NOVA

University of Newcastle Research Online

nova.newcastle.edu.au

Ong, Lawrence; "A new class of index coding instances where linear coding is optimal". Published in Proceedings of the 2014 International Symposium on Network Coding (Aalborg, Denmark 27-28 June, 2014) (2014)

Available from: http://dx.doi.org/10.1109/NETCOD.2014.6892122

© 2014 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Accessed from: http://hdl.handle.net/1959.13/1064767

A New Class of Index Coding Instances Where Linear Coding is Optimal

Lawrence Ong
School of Electrical Engineering and Computer Science, The University of Newcastle, Australia

***Abstract*—We study index-coding problems (one sender broadcasting messages to multiple receivers) where each message is requested by one receiver, and each receiver may know some messages a priori. This type of index-coding problems can be fully described by directed graphs. The aim is to find the minimum codelength that the sender needs to transmit in order to simultaneously satisfy all receivers' requests. For any directed graph, we show that if a maximum acyclic induced subgraph (MAIS) is obtained by removing two or fewer vertices from the graph, then the minimum codelength (i.e., the solution to the index-coding problem) equals the number of vertices in the MAIS, and linear codes are optimal for this index-coding problem. Our result increases the set of index-coding problems for which linear index codes are proven to be optimal.**

I. Introduction

We consider index-coding problems, first introduced by Birk and Kol [1], where a sender communicates with multiple receivers simultaneously through a shared broadcast medium. The aim is to find the shortest codeword that the sender needs to broadcast in order for each receiver, knowing some of the messages broadcast by the sender a priori, to obtain its requested message. Index-coding problems have been receiving much attention lately due to its equivalence to network-coding problems [2]–[4].

Each index-coding problem instance can be fully described by a directed or an undirected graph. Bar-Yossef et al. [5] characterized the optimal index codelength for graphs of the following types: (a) directed and acyclic, (b) undirected and perfect, (c) undirected odd holes of five or more vertices, and (d) undirected odd anti-holes of five or more vertices. In general, the index-coding problem remains open to date, though lower and upper bounds have been obtained [5]–[9]. A lower bound is given by the number of vertices in a maximum acyclic induced subgraph (MAIS) [5]. The *minrank* function [5] of the graph gives an upper bound (i.e., achievability), and it also gives the optimal *linear* index codelength. Both the MAIS lower bound and the minrank upper bound are NP-hard to compute [10], [11], and both have been shown to be loose in some instances [5], [12]. This implies that linear index codes, though having practical advantages of simplifying encoding and decoding, are not necessarily optimal.

In this paper, we extend existing results to a new class of problem instances: we show that if an MAIS is formed by removing two or fewer vertices, then the MAIS lower bound is achievable using linear index codes, meaning that linear index codes are optimal for this class of index-coding problems. To this end, we show that this class of graphs must contain some special configurations; by proposing a new coding scheme on these special configurations, we are able to construct the required optimal index code. This incidentally characterizes a class of (infinitely many) graphs where the minrank upper bound and the MAIS lower bound coincide.

Lawrence Ong is the recipient of an Australian Research Council Discovery Early Career Researcher Award (project number DE120100246).

II. Notation and An Existing Lower Bound to The Optimal Index Codelength

Let there be n receivers, $\{1,2,\ldots,n\}$, and each receiver i requests a message $x_i \in \mathcal{X} \triangleq \{0,1,\ldots,|\mathcal{X}|-1\}$ from the sender. The sender knows all the messages, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$. It encodes $\boldsymbol{x}$ into a length-ℓ codeword $\mathbb{E}(\boldsymbol{x}) \in \mathcal{X}^\ell$. It then broadcasts the codeword to all receivers noiselessly to allow each receiver i—who knows some prior side information $\mathcal{K}_i \subseteq \{x_1, x_2, \ldots, x_n\} \setminus \{x_i\}$—to decode its requested message, i.e., $\mathbb{D}_i(\mathbb{E}(\boldsymbol{x}), \mathcal{K}_i) = x_i$, for all $i \in \{1,2,\ldots,n\}$. Here, $\mathbb{E}(\boldsymbol{x})$ is the index code, and ℓ is the index codelength. The aim is the find the minimum index codelength, denoted by ℓ^*.

