The Multi-Sender Multicast Index Coding

Lawrence Ong
The University of Newcastle, Australia
Email: lawrence.ong@cantab.net

Fabian Lim
Massachusetts Institute of Technology, USA
Email: flim@mit.edu

Chin Keong Ho
Institute for Infocomm Research, Singapore
Email: hock@i2r.a-star.edu.sg

***Abstract*—We focus on the following instance of an index coding problem, where a set of receivers are required to decode multiple messages, whilst each knows one of the messages a priori. In particular, here we consider a generalized setting where they are multiple senders, each sender only knows a subset of messages, and all senders are required to collectively transmit the index code. For a single sender, Ong and Ho (ICC, 2012) have established the optimal index codelength, where the lower bound was obtained using a pruning algorithm. In this paper, the pruning algorithm is simplified, and used in conjunction with an appending technique to give a lower bound to the multi-sender case. An upper bound is derived based on network coding. While the two bounds do not match in general, for the special case where no two senders know any message bit in common, the bounds match, giving the optimal index codelength. The results are derived based on graph theory, and are expressed in terms of strongly connected components.**

I. Introduction

We consider a generalization of the well-known index coding problem to the multi-sender setting where the senders are *constrained* to know only certain messages. As in the typical setup we have n receivers each requiring some of the m independent messages in the set $\mathcal{M} = \{x_1, x_2, \ldots, x_m\}$, but in addition, we have S separate senders who know only subsets of $\mathcal{M}$. The problem is precisely given by $(\{\mathcal{M}_s : s \in \{1, 2, \ldots, S\}\}, \{(\mathcal{W}_r, \mathcal{K}_r) : r \in \{1, 2, \ldots, n\})$. Here, $(\mathcal{W}_r, \mathcal{K}_r)$ corresponds to receiver r, where $\mathcal{K}_r \subseteq \mathcal{M}$ is the subset of messages it knows a priori, and $\mathcal{W}_r \subseteq \mathcal{M}$ is the subset of messages it requires. Furthermore, $\mathcal{M}_s \subseteq \mathcal{M}$ denotes messages that sender s is constrained to know. Clearly $\mathcal{W}_r \cap \mathcal{K}_r = \{\}$. Without loss of generality, we assume that $\bigcup_{s=1}^{S} \mathcal{M}_s = \mathcal{M}$, meaning that each message bit is available at some sender(s).

We define a multi-sender index code for the above setup:

Definition 1 (Multi-Sender Index Code): An index code for problem instance $(\{\mathcal{M}_s\}, \{(\mathcal{W}_r, \mathcal{K}_r)\})$ consists of

1) an encoding function for each sender $s \in \{1, 2, \ldots, S\}$, $E_s : \{0,1\}^{|\mathcal{M}_s|} \mapsto \{0,1\}^{\ell_s}$ such that $\boldsymbol{c}_s = E_s(\mathcal{M}_s)$,
2) a decoding function for each receiver $r \in \{1, 2, \ldots, n\}$, $D_r : \{0,1\}^{(\sum_{s=1}^{S} \ell_s) + |\mathcal{K}_r|} \mapsto \{0,1\}^{|\mathcal{W}_r|}$ such that $\mathcal{W}_r = D_r(\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_S, \mathcal{K}_r)$.

That is, each sender encodes its $|\mathcal{M}_s|$-bit message into an ℓ_s-bit codeword. The codewords of all senders are given to all receivers. The total number of transmitted bits is thus $\tilde{\ell} = \sum_{s=1}^{S} \ell_s$, and $\tilde{\ell}$ the index codelength for the multi-sender generalization of the index coding problem. We seek the optimal (i.e., the minimum) index codelength, denoted $\tilde{\ell}^*$, and an optimal index code (i.e., an index code of length $\tilde{\ell}^*$). In this paper, we assume that each message bit and each codebit is binary, but our results hold as long as all messages and codeletters take values in the same alphabet.

The case $S = 1$ reduces to the usual single-sender index coding problem studied in many works [1]–[5]. This generalization is of interest, for example, in distributed settings where senders are constrained to know only part of the entire message due to, for instance, limited bandwidth between senders for sharing the messages or decoding errors when downloading the messages from a central processor. Clearly, a multi-sender index code for any $S > 1$ case will also be a code for the single-sender $S = 1$ case for the same $\{(\mathcal{W}_r, \mathcal{K}_r)\}$ decoding requirements. But the converse is not true. Hence, the techniques described here are new, and previous techniques for the single-sender case do not straightforwardly apply.

This paper also differentiates from other works [1]–[4] by considering a *multicast* setup. The classical setup (which is *multiprior unicast*) is that each receiver r requires only one unique message (i.e., each $\mathcal{W}_r = \{x_r\}$), knowing a set of message $\mathcal{K}_r$ a priori. Here we consider the case where each receiver r knows only one unique message (i.e., each $\mathcal{K}_r = \{x_r\}$), but requires a set of messages $\mathcal{W}_r$, which can be a large subset of $\mathcal{M}$. We call this the *uniprior multicast* problem; this setup (first looked at in [5]) is motivated by the *multi-way relay channel* [6], [7]. For the single-sender case, the uniprior multicast problem is completely solved in [5], which interestingly shows the optimal scheme to be *linear*; this contrasts with the classical setup where it is known that linear codes can be sub-optimal [8]. Here we explore the multi-sender generalization. Note that for both unicast and uniprior multicast setups, the number of receivers, n, equals the number of messages, m.

II. Graph Representations

In this paper, we introduce a graphical representation of the uniprior multicast problem $(\{(\mathcal{M}_s)\}, \{(\mathcal{W}_r, x_r)\})$ to capture both decoding requirements and sender constraints. This graphical representation shall be useful for stating and proving our subsequent results.

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ is a tuple of a vertex set $\mathcal{V}$ and an arc/edge set $\mathcal{A}$. We correspond a vertex to each of the n receivers. An arc $(i \to j)$ conveys directional information from vertex i to vertex j, while an edge (i, j) is undirected. An

(undirected) graph has only edges, while a (directed) *digraph* has only arcs; both cannot have self-loops. We represent the multi-sender uniprior multicast problem as follows.

Definition 2: The decoding requirements determined from $\{(\mathcal{W}_r, x_r)\}$ is represented by an n-vertex *information-flow* digraph[1] $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $(i \to j) \in \mathcal{A}$ if and only if (iff) receiver j requires x_i, i.e., $x_i \in \mathcal{W}_j$. The sender constraints determined from $\{\mathcal{M}_s\}$ is represented by an n-vertex *message* graph $\mathcal{U} = (\mathcal{V}, \mathcal{E})$, where $(i, j) \in \mathcal{E}$ iff messages x_i and x_j are known to the same sender, i.e., $i, j \in \mathcal{M}_s$ for some s. Note, both $\mathcal{G}$ and $\mathcal{U}$ share the *same* vertex set $\mathcal{V}$.

We denote the optimal index codelength for the problem represented by $(\mathcal{G},\mathcal{U})$ as $\tilde{\ell}^*(\mathcal{G},\mathcal{U})$.

In the sequel, we work on the *simplified model* described as follows: For any index coding problem $(\mathcal{G}_1,\mathcal{U}_1)$, we construct a *simplified* $(\mathcal{G}_2,\mathcal{U}_2)$ by removing every message x_i that is not required by any receiver (meaning that vertex i has no outgoing arc in $\mathcal{G}_1$) from the receivers and the senders; equivalently, we set $x_i = \varnothing$. There is no loss of generality because an optimal index code for $(\mathcal{G}_2,\mathcal{U}_2)$ is an *optimal* index code for $(\mathcal{G}_1,\mathcal{U}_1)$; see Appendix A for proof.

A. Terminology

We will use common graph terminology [9]: A *strongly connected component* (SCC) of a digraph is a *maximal* subgraph of the digraph such that in the subgraph, for any vertex pair i, j, there is a directed path from i to j and another from j to i. A *leaf* vertex in a digraph has no outgoing arcs. A vertex j is a *predecessor* of vertex i iff there is a directed path from j to i. A *tree* is a connected undirected subgraph with no cycle.

III. Results

The main contribution of this paper is the technique we propose to obtain a lower bound to $\tilde{\ell}^*$, which we will show to be tight in a few cases.

The lower bound and achievability in this paper will be stated in terms of *leaf SCCs* in the information digraph $\mathcal{G}$. A leaf SCC is an SCC that has

- no outgoing arc (i.e., from a vertex in the SCC to a vertex outside the SCC), and
- at least two vertices (i.e., non trivial).

A. A Lower Bound to $\tilde{\ell}^*$

The following characterization of leaf SCCs in $\mathcal{G}$ is crucial to our results/techniques. They involve the message graph $\mathcal{U}$:

1) A leaf SCC is *message connected* iff there exists a path in $\mathcal{U}$ between any two vertices in the SCC, where the path contains vertices only in the SCC.
2) A leaf SCC is *message disconnected* iff there are two vertices in the SCC with no path in $\mathcal{U}$ between them.
3) A leaf SCC which is neither message connected nor message disconnected is *semi-message connected*, referred also as semi leaf SCC. Any two vertices in the semi leaf SCC is connected by a path in $\mathcal{U}$, but there exists a vertex pair which must be connected by a path that includes vertices outside the leaf SCC.

[1]The definition of the information-flow graph defined for the uniprior multicast problem here (where the arcs capture what the receivers *require*) is different from the side-information graph defined for the unicast problem [3] (where the arcs capture what the receivers *know*).

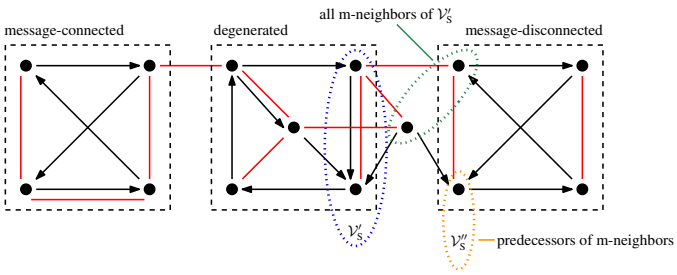

Fig. 1. Example of an index coding problem, diagrammed by super-imposing an information-flow digraph $\mathcal{G}$ (arcs in black) and a message graph $\mathcal{U}$ (edges in red). These graphs illustrate concurrently three leaf SCCs types: (i) message connected, where there is a (red) path between any two vertices through only vertices in the SCC; (ii) message disconnected, containing two vertices cannot be connected with a red path; and (iii) semi message connected, where some vertices must be connected by a path with vertices outside the SCC. Note, the semi leaf SCC here is degenerated because we can find two vertex sets $\mathcal{V}'_S$ and $\mathcal{V}''_S$, such that all m-neighbors of $\mathcal{V}'_S$ are predecessors of $\mathcal{V}''_S$.

Semi leaf SCCs are also further classified using the following property. For a vertex set $\mathcal{V}_S \subseteq \mathcal{V}$, a vertex $i \notin \mathcal{V}_S$ is an *m-neighbor* of $\mathcal{V}_S$ iff there is an edge (i, v) between i and some $v \in \mathcal{V}_S$. Then, a semi leaf SCC with vertex set $\mathcal{V}_S$, is said to be *degenerated* iff

1) $\mathcal{V}_S$ can be partitioned into *two* parts $\mathcal{V}'_S$ and $\mathcal{V}_S \setminus \mathcal{V}'_S$ such that there is no edge in $\mathcal{U}$ across vertices from different parts, and
2) there exists a vertex subset not in $\mathcal{V}_S$, denoted by $\mathcal{V}''_S \subseteq \mathcal{V} \setminus \mathcal{V}_S$, which
   - can only have at most one non-leaf vertex (the other vertices must strictly be leaf)

   such that
   - every m-neighbor of $\mathcal{V}'_S$ is in $\mathcal{V}''_S$ or is a predecessor of some vertex in $\mathcal{V}''_S$.

Figure 1 illustrates these different types of leaf SCCs.

We will see later that a lower bound is easily obtained if $\mathcal{G}$ has no leaf SCC. To this end, we propose Algorithm 1 that *breaks* all leaf SCCs, such that the leaf SCCs are no longer leaf SCCs. In the algorithm, we *prune* a leaf SCC $\mathcal{V}_S$ by arbitrarily selecting a vertex $v \in \mathcal{V}_S$ and removing all outgoing arcs *from* v; we *append* a dummy leaf vertex i to a leaf SCC $\mathcal{V}_S$ by adding i and an outgoing arc $(v \to i)$ from some $v \in \mathcal{V}_S$ to i.

Algorithm 1 has two distinct phases. In phase 1, the procedure BreakLeafSCC is run once to break all message-connected, all message-disconnected, and some degenerated leaf SCCs. Phase 2 breaks the remaining leaf SCCs. In step (iv-a), the choice of leaf SCC for adding edges is arbitrary. Nevertheless, a proper choice of leaf SCC(s) *minimizes* the number of rounds that step (iv) iterates, which we shall see will then give a better lower bound.

Let $V_{\text{out}}(\mathcal{G})$ denote the number of non-leaf vertices in $\mathcal{G}$ (i.e., each with at least one outgoing arc). Further, let N_{conn} denote the number of message-connected leaf SCCs in $\mathcal{G}$ (note that N_{conn} for each $\mathcal{G}$ is fixed, independent of the algorithm), N_{rem} denote the number of leaf SCCs that remain after the initial

Algorithm 1: Breaking all Leaf SCCs

function BREAKLEAFSCC
- **foreach** *message-connected leaf SCC* **do**
  - (i) Prune the leaf SCC;
- **foreach** *message-disconnected leaf SCC* **do**
  - (ii) Append a dummy vertex to the leaf SCC;
- **while** *there exists degenerated leaf SCC, with vertex set* $\mathcal{V}_S$, **do**
  - **if** *the set* $\mathcal{V}_S'' \subseteq \mathcal{V} \setminus \mathcal{V}_S$ *contains one non-leaf vertex, denoted by* $v'' \in \mathcal{V}_S''$, **then**
    - (iii-a) Add an arc from any vertex in $\mathcal{V}_S'$ to the vertex v'';
  - **else**
    - (iii-b) Add an arc from any vertex in $\mathcal{V}_S'$ to any vertex in $\mathcal{V}_S''$;

begin `// The algorithm starts here`
- `// Phase 1`
- Run BREAKLEAFSCC;
- `// Phase 2: Iteration & optimization`
- **while** *there exists leaf SCC (only semi SCCs left)* **do**
  - (iv-a) Select one semi leaf SCC;
  - (iv-b) Arbitrarily add edges between vertex pairs until the leaf SCC is message connected;
  - (iv-c) Run BREAKLEAFSCC;

run of BREAKLEAFSCC in phase 1 (they must be semi leaf SCCs), and $N_{\text{(iv)}}$ denote the number of iterations in phase 2. We will show (in Section IV-B) that each iteration of step (iv) always reduces the number of leaf SCCs, and so $N_{\text{(iv)}} \leq N_{\text{rem}}$. We now state the main result of this paper:

Theorem 1 (Lower bound): The optimal multi-sender index codelength is lower bounded as

$$\tilde{\ell}^* \geq V_{\text{out}}(\mathcal{G}) - (N_{\text{conn}} + N_{\text{(iv)}}). \tag{1}$$

We will prove Theorem 1 in Sec. IV. As mentioned earlier, the lower bound is optimized by finding the smallest $N_{\text{(iv)}}$.

B. Achievability

Our achievability scheme is based on the construction of special trees in the message graph $\mathcal{U}$, referred to as *connecting trees*, which has all the following properties placed on its vertex set $\mathcal{V}^{\text{T}}$:

1) Each vertex in $\mathcal{V}^{\text{T}}$ has one or more outgoing arcs in $\mathcal{G}$.
2) Each vertex in $\mathcal{V}^{\text{T}}$ has no outgoing arc in $\mathcal{G}$ to $\mathcal{V} \setminus \mathcal{V}^{\text{T}}$,
3) No vertex in $\mathcal{V}^{\text{T}}$ belongs to any message-connected leaf SCCs or another connecting tree.

Let N_{tree} denote the number of connecting trees that can be found. We will propose a coding scheme that achieves the following index codelength:

Theorem 2 (Achievability): The optimal multi-sender index codelength is upper bounded as

$$\tilde{\ell}^* \leq V_{\text{out}}(\mathcal{G}) - (N_{\text{conn}} + N_{\text{tree}}). \tag{2}$$

We prove Theorem 2 in Sec. V. The achievability is optimized by finding the maximum number of connecting trees.

C. Special Cases

Combining Theorems 1 and 2, we conclude $N_{\text{(iv)}} \geq N_{\text{tree}}$, and thus the optimal index codelength is found within $N_{\text{(iv)}} - N_{\text{tree}}$ bits. In the following special cases, we have $N_{\text{(iv)}} = N_{\text{tree}}$, and the lower bound is tight.

Corollary 1: If no leaf SCC remains after running phase 1 of Algorithm 1, then

$$\tilde{\ell}^* = V_{\text{out}}(\mathcal{G}) - N_{\text{conn}}. \tag{3}$$

Proof: Since $N_{\text{rem}} \geq N_{\text{(iv)}} \geq N_{\text{tree}}$, $N_{\text{rem}} = 0$ implies that $N_{\text{(iv)}} = N_{\text{tree}} = 0$. ■

Corollary 2: If each bit x_i in the message set $\mathcal{M}$ is known to only one sender (i.e., the n sender constraint sets $\mathcal{M}_s$ partition $\mathcal{M}$), then the optimal index codelength is given by (3).

Proof: If messages x_i and x_j belong to some sender s (i.e. $x_i, x_j \in \mathcal{M}_s$), then there exists an edge (i, j) in the message graph $\mathcal{U}$. Otherwise, if the messages x_i, x_j belong to different senders, it is impossible to have a *path* between i and j. This means we have only message connected or disconnected leaf SCCs, i.e., there is no semi leaf SCC. Thus, $N_{\text{rem}} = 0$. ■

Corollary 2 includes the result of the single-sender problem [5] as a special case. For more examples, we refer the reader to the extended version of this paper [10].

IV. PROOF OF THEOREM 1 (LOWER BOUND)

We will refer to each vertex i as receiver i (and vice versa), and x_i as the message of receiver/vertex i. The following lemmas will be useful for deriving the lower bound:

Lemma 1: From any index code, each receiver i must be able to decode the messages of all its predecessors.

Proof: From every arc $(j \to i)$, receiver i must be able to decode x_j. Having decoded x_j, receiver i knows the only a priori message that receiver j has. Therefore, receiver i must also be able to decode all messages required (and hence decodable) by receiver j, i.e., $\mathcal{W}_j = \{x_k : (k \to j) \in \mathcal{A}\}$. Further chaining of this argument shows that receiver i must be able to decode messages x_j of all predecessors j of i. ■

Lemma 2: For any index code, any receiver must be able to decode the messages of all predecessors of any leaf vertex.

Proof: Recall that if i is a leaf vertex, then $x_i = \varnothing$, i.e., it has no prior message. Consequently any receiver p, regardless of the prior knowledge $\mathcal{K}_p$ it possesses, is as good as the leaf vertex (receiver) i. Hence p must be able to decode all messages decodable by receiver i, and the result follows from the proof of Lemma 1. ■

A. Sketch Proof of Theorem 1

The key to our lower bound is Lemma 2. Given $(\mathcal{G}, \mathcal{U})$, the idea is to modify $\mathcal{G}$ to form a *grounded* digraph $\mathcal{G}^\dagger$—with the property that every vertex is *grounded*, i.e., every vertex is either a leaf vertex or a predecessor of some leaf vertex. Invoking Lemma 2, any receiver (even those with no prior, i.e., $\mathcal{K}_j = \{\}$) must be able to decode $V_{\text{out}}(\mathcal{G}^\dagger)$ bits (the messages of all non-leaf vertices in $\mathcal{G}^\dagger$). Hence, any index code—for the modified problem—must contain at least $V_{\text{out}}(\mathcal{G}^\dagger)$ bits, i.e.,

$$\tilde{\ell}^*(\mathcal{G}^\dagger, \mathcal{U}^\dagger) \geq V_{\text{out}}(\mathcal{G}^\dagger), \tag{4}$$

$\mathcal{U}^\dagger$ is the message graph appropriately modified from $\mathcal{U}$.

From a given $\mathcal{G}$ there will be many possible $\mathcal{G}^\dagger$; the main difficulty is to find one that gives the tightest lower bound. Also, care must be taken to enforce that

$$\tilde{\ell}^*(\mathcal{G},\mathcal{U}) \geq \tilde{\ell}^*(\mathcal{G}^\dagger,\mathcal{U}^\dagger). \tag{5}$$

In Section IV-B, we will prove that Algorithm 1 produces a grounded $\mathcal{G}^\dagger$, meaning that (4) holds; in Section IV-C, we will prove that $\mathcal{G}^\dagger$ produced by Algorithm 1 satisfies (5). This gives

$$\tilde{\ell}^*(\mathcal{G},\mathcal{U}) \geq V_{\text{out}}(\mathcal{G}^\dagger). \tag{6}$$

B. Algorithm 1 Produces a Grounded Digraph

If a digraph contains no leaf SCC, then it is grounded; see Appendix B for proof. We now show that Algorithm 1 produces a digraph with no leaf SCC, and hence grounded. As the algorithm terminates after all leaf SCCs have been broken, it suffices to show that the algorithm always terminates, i.e., step (iv) iterates for a finite number of times. This is true if step (iv) always reduces the number of leaf SCCs.

We first show that any of the steps (i), (ii), and (iii-b) reduces the number of leaf SCCs by one. In step (i), after removing all outgoing arcs from some vertex v in a leaf SCC, v and other vertices in the SCC (each having a directed path to v) are grounded. In steps (ii) and (iii-b), an arc is added from a leaf SCC to a leaf vertex. This will also ground all vertices in the SCC. As any grounded vertex cannot belong to a leaf SCC, each of these steps breaks the leaf SCC it "operates" on, thereby reducing the number of leaf SCCs by one.

We now show that step (iii-a) cannot increase the number of leaf SCCs. Step (iii-a) adds an arc from a leaf SCC (denote by $\mathcal{G}_{\text{leaf}}$) to some vertex v'' not in $\mathcal{G}_{\text{leaf}}$. There are three possibilities: (1) v'' is grounded. Using the argument for step (ii), the number of leaf SCCs decreases by one. (2) v'' is not grounded and has no directed path to $\mathcal{G}_{\text{leaf}}$. In this case $\mathcal{G}_{\text{leaf}}$ is made non-leaf, and the number of leaf SCCs decreases by one. (3) v'' is not grounded and has a directed path to $\mathcal{G}_{\text{leaf}}$. In this case the SCC $\mathcal{G}_{\text{leaf}}$ expands to include more vertices (including v'') and arcs. The number of leaf SCCs decreases (if the expanded SCC is non-leaf) or stays the same (otherwise).

Finally, consider each iteration (iv). Step (iv-b) makes a semi leaf SCC message connected. This step, only adding edges, does not change the number of leaf SCCs. When running BREAKLEAFSCC in step (iv-c), the leaf SCC that has been made message-connected will be broken in step (i), and other steps (ii) and (iii) cannot increase the number of leaf SCCs. So, step (iv) always reduces the number of leaf SCCs.

C. Algorithm 1 Cannot Increase the Optimal Codelength

Now, we prove (5) by showing that each of the steps (i)–(iv) cannot increase the optimal index codelength, i.e.,

$$\tilde{\ell}^*(\mathcal{G}',\mathcal{U}') \leq \tilde{\ell}^*(\mathcal{G},\mathcal{U}), \tag{7}$$

where $\mathcal{G} = (\mathcal{V},\mathcal{A})$ and $\mathcal{U} = (\mathcal{V},\mathcal{E})$ respectively denote information-flow and message graphs before each of the steps (i)–(iv), $\mathcal{G}' = (\mathcal{V}',\mathcal{A}')$ and $\mathcal{U}' = (\mathcal{V}',\mathcal{E}')$, after the step.

1) Step (i): Removing arcs in the information-flow digraph is equivalent to removing decoding requirements for the receivers. Hence, we have (7).

2) Step (ii): As adding an arc $(v \to i)$ and a dummy receiver i increases decoding requirements, we have $\tilde{\ell}^*(\mathcal{G}',\mathcal{U}') \geq \tilde{\ell}^*(\mathcal{G},\mathcal{U})$. But, Lemma 3 below says that using any optimal index code for $(\mathcal{G},\mathcal{U})$—of length $\tilde{\ell}^*(\mathcal{G},\mathcal{U})$—a dummy receiver can decode all messages of all message-disconnected leaf SCCs. Hence, this index code also satisfies the decoding requirements of $(\mathcal{G}',\mathcal{U}')$, i.e., $\tilde{\ell}^*(\mathcal{G},\mathcal{U})$ is achievable for $(\mathcal{G}',\mathcal{U}')$. So, (7) in fact holds with equality.

Lemma 3: For any index code, any receiver is able to decode the messages of all message-disconnected leaf SCCs.

Proof: Let $\mathcal{V}_{\text{S}}$ be the vertex set of a message-disconnected leaf SCC. By definition, we can partition all vertices $\mathcal{V}$ into two non-empty sets $\mathcal{V}_1$ and $\mathcal{V}_2$, such that two vertices $a, b \in \mathcal{V}_{\text{S}}$ in the leaf SCC lie on separate partitions and cannot have an undirected path (in $\mathcal{U}$) between them. Let $a \in \mathcal{V}_1$ and $b \in \mathcal{V}_2$.

The lack of edge-connectivity between $\mathcal{V}_1$ and $\mathcal{V}_2$ implies that any index code can be partitioned into two parts, $\boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$, such that every bit in $\boldsymbol{c}_1$ depends on only $\{x_i : i \in \mathcal{V}_1\}$ and not on $\{x_j : j \in \mathcal{V}_2\}$, and vice versa.

Since a and b belong to an SCC, receiver a must decode x_b (see Lemma 1). Note that $\boldsymbol{c}_1$ (which contains x_a) and $\boldsymbol{c}_2$ do not have any message bit in common. So, knowing x_a can help receiver a decode only messages in $\boldsymbol{c}_1$. Also, receiver a must decode x_b using solely $\boldsymbol{c}_2$, without using its prior. Hence, if a can decode x_b, so can any receiver—even one without prior. Since the choice of a, b was arbitrary, we have Lemma 3. ■

3) Step (iii): Recall the vertex subsets $\mathcal{V}'_{\text{S}} \subset \mathcal{V}_{\text{S}}$ and $\mathcal{V}''_{\text{S}} \subseteq \mathcal{V} \setminus \mathcal{V}_{\text{S}}$. In step (iii), we append an arc from $\mathcal{V}'_{\text{S}}$ to some vertex $v'' \in \mathcal{V}''_{\text{S}}$. If we can show that receiver v'' can decode all messages of $\mathcal{V}'_{\text{S}}$ (using any index code for $(\mathcal{G},\mathcal{U})$), then by the arguments for step (ii), we conclude (7) holds with an equality.

By Lemmas 1 and 2, receiver v'' can decode the messages of all m-neighbors of $\mathcal{V}'_{\text{S}}$, denoted as $\mathcal{N}(\mathcal{V}'_{\text{S}})$, as each vertex in $\mathcal{N}(\mathcal{V}'_{\text{S}})$ is either grounded, a predecessor of v'', or v'' itself.

Let $a \in \mathcal{V}_{\text{S}} \setminus \mathcal{V}'_{\text{S}}$ and $b \in \mathcal{V}'_{\text{S}}$, where a must decode x_b. We will show that if a can decode x_b, so can v''. Consequently, v'' can decode all messages of $\mathcal{V}'_{\text{S}}$. There is no edge across $\mathcal{V}'_{\text{S}}$ and $\mathcal{V}_{\text{S}} \setminus \mathcal{V}'_{\text{S}}$, meaning that any index codebit cannot be a function of messages from both the sets. So, we can partition any index code for $(\mathcal{G},\mathcal{U})$ into $\boldsymbol{c} = (\boldsymbol{c}_1, \boldsymbol{c}_2)$, where $\boldsymbol{c}_1$ does not contain any message of $\mathcal{V}'_{\text{S}}$, and $\boldsymbol{c}_2$ contains only the messages of $\mathcal{V}'_{\text{S}}$ and $\mathcal{N}(\mathcal{V}'_{\text{S}})$. Any advantage in decoding that a has over v'' is due to knowing x_a, but x_a can help a only in decoding the messages in $\boldsymbol{c}_1$, which a can then use to decode the messages in $\boldsymbol{c}_2$ (which contain x_b). As v'' is able decode the messages of $\mathcal{N}(\mathcal{V}'_{\text{S}})$, which contains all the overlap of messages in $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$, v'' is as capable as a in decoding x_b.

4) Steps (iv-a)–(iv-b): Appending an edge (i, j) is equivalent to appending messages $\{x_i, x_j\}$ to some sender's message set $\mathcal{M}_s$. Doing so relaxes the sender constraints, and thus the optimal codelength can only decrease. Hence (7) holds.

D. Evaluating the Lower Bound

In each iteration (iv), step (i) is run *once* through BREAKLEAFSCC, as step (iv-b) has made only *one* leaf SCC message connected. Out of steps (i)–(iii), (iv-a), and (iv-b), only step (i) changes $V_{\text{out}}(\cdot)$—reducing it by one. So when Algorithm 1 terminates, we have

$$V_{\text{out}}(\mathcal{G}^\dagger) = V_{\text{out}}(\mathcal{G}) - N_{\text{(i)}} = V_{\text{out}}(\mathcal{G}) - (N_{\text{conn}} + N_{\text{(iv)}}), \quad (8)$$

where $N_{(a)}$ is the number of times step (a) is run. Combining (6) and (8), we have Theorem 1. ∎

V. PROOF OF THEOREM 2 (ACHIEVABILITY)

We now show that there exists an index code of length $\tilde{\ell} = V_{\text{out}}(\mathcal{G}) - (N_{\text{conn}} + N_{\text{tree}})$. Let a set of connecting trees be $\{\mathcal{T}_t = (\mathcal{V}_t^{\text{T}}, \mathcal{E}_t^{\text{T}}) : t \in \{1, \ldots, N_{\text{tree}}\}\}$, and all the message-connected leaf SCCs in $\mathcal{G}$ be $\{\mathcal{C}_c = (\mathcal{V}_c^{\text{C}}, \mathcal{A}_c^{\text{C}}) : c \in \{1, 2, \ldots, N_{\text{conn}}\}\}$. Further, let the remaining vertices in $\mathcal{G}$ be $\mathcal{V}' = \mathcal{V} \setminus \{\bigcup_{t=1}^{N_{\text{tree}}} \mathcal{V}_t^{\text{T}} \cup \bigcup_{c=1}^{N_{\text{conn}}} \mathcal{V}_c^{\text{C}}\}$. Denote by $\mathcal{V}'_{\text{out}}$ the set of all non-leaf vertices in $\mathcal{V}'$. By definition, all $\mathcal{V}_t^{\text{T}}, \mathcal{V}_c^{\text{C}}$, and $\mathcal{V}'$ are disjoint.

Our coding scheme is as follows:

1) For each connecting tree $(\mathcal{V}_t^{\text{T}}, \mathcal{E}_t^{\text{T}})$, we transmit all $\{x_i \oplus x_j : (i,j) \in \mathcal{E}_t^{\text{T}}\}$, i.e., we transmit the network-coded bits of the associated message pair for each edge. Note that we transmit $|\mathcal{V}_t^{\text{T}}| - 1$ bits.
2) For each message-connected leaf SCC $(\mathcal{V}_c^{\text{C}}, \mathcal{A}_c^{\text{C}})$ (which is edge-connected by definition), we first obtain a spanning tree, denoted by $\mathcal{T}_c^{\text{ST}} = (\mathcal{V}_c^{\text{C}}, \mathcal{E}_c^{\text{ST}})$, where $\mathcal{E}_c^{\text{ST}} \subseteq \mathcal{E}$. We then transmit all $\{x_i \oplus x_j : (i,j) \in \mathcal{E}_c^{\text{ST}}\}$. Note that we transmit $|\mathcal{V}_c^{\text{C}}| - 1$ bits.
3) For the rest of the non-leaf vertices, we transmit $\{x_i : i \in \mathcal{V}'_{\text{out}}\}$, i.e., we transmit the message bits uncoded.

Each vertex in the connecting trees and the message-connected SCCs has at least one outgoing arc. Hence, the coding scheme generates an index code of length $V_{\text{out}}(\mathcal{G}) - (N_{\text{conn}} + N_{\text{tree}})$.

We can easily verify that the index code can be transmitted, as each message pair to be XORed is associated with an edge, i.e., both the message bits belong to some sender.

Finally, we show that each receiver is able to obtain its required messages. Recall that each receiver i needs to decode all messages in $\{x_j : (j \to i) \in \mathcal{A}\}$. Now, each receiver i must belong to one—and only one—of the following groups:

1) (Connecting tree) $i \in \mathcal{V}_t^{\text{T}}$: Knowing x_i, receiver i can decode all $\{x_j : j \in \mathcal{V}_t^{\text{T}}\}$ from $\{x_j \oplus x_k : (j,k) \in \mathcal{E}_t^{\text{T}}\}$ by traversing the tree (which is connected by definition). It can also decode the messages $\{x_k : k \in \mathcal{V}'_{\text{out}}\}$, sent uncoded. Since all connecting trees and message-connected leaf SCCs have no outgoing arcs, each incoming arc to i must be from either $\mathcal{V}_t^{\text{T}} \setminus \{i\}$ or $\mathcal{V}'_{\text{out}}$. So, receiver i is able to decode all its required messages.
2) (Message-connected leaf SCC) $i \in \mathcal{V}_c^{\text{C}}$: Using the same argument as that for the connecting trees, we can show that receiver i can decode all its required messages.
3) (The remaining vertices) $i \in \mathcal{V}'$: Using the argument in point 1, all incoming arcs to vertex i must come from $\mathcal{V}'_{\text{out}} \setminus \{i\}$. Since we sent $\{x_j : j \in \mathcal{V}'_{\text{out}}\}$ uncoded, receiver i can decode all its required messages. ∎

APPENDIX A

Consider an index coding problem IC1, and denote its optimal codelength by $\tilde{\ell}^*(\mathsf{IC1})$. Let $\mathcal{L}$ be the set of receivers whose messages $\{x_i : i \in \mathcal{L}\}$ are not required by any other receiver. Consider the simplified problem IC2 where $x_i = \varnothing$, for all $i \in \mathcal{L}$. Denote its optimal codelength by $\tilde{\ell}^*(\mathsf{IC2})$.

Proposition 1: Any index code for IC2 is an index code for IC1.

Proof: Any index code $\boldsymbol{c}$ for IC2 can also be transmitted by the senders of IC1. Since $\boldsymbol{c}$ satisfies the decoding requirement of IC2, it must also satisfy the those of IC1. ∎

Proposition 2: $\tilde{\ell}^*(\mathsf{IC1}) = \tilde{\ell}^*(\mathsf{IC2})$

Proof: It follows from Proposition 1 that $\tilde{\ell}^*(\mathsf{IC1}) \leq \tilde{\ell}^*(\mathsf{IC2})$. Now consider any optimal index code $\boldsymbol{c}^*$ for IC1. By definition it can contain $\{x_i : i \in \mathcal{L}\}$. Since $\boldsymbol{c}^*$ is an index code, all receivers in IC1 can decode their required messages when $x_i = 1$ for all $i \in \mathcal{L}$ (these messages x_i are not required by any receiver). So, $\boldsymbol{c}^*$ with all x_i set to 1 is also an optimal index code. Denote this code by $\boldsymbol{c}'$. Now, since $\boldsymbol{c}'$ does not depend on the actual contents of $\{x_i : i \in \mathcal{L}\}$, $\boldsymbol{c}'$ can also be sent by the senders in IC2, and it also satisfies the decoding requirement of IC2. This means $\boldsymbol{c}'$ is an index code for IC2, and hence $\tilde{\ell}^*(\mathsf{IC2}) \leq \tilde{\ell}^*(\mathsf{IC1})$. ∎

APPENDIX B

Given any digraph $\mathcal{G}$, we form a *supergraph* $\mathcal{G}_{\text{s}}$ by "collapsing" each SCC—leaf or non-leaf, with at least two vertices—into a *supernode*. First, $\mathcal{G}_{\text{s}}$ cannot contain any directed cycle. Otherwise, all supernodes and vertices in the cycle form an SCC, and it would have been collapsed into a supernode. Further, if $\mathcal{G}$ has no leaf SCC, meaning that $\mathcal{G}_{\text{s}}$ has no leaf supernode, then every supernode and non-leaf vertex must have a path to a leaf vertex. This means $\mathcal{G}$ is grounded. ∎

REFERENCES

[1] 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, June 2006.

[2] S. El Rouayheb, A. Sprintson, and C. Georghiades, "On the index coding problem and its relation to network coding and matroid theory," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3187–3195, July 2010.

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

[5] L. Ong and C. K. Ho, "Optimal index codes for a class of multicast networks with receiver side information," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Ottawa, Canada, June 2012, pp. 2223–2228.

[6] T. J. Oechtering, C. Schnurr, and H. Boche, "Broadcast capacity region of two-phase bidirectional relaying," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 454–458, Jan. 2008.

[7] L. Ong, C. M. Kellett, and S. J. Johnson, "On the equal-rate capacity of the AWGN multiway relay channel," *IEEE Trans. Inf. Theory*, vol. 58, no. 9, pp. 5761–5769, Sept. 2012.

[8] 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.

[9] J. Bang-Jensen and G. Gutin, *Digraphs: Theory, Algorithms and Applications*. Springer Verlag, 2007.

[10] L. Ong, F. Lim, and C. K. Ho. (2013, May 13) The multi-sender multicast index coding. [Online]. Available: http://arxiv.org/abs/1305.2679