\overline{C_j}) = |\{j : \text{used for } \tilde{N}_i\}| = \chi(\overline{(G|_{\tilde{N}_i})_{both}}).$$

□

Lemma 7: Given an EIC problem (R, B) with problem graph G and clique cover $\mathcal{C} := \{C_j\} \subseteq \mathscr{C}$, there is a corresponding choice of neighborhood partition $\tilde{N}_1, \ldots, \tilde{N}_n$ such that

$$\sum \chi\left(\overline{C_j}\right) \geq \sum \chi\left(\overline{(G|_{\tilde{N}_i})_{both}}\right).$$

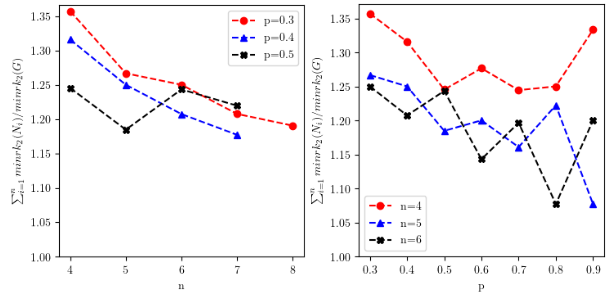
Proof: For each i, let $\tilde{N}_i := \bigcup\{C_j \in \mathcal{C} : C_j \subseteq N_i\}$. Now letting $c := |\{C_j \in \mathcal{C} : C_j \subseteq N_i\}|$, we can color $\overline{(G|_{\tilde{N}_i})_{both}}$ with c colors, so $\chi(\overline{(G|_{\tilde{N}_i})_{both}}) \leq c = \sum \chi(\overline{C_j})$. □

Theorem 5 follows immediately from Lemmas 6 and 7, as well as the observation that in the proof of Lemma 7 we can efficiently find the optimal $\tilde{N}_i$ as unions of the sets $C_j \in \mathcal{C}$.

This inspires an algorithm to find $\tilde{N}_1, \ldots, \tilde{N}_n$. Since min-cover is NP-hard solving for these will be as well, but we can use existing min-cover approximation algorithms. Below, Algorithm 2 computes the neighborhood partition which minimizes $\sum_{i=1}^{n} \chi(\overline{G}|_{\tilde{N}_i})$ and the length of the minimum task-based solution using that partition.

Algorithm 2: Given an EIC problem (R, B):

1) Construct the problem graph G
2) Let $\mathscr{C} = \emptyset$
3) For each $v_i \in V(G)$
    a) Let $N_i \subset V(G)$ be the out-neighborhood of v_i
    b) Compute $G|_{N_i}$, the subgraph induced by N_i
    c) Compute the set of maximal cliques in $(G|_{N_i})_{both}$ and add each to $\mathscr{C}$
4) Compute minimum clique cover $\mathcal{C}$ of $V(G)$ using elements of $\mathscr{C}$.
5) While there exist $C_i, C_j \in \mathcal{C}'$ such that $C_i \cap C_j \neq \emptyset$
    a) Replace C_i in $\mathcal{C}$ with $C_i \setminus (C_i \cap C_j)$
6) Let $T = 0$
7) For each $v_i \in V(G)$
    a) Let $\tilde{N}_i = \bigcup\{C_j \in \mathcal{C} : C_j \subseteq N_i\}$
    b) Compute $G|_{\tilde{N}_i}$, the subgraph induced by $\tilde{N}_i$
        i) Let $(R^{(i)}, B^{(i)})$ be a EIC problem with problem graph $G|_{\tilde{N}_i}$, keeping vertex labels from (R, B)
    c) $T = T + \text{r-minrk}_2(G|_{\tilde{N}_i}, \mathcal{A}_{(R^{(i)}, B^{(i)})})$
8) T is the total cost of the optimal task-based solution given neighborhood partition $\{\tilde{N}_1, \ldots, \tilde{N}_n\}$.



(a) (b)

Fig. 8. Ratio of the length of our task-based solution returned by Algorithm 2 to the length of the optimal centralized solution.[7]

Remark 8: Note that after step 5 of Algorithm 2, $\mathcal{C}$ is a clique partition of $V(G)$ of minimum size. This is because removing vertices from a clique in $C_i \in \mathcal{C}$ produces a new clique, and if a smaller clique partition existed then it would also be a smaller clique cover, contradicting choice of $\mathcal{C}$ in step 4 as a minimum clique cover.

Figure 8 shows the ratio of the length of our approximately optimal task-based solution compared to the length of the optimal centralized solution. This ratio upper bounds the ratio of a true optimal task-based solution to the corresponding centralized solution. In all of our experiments this approximation ratio is upper-bounded by 1.4. As in the experiments in Figure 7, Erdős-Renyi graphs are randomly generated for a variety of values for n, the number of nodes, and p, the directed edge probability. As the size of the graph increases for a fixed edge probability, the ratio appears to converge. For a fixed number of nodes, there also appears to be some upper bound on the ratio even as the probability of each edge goes to 1.

Remark 9: It has been shown [14] that with high probability, Erdős-Renyi graphs G with n nodes have $\text{minrk}(G) = \theta(n/\log n)$ for any constant edge probability p. As a result, in the experiments in Figure 8 we expect the optimal task-based solution length to be within a constant factor of the length of optimal centralized (and decentralized) solutions.

VI. Conclusion

In this paper we defined embedded index coding, a special case of multi-sender index coding in which each node of the network is both a broadcast sender and a receiver. We characterized an EIC problem using a problem graph, and we used this formulation to show that the optimal length of a solution to an EIC problem is bounded by twice the

[7] Sample sizes in these experiments are 10 random graphs, except $p = 0.9, n = 6$ which only uses 5, since the search space for the brute-force minrank algorithm explodes, increasing exponentially in the number of graph edges.

length of the optimal centralized index coding solution. We also defined *task-based* solutions to EIC problems, in which the set messages broadcast by each node can be decoded independently of messages from other senders, and we proved characterizations and bounds for task-based solutions. Finally, we used these bounds to develop heuristics for finding good solutions to EIC problems, and showed empirically that these heuristics perform well.

We end with some open questions and future directions. Since this work first appeared, it was shown by [17] that for any integer k, there exists an index coding problem with problem graph G and $\mathrm{minrk}_2(G) = k$, such that the task-based solution cost is $\Theta(k^2)$. Since we've shown an optimal decentralized solution has cost within a constant factor of the optimal centralized solution cost, i.e. $\mathrm{minrk}_2(G)$, this result also shows a gap between general decentralized and task-based solutions. However, the exact relationship between decentralized solutions and centralized solutions to embedded problems remains open.

It is also an interesting question to improve on algorithms for finding task-based solutions. Our current approach uses an upper bound on minrank, given by the chromatic number of the complement of the problem graph. This bound is known to be quite loose in some settings. The fractional chromatic number of the complement of the problem graph, $\chi_f(\overline{G})$, has been used to tighten the upper on minrank of G [5], and it was also shown by [30] that the optimal centralized index coding solution size with locality of one is $\chi_f(\overline{G})$. Thus the fractional chromatic number may be a useful approach in this direction.

Additionally, the Lovász Theta function has been shown to be greater than Shannon capacity (and thus greater than independence number) but less than minrank for some graphs, but greater than minrank for others [8], [13]. As suggested by [16], the Lovász Theta function may be useful for better approximating minrank, which could lead to better algorithms for centralized, decentralized, and task-based solutions.

Additionally, the Lovász Theta function has been shown to be greater than Shannon capacity (and thus greater than independence number) but less than minrank for some graphs, but greater than minrank for others [8,13]. As suggested by [14], the Lovász Theta function may be useful for better approximating minrank, which could lead to better algorithms for centralized, decentralized, and task-based solutions.

APPENDIX

Claim: No centralized solution of length less than 3 exists for the problem graph in Figure 6a.

Proof: All five requested blocks $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_5$ must be included in at least one message. Clearly the only solution candidate of length 1 is $\mathcal{D}_1 \oplus \mathcal{D}_2 \oplus \mathcal{D}_3 \oplus \mathcal{D}_4 \oplus \mathcal{D}_5$ and would not work. Next we show no solution of length 2 is possible. Suppose such a solution did exist and let A, B be the two messages. Then each node could compute $A \oplus B$ and any sum over $A, B, A \oplus B$ and its side information. We list the requirements for $A, B, A \oplus B$ imposed by three nodes, w, y, z below. At least one of the "message options" for each node must be included in $\{A, B, A \oplus B\}$ for A, B to form a solution:

node	has	wants	message options
w	$\mathcal{D}_2$	$\mathcal{D}_3$	$\{\mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_2 \oplus \mathcal{D}_3\}$
y	$\{\mathcal{D}_2, \mathcal{D}_3\}$	$\mathcal{D}_4$	$\{\mathcal{D}_4, \mathcal{D}_2 \oplus \mathcal{D}_4, \mathcal{D}_3 \oplus \mathcal{D}_4, \mathcal{D}_2 \oplus \mathcal{D}_3 \oplus \mathcal{D}_4\}$
z	$\{\mathcal{D}_1, \mathcal{D}_3\}$	$\mathcal{D}_5$	$\{\mathcal{D}_5, \mathcal{D}_3 \oplus \mathcal{D}_5, \mathcal{D}_1 \oplus \mathcal{D}_5, \mathcal{D}_1 \oplus \mathcal{D}_3 \oplus \mathcal{D}_5\}$

Without loss of generality (since we are taking sums over $\mathbb{F}_2$), if there is a possible solution we can express it as A from the set of options for w and B from the set of options for y such that $A \oplus B$ is in the set of options for z. Since $\mathcal{D}_5$ appears in all options for z and in none of the options for w, y clearly choosing such an A, B is not possible. Thus no solution of length 2 exists for these three nodes (or the graph as a whole). □

Claim: No decentralized solution of length less than 4 exists for the problem graph in Figure 6a.

Proof: We show a solution of length 3 cannot exist; thus a solution of length 2 or 1 also cannot exist because it could trivially be extended by duplicating a message.

First, consider the requested blocks, $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_5$. All five blocks are requested by at least one node (that doesn't already have it), so all five blocks must be included in a solution. Additionally, each node has one or two of the five blocks, so it can send a single data block or, if it has two data blocks, their sum. Thus any decentralized solution of length 3 consists of either two sums of two data blocks each and one additional block (so that the five distinct data blocks are all used) or three sums of two data blocks each, in which case one data block appears more than once. We list all possible distinct cases for assembling messages in this way (excluding cases where not all blocks appear at least once):

Set #	Message 1	Message 2	Message 3
1	$\mathcal{D}_1 \oplus \mathcal{D}_4$	$\mathcal{D}_3 \oplus \mathcal{D}_5$	$\mathcal{D}_2$
2	$\mathcal{D}_1 \oplus \mathcal{D}_4$	$\mathcal{D}_2 \oplus \mathcal{D}_3$	$\mathcal{D}_3 \oplus \mathcal{D}_5$
3	$\mathcal{D}_1 \oplus \mathcal{D}_4$	$\mathcal{D}_2 \oplus \mathcal{D}_3$	$\mathcal{D}_5$

For each of these three message sets, at least one node would be unable to get its requested information. From message set #1, node w cannot get $\mathcal{D}_3$ (w has $\mathcal{D}_2$ as side information). From message sets #2 and #3, node u cannot get $\mathcal{D}_2$ (u has $\mathcal{D}_1, \mathcal{D}_4$ as side information). Thus there is no solution of length less than 4. □

REFERENCES

[1] F. Arbabjolfaei, B. Bandemer, Y.-H. Kim, E. Sasoglu, and L. Wang, "On the capacity region for index coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2013, pp. 962–966.

[2] F. Arbabjolfaei and Y.-H. Kim, "Three stories on a two-sided coin: Index coding, locally recoverable distributed storage, and guessing games on graphs," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2015, pp. 843–850.

[3] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1479–1494, Mar. 2011.

[4] Y. Birk and T. Kol, "Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2825–2830, Jun. 2006.

[5] A. Blasiak, R. Kleinberg, and E. Lubetzky, "Index coding via linear programming," 2010, *arXiv:1004.1379*. [Online]. Available: http://arxiv.org/abs/1004.1379

[6] M. A. R. Chaudhry, Z. Asad, A. Sprintson, and M. Langberg, "On the complementary index coding problem," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2011, pp. 244–248.
[7] M. A. R. Chaudhry and A. Sprintson, "Efficient algorithms for index coding," in *Proc. IEEE INFOCOM-IEEE Conf. Comput. Commun. Workshops*, Apr. 2008, pp. 1–4.
[8] A. Coja-Oghlan, "The Lovász number of random graphs," *Combinatorics, Probab. Comput.*, vol. 14, no. 4, pp. 439–465, Jul. 2005.
[9] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.
[10] M. Effros, S. El Rouayheb, and M. Langberg, "An equivalence between network coding and index coding," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2478–2487, May 2015.
[11] S. El Rouayheb, A. Sprintson, and C. Georghiades, "On the relation between the index coding and the network coding problems," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2008, pp. 1823–1827.
[12] S. E. Rouayheb, A. Sprintson, and P. Sadeghi, "On coding for cooperative data exchange," in *Proc. IEEE Inf. Theory Workshop (ITW, Cairo)*, Jan. 2010, pp. 1–5.
[13] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4388–4413, Jul. 2017.
[14] A. Golovnev, O. Regev, and O. Weinstein, "The minrank of random graphs," *IEEE Trans. Inf. Theory*, vol. 64, no. 11, pp. 6990–6995, Nov. 2018.
[15] W. Haemers, "On some problems of Lovász concerning the Shannon capacity of a graph," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 2, pp. 231–232, Mar. 1979.
[16] I. Haviv, "On minrank and the Lovász theta function," in *Proc. Approx., Randomization, Combinat. Optim. Algorithms Techn. (APPROX/RANDOM)*, vol. 44, no. 42. Wadern, Germany: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018, p. 22.
[17] I. Haviv, "Task-based solutions to embedded index coding," *IEEE Trans. Inf. Theory*, vol. 66, no. 10, pp. 6144–6149, Oct. 2020.
[18] J.-W. Kim and J.-S. No, "Linear index coding with multiple senders and extension to a cellular network," *IEEE Trans. Commun.*, vol. 67, no. 12, pp. 8666–8677, Dec. 2019.
[19] A. Le, A. S. Tehrani, A. G. Dimakis, and A. Markopoulou, "Instantly decodable network codes for real-time applications," in *Proc. Int. Symp. Netw. Coding (NetCod)*, Jun. 2013, pp. 1–6.
[20] M. Li, L. Ong, and S. J. Johnson, "Improved bounds for multi-sender index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 3060–3064.
[21] M. Li, L. Ong, and S. J. Johnson, "Cooperative multi-sender index coding," *IEEE Trans. Inf. Theory*, vol. 65, no. 3, pp. 1725–1739, Mar. 2019.
[22] M. Li, L. Ong, and S. J. Johnson, "Multi-sender index coding for collaborative broadcasting: A rank-minimization approach," *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1452–1466, Feb. 2019.
[23] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.
[24] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Fundamental tradeoff between computation and communication in distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Dec. 2016, pp. 1814–1818.
[25] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "A unified coding framework for distributed computing with straggling servers," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2016, pp. 1–6.
[26] E. Lubetzky and U. Stav, "Nonlinear index coding outperforming the linear optimum," *IEEE Trans. Inf. Theory*, vol. 55, no. 8, pp. 3544–3551, Aug. 2009.
[27] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.
[28] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015.
[29] A. Mazumdar, "On a duality between recoverable distributed storage and index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun./Jul. 2014, pp. 1977–1981.
[30] L. Natarajan, P. Krishnan, V. Lalitha, and H. Dau, "Locally decodable index codes," *IEEE Trans. Inf. Theory*, vol. 66, no. 12, pp. 7387–7407, Dec. 2020.
[31] M. J. Neely, A. S. Tehrani, and Z. Zhang, "Dynamic index coding for wireless broadcast networks," *IEEE Trans. Inf. Theory*, vol. 59, no. 11, pp. 7525–7540, Nov. 2013.
[32] L. Ong, C. K. Ho, and F. Lim, "The single-uniprior index-coding problem: The single-sender case and the multi-sender extension," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3165–3182, Jun. 2016.
[33] P. Sadeghi, F. Arbabjolfaei, and Y.-H. Kim, "Distributed index coding," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Sep. 2016, pp. 330–334.
[34] J. Seppala, T. Koskela, T. Chen, and S. Hakola, "Network controlled device-to-device (D2D) and cluster multicast concept for LTE and LTE—A networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Mar. 2011, pp. 986–991.
[35] K. Shanmugam and A. G. Dimakis, "Bounding multiple unicasts through index coding and locally repairable codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2014, pp. 296–300.
[36] K. Shanmugam, A. G. Dimakis, and M. Langberg, "Local graph coloring and index coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2013, pp. 1152–1156.
[37] K. Shanmugam, A. G. Dimakis, and M. Langberg, "Graph theory versus minimum rank for index coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2014, pp. 291–295.
[38] L. Song, C. Fragouli, and T. Zhao, "A pliable index coding approach to data shuffling," *IEEE Trans. Inf. Theory*, vol. 66, no. 3, pp. 1333–1353, Mar. 2020.
[39] M. Tahmasbi, A. Shahrasbi, and A. Gohari, "Critical graphs in index coding," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 2, pp. 225–235, Feb. 2015.
[40] C. Thapa, L. Ong, and S. J. Johnson, "Interlinked cycles for index coding: Generalizing cycles and cliques," *IEEE Trans. Inf. Theory*, vol. 63, no. 6, pp. 3692–3711, Jun. 2017.
[41] D. B. West *et al.*, *Introduction to Graph Theory*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

Alexandra Porter received the B.S. degree in math and computer science from Arizona State University in 2017. She is currently pursuing the degree. She is also an NSF Graduate Research Fellow with the Computer Science Department, Stanford University, advised by M. Wootters and J. Leskovec. Her research interests include error correcting codes and graph algorithms.

Mary Wootters (Member, IEEE) received the B.A. degree in math and computer science from Swarthmore College, in 2008, and the Ph.D. degree in mathematics from the University of Michigan, in 2014. She was an NSF Post-Doctoral Fellow with Carnegie Mellon University from 2014 to 2016. She is currently an Assistant Professor of computer science and electrical engineering with Stanford University. She also works in theoretical computer science, applied math, and information theory. Her research interests include error correcting codes and randomized algorithms for dealing with high-dimensional data. She was a recipient of the NSF CAREER Award and was named a Sloan Research Fellow in 2019. She was named to the Stanford Tau Beta Pi Teaching honor roll from 2018 to 2019 and from 2019 to 2020.