Each index-coding problem instance is completely specified by $\mathcal{K}_i$ for all $i \in \{1,2,\ldots,n\}$. It can also be fully described by a directed graph G, consisting of a set of vertices $V(G) = \{1,2,\ldots,n\}$ and a set of arcs $A(G)$. An arc from vertex i to vertex j exists, denoted by $(i \to j) \in A(G)$, if and only if receiver i knows x_j, or equivalently, $x_j \in \mathcal{K}_i$. By definition, there is no self loop or parallel arc. For an arc $(i \to j)$, vertex i is the *tail* and j the *head*. We term this graphical representation *side-information graph*. Each vertex i in G represents both message x_i and receiver i.

For the special case where for each $(i \to j) \in A(G)$, there exists an arc $(j \to i) \in A(G)$, the index-coding problem instance can also be represented by an undirected graph G'', consisting of the same set of vertices $V(G'') = V(G)$, and a set of edges $E(G'')$, where $(i,j) \in E(G'')$ if and only if $(i \to j) \in A(G)$, which also means $(j \to i) \in A(G)$.

We denote by $\ell^*(G)$ the optimal (or minimum) index codelength of an index-coding problem instance represented by graph G, which can be directed or undirected. Bar-Yossef et al. have shown the following lower bound [5]:

$$\ell^*(G) \geq \mathsf{MAIS}(G), \tag{1}$$

where $\mathsf{MAIS}(G)$ is the number of vertices in a maximum acyclic induced subgraph (MAIS) of G. An MAIS is obtained

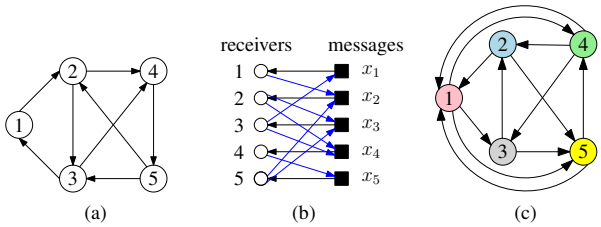

Fig. 1. The same index-coding instance represented by (a) a side-information graph, where the arrows represent what the receivers know; and (b) a bipartite graph, where the black arrows represent what the receivers want, and blue arrows what they know. Graph (c) is the complement of graph (a). Ignoring the arc direction, graph (c) is 5-chromatic. The local chromatic number is four (consider the number of colors in the out-neighborhood of vertex 1).

by finding an acyclic vertex-induced subgraph that has the largest number of vertices.

For any G, denote an MAIS by G', and the set of vertices removed by $V_{\rm r} = V(G) \setminus V(G')$, where $|V_{\rm r}|$ is the number of removed vertices. Note that $|V_{\rm r}|$ for each G is fixed, but the MAIS may not be unique. For a *vertex-induced* subgraph, when we remove a vertex y from a graph G, we remove all outgoing arcs from and all incoming arcs to y, but we must keep all other remaining arcs in G.

III. Main Results

In this paper, we derive the optimal index codelength, together with the corresponding optimal index code, for a class of graphs, for any message alphabet size $|\mathcal{X}| \geq 2$. Recall that for any directed graph G, the number of vertices we need to remove to obtain an MAIS is denoted by $|V_{\rm r}|$. With this, we now state the main result of this paper.

Theorem 1: For any G where $|V_{\rm r}| \leq 2$, the optimal index codelength is given by

$$\ell^*(G) = \mathsf{MAIS}(G) \triangleq |V(G)| - |V_{\rm r}|. \tag{2}$$

Consider the index-coding problem instance depicted in Figure 1(a) as an example. An MAIS can be formed by removing $V_{\rm r} = \{2, 3\}$ (the choice is not unique). From Theorem 1, we have $\ell^* = 3$. An optimal index code is $(x_1 \oplus x_2, x_2 \oplus x_3 \oplus x_4, x_4 \oplus x_5)$ (which is also not unique), where $\oplus$ denotes addition modulo $|\mathcal{X}|$. This problem instance is not covered by the results by Bar-Yossef et al. [5], as the side-information graph is neither acyclic nor undirected.

Neely et al. [6] and Tehrani et al. [7] have shown the following achievability:

Lemma 1 ([6], [7]): If a directed graph G contains N vertex-disjoint cycles, then the index codelength of $|V(G)|-N$ is achievable.

For Figure 1(a), their scheme only achieves $\ell = 4$, which is strictly suboptimal.

Recently, Shanmugam et al. [8] have shown that an upper bound of ℓ^* is given by the *local chromatic number* of the complement graph of G, denoted by $\bar{G}$. The local chromatic number of $\bar{G}$ is the maximum number of colors in any out-neighborhood, minimized over all proper coloring of the undirected counterpart (by ignoring the arc direction) of $\bar{G}$. From Figure 1(c), we see that this scheme achieves $\ell = 4$.

Bipartite graphs are also used to represent a more general—in fact, the most general—class of index-coding problem instances where a message can be requested by more than one receiver (cf. side-information graphs). Neely et al. [6, Theorem 1] found ℓ^* for all acyclic bipartite graphs; Yu and Neely [9] found ℓ^* for all planar bipartite graphs. The bipartite graph that represents our example is depicted in Figure 1(b), which contains cycles. Also, ignoring the arc direction, we *contract* the edges $\{(1, x_1), (1, x_2), (4, x_4), (4, x_5)\}$ to obtain a complete bipartite graph on three and three vertices, commonly denoted by $K_{3,3}$. Since any planar graph cannot contain a $K_{3,3}$ minor, Figure 1(b) is not planar.

Tehrani et al. [7] have proposed a *packet decomposition* scheme to obtain an upper bound on bipartite graphs. Achievability of the scheme was derived on the assumption that $\mathcal{X}$ is a large finite field. Even if the results can be shown to hold for smaller alphabets, the scheme can only achieve $\ell = 4$ for the instance in Figure 1(a).

Consequently, our results strictly extend the existing results. Recently, we built on the results of this paper to characterize ℓ^* for all G up to five vertices [13].

The optimal *linear* index codelength for any graph G is given by its minrank value [5]. Characterizing graphs having a certain minrank value is hard; Dau et al. [14] managed to characterize all undirected graphs whose minrank value is $|V(G)| - 2$ or $|V(G)| - 1$, and all directed graphs whose minrank value is 2 or $|V(G)|$. They are, however, unable to characterize directed graphs whose minrank value is $|V(G)| - 1$ or $|V(G)| - 2$. For any directed graph G whose $\mathsf{MAIS}(G)$ equals $|V(G)| - 1$ or $|V(G)| - 2$, we show in this paper that linear index codes are optimal, meaning that $\mathsf{MAIS}(G)$ equals its minrank. So, we have incidentally characterized a subset of directed graphs whose minrank equals $|V(G)| - 1$ or $|V(G)| - 2$.

IV. Proof of Theorem 1

As $\mathsf{MAIS}(G)$ is a lower bound to ℓ^*, we only need to prove achievability. Recall that $|V_{\rm r}|$ is the minimum number of vertices we need to remove from G to make it acyclic, i.e., to obtain an MAIS.

Firstly, suppose that $|V_{\rm r}| = 0$. Sending all messages uncoded achieves the index codelength $|V(G)|$, and we have (2).

Next, suppose that $|V_{\rm r}| = 1$. The directed graph G must contain at least one cycle; otherwise, $|V_{\rm r}| = 0$. It follows from Lemma 1 that $|V(G)| - 1$ is achievable.

Lastly, $|V_{\rm r}| = 2$. There are two possibilities for G:

(i) There exist two vertex-disjoint cycles, or

(ii) There are no two vertex-disjoint cycles.

For case (i), it again follows from Lemma 1 that $|V(G)| - 2$ is achievable. The savings of two symbols (compared to sending all $|V(G)|$ symbols uncoded) can be achieved using a cyclic code on each disjoint cycle. For example, for a cycle of length L, say $1 \to 2 \to \cdots \to L$, we send the following $(L-1)$ coded symbols, thereby saving one symbol: $(x_1 \oplus x_2, x_2 \oplus x_3, \ldots, x_{L-1} \oplus x_L)$. Since each receiver i knows at least one other symbol in the cyclic code, it can decode its required x_i.

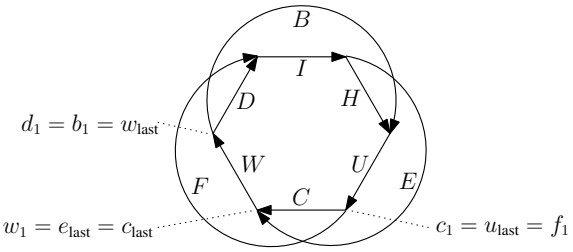

Fig. 2. An important element in proving Theorem 1 is to show that if $|V_{\text{r}}| = 2$ and condition (ii) is true, then G must contain a subgraph G_{sub} shown above. Here, every arrow represents a path, which is denoted by a capital letter. The paths do not share common vertices except the end points. Vertices in each path is denoted by the corresponding small letter, indexed in the direction of the arcs, e.g., path C is $c_1 \to c_2 \to \cdots \to c_{\text{last}}$. All paths except I, W, and U must contain one or more arcs.

The main challenge of Theorem 1 is to show that for case (ii), even though we cannot find two vertex-disjoint cycles, we can still save two symbols. To this end, we will show that if $|V_{\text{r}}| = 2$ and if condition (ii) above is true, then there exists a subgraph in G of a certain configuration, stated as follows:

Lemma 2: If $|V_{\text{r}}| = 2$, and there are no two vertex-disjoint cycles (i.e., condition (ii)), then G must contain a subgraph (not necessarily an induced subgraph) shown in Figure 2.

We will design a special code on this subgraph to show that $|V(G)|-2$ is indeed achievable. We will first present the code in the next section, and then prove Lemma 2 in Section V.

A. A New Coding Scheme

We now propose a new coding scheme that achieves the codelength $|V(G)| - 2$ for case (ii). We need to show that each receiver $i \in V(G)$ can decode its intended message, i.e., x_i. We will propose a code for the subgraph in Figure 2, denoted by G_{sub}, and send the rest of the messages (which correspond to the vertices in $V(G) \setminus V(G_{\text{sub}})$) uncoded. This means all receivers $i \in V(G) \setminus V(G_{\text{sub}})$ can decode their intended messages, and we only need to show that all receivers $j \in V(G_{\text{sub}})$ can also decode their intended messages. We propose the following coding strategy: for each vertex $a \in V(G_{\text{sub}})$, with all its outgoing arcs in G_{sub} denoted by $\{(a \to a_{\text{out}1}), (a \to a_{\text{out}2}), \ldots, (a \to a_{\text{out}T})\}$, we send the code symbol $x_a \oplus x_{a_{\text{out}1}} \oplus x_{a_{\text{out}2}} \oplus \cdots \oplus x_{a_{\text{out}T}} \in \mathcal{X}$.

For each path in G_{sub} (denoted by a capital letter), we denote the vertices therein by its corresponding small letter, indexed in the direction of the arcs. For example, path C is $c_1 \to c_2 \to \cdots \to c_{\text{last}}$. In Figure 2, we have $c_1 = u_{\text{last}} = f_1$, i.e., the first vertex in path C is the last vertex in path U, which is also the first vertex in path F. We use the above coding strategy to send the code symbol for all vertices in G_{sub} except c_1 and d_1. By design, all receivers—except c_1 and d_1—can decode their requested messages, as each receiver a knows a priori the messages corresponding to the head of all outgoing arcs from a. So, we only need to show that receivers c_1 and d_1 can decode their respective requested messages, x_{c_1} and x_{d_1}.

We start with receiver $c_1 = f_1$. Knowing x_{f_2} a priori, it decodes along path F (i.e., $x_{f_2} \oplus x_{f_3}, x_{f_3} \oplus x_{f_4}, \ldots, x_{f_{\text{last}-1}} \oplus x_{f_{\text{last}}}$), to get $x_{f_{\text{last}}} = x_{i_1}$, and continues along path I to get $x_{i_{\text{last}}} = x_{e_1}$. In the event that path I has zero arc, it can also obtain x_{e_1}, which is $x_{f_{\text{last}}}$. Also knowing x_{c_2} a priori, receiver c_1 decodes along path C to get $x_{c_{\text{last}}} = x_{e_{\text{last}}}$, from which it can decode backward along path E to get x_{e_2}. Having decoded x_{e_1} earlier and now x_{e_2}, it obtains x_{h_2} from $x_{e_1} \oplus x_{e_2} \oplus x_{h_2}$. With x_{h_2}, it decodes along path H and then path U to get $x_{u_{\text{last}}} = x_{c_1}$. In the event that path U has zero arc, the receiver would have obtained $x_{c_1} = x_{h_{\text{last}}}$ earlier.

We can similarly show that receiver d_1 can decode x_{d_1}.

In this coding scheme, we send one symbol for each vertex (coded symbols for $V(G_{\text{sub}})$, and uncoded symbols for the rest) except for c_1 and d_1. We have shown that this index code satisfies the decoding requirements of all receivers, meaning that $|V(G)| - 2$ is achievable. ∎

V. Proof of Lemma 2: A Special Configuration

We first give an intuition for Lemma 2, by showing that there must exists three interlinked cycles in G, in Subsection V-A. In Subsections V-B to V-F, we prove that these three interlinked cycles must assume the configuration in Figure 2.

A. The Existence of Three Interlinked Cycles

Let $V_{\text{r}} = \{u, v\}$, i.e., vertices u and v are removed from G to get an MAIS. We first show the following:

Proposition 1: There exist three cycles in G, each containing either u, v, or both u and v.

Proof: Every cycle must contain u, v, or both. Otherwise, removing u and v will not give an acyclic induced subgraph.

Suppose that there is only one cycle in G. Removing any vertex from the cycle gives an acyclic induced subgraph. Hence, $|V(G)| - \mathsf{MAIS}(G) = 1$. (Contradiction)

Suppose that there are only two cycles in G. Note that these two cycles cannot be vertex-disjoint, as per condition (ii) above. So, these two cycles must shared at least one vertex, and removing only this shared vertex gives an acyclic induced subgraph, i.e., $|V(G)| - \mathsf{MAIS}(G) = 1$. (Contradiction)

So, there must exist at least three cycles. ∎

We further show some properties of these three cycles:

Proposition 2: There exist three cycles in G, where

1) any two cycles must have at least one common vertex, and
2) the three cycles do not have any common vertex.

Proof: It follows from Proposition 1 that there are at least three cycles. As no two cycles are vertex-disjoint, we have property 1. Arbitrarily select one cycle, say C'. Consider every other cycle $C_k \neq C'$, and denote the set of common vertices between C_k and C' as $V_{\text{common}}(k) \triangleq V(C_k) \cap V(C')$. Since every C_k shares some vertex with C', we have $V_{\text{common}}(k) \neq \emptyset$.

Now suppose that $\bigcap_{\text{all } C_k \neq C'} V_{\text{common}}(k) \neq \emptyset$, meaning that some vertex is shared among all cycles. Then removing only this vertex from G would have resulted in an acyclic subgraph (contradiction). So, there must exist two cycles, say C_1 and C_2, where $V_{\text{common}}(1) \cap V_{\text{common}}(2) = \emptyset$. Selecting C', C_1, and C_2 gives property 2. ∎

Denote the subgraph formed by the three cycles in Proposition 2 by G_{sub}. We have the following:

Proposition 3: The subgraph G_{sub}, formed by the three cycles in Proposition 2, satisfies both the following: (1) we cannot find two vertex-disjoint cycle in G_{sub}, and (2) we need to remove two—not fewer—vertices to make G_{sub} acyclic.

Proof: Since G cannot contain two vertex-disjoint cycles, so does any of its subgraphs. We have property 1. Denote by N the minimum number of vertices we need to remove to make G_{sub} acyclic. From Proposition 2, there is no common vertex among the three cycles. So, removing any one vertex will not disconnect all three cycles simultaneously, i.e., $N \geq 2$. On the other hand, we only need to remove two vertices, V_{r}, to make G acyclic. So, removing $V_{\text{r}} \cap V(G_{\text{sub}})$ from G_{sub} will definitely make it acyclic, i.e., $N \leq 2$. So, we have property 2. ■

Note that these three cycles, G_{sub}, capture all the constraints we impose on G in Lemma 2.

B. The Three Interlinked Cycles Must Assume Figure 2

We will proceed to show that G_{sub} must assume the configuration in Figure 2. We will build the configuration from a cycle, say C_1, in G_{sub}. We call it the *center cycle*. We re-label the vertices in G_{sub} such that the vertices in C_1 are in ascending order in the direction of the arcs, i.e., $1 \to 2 \to \cdots \to (|V(C_1)|-1) \to |V(C_1)| \to 1$, where the choice of vertex 1 is arbitrary.

For any path P that originates from vertex b and terminates at vertex c, i.e., $b \to \cdots \to c$, we refer to all $\{z : z \in V(P) \setminus \{b, c\}\}$ as *inner vertices*. Here, we allow $b = c$; in such a case, P is a cycle.

We first show the following:

Proposition 4: Consider the subgraph G_{sub} and the cycle C_1 in the subgraph. Every arc not in C_1 belongs to some *outer path*, defined as a path that originates from a vertex in C_1 and terminates at a vertex (which can be the same vertex) in C_1, but with all arcs and all inner vertices (if exists) not in C_1.

Proof: Since G_{sub} is constructed by three cycles, any arc, say $(i \to j)$, not in C_1 must belong to either C_2 or C_3 (or both). Furthermore, from Proposition 2, C_2 and C_3 must each share some vertex with C_1. Hence, $(i \to j)$ must belong to an outer path that originates from C_1 and terminates at C_1. ■

Note that the outer paths cannot form any cycle outside C_1. Otherwise, we have two vertex-disjoint cycles.

It follows from Proposition 4 that G_{sub} consists of only a cycle C_1 and outer paths (from C_1 and back to C_1). Figure 4(a) shows an example of G_{sub} where C_1 is marked with thick arrows and all outer paths thin arrows.

We now prove a key proposition for proving Lemma 2.

Proposition 5: Remove vertex 1 in C_1. There exists another cycle in G_{sub} if and only if there is an outer path from some $b \in V(C_1) \setminus \{1\}$ to some $c \in V(C_1) \setminus \{1\}$, where $b \geq c$.

Proof: [The *only if* part:] We remove vertex 1. If there is another cycle, then there is a vertex (not vertex 1) in C_1 that has a path back to itself (this is because any cycle must share some vertex with C_1). This cannot happen if every outer path terminates at a higher-indexed vertex (we can ignore all outer paths that originate or terminate at vertex 1 as the vertex has been removed). So, there must exist an outer path with $b \geq c$.

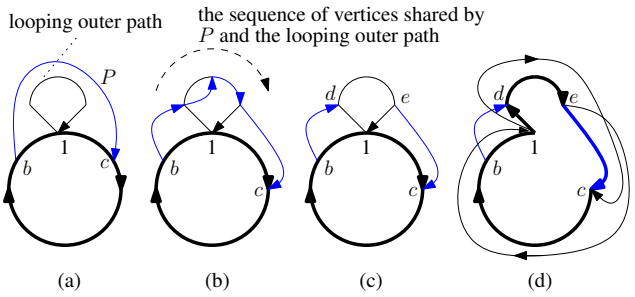

Fig. 3. Case 1 where there exists a looping outer path (drawn with thin black lines) that starts and ends at vertex 1. The center cycle C_1 is drawn with thick lines, and the second outer path (denoted as P) from b to c, blue lines. To get another cycle after removing vertex 1, we must have that $1 < c \leq b \leq |V(C_1)|$, as shown in subfigure (a). However, there are two vertex-disjoint cycles in subfigure (a). So, P must touch the looping outer path, as shown in subfigure (b). Taking the segment of P from C_1 to the looping outer path, and that from the looping outer path back to C_1, we have subfigure (c). We can re-draw the path from 1 to c and that from e to 1 in subfigure (c) to get subfigure (d), where we have drawn the new center cycle with thick lines.

[The *if* part:] Clearly, if $b = c$, we have another cycle formed by the outer path. Otherwise, i.e., $b > c$, the outer path and the path along C_1 from c to b form a cycle. See Figure 3(a) for an example. ■

Next, we define a *looping* outer path as an outer path that originates and terminates at the same vertex in C_1. The graph G_{sub} can be categorized as follows:

- there exists one or more looping outer path (Case 1), or
- there is no looping outer path (which we will further divide into Cases 2 and 3).

We will show that in any case, we have Figure 2.

C. Case 1: There Exists a Looping Outer Path

Suppose that there exists a looping outer path from and to vertex $1 \in V(C_1)$. This incurs no loss of generality as the choice of vertex 1 is arbitrary. Removing vertex 1 disconnects cycle C_1 and the cycle formed by the looping outer path. Recall that we need to remove two vertices to disconnect all cycles in G_{sub}. So, there must exist another cycle in G_{sub}.

From Proposition 5, there exists another outer path P from $b \in V(C_1) \setminus \{1\}$ to $c \in V(C_1) \setminus \{1\}$, where $b \geq c$. The outer path P must share some vertex with the looping outer path; otherwise there exist two cycles as shown in Figure 3(a).

Re-label the inner vertices of the looping outer path in ascending order, as follows: $1 \to (|V(C_1)|+1) \to (|V(C_1)|+2) \to \cdots \to (|V(C_1)|+L) \to 1$, where L is the number of inner vertices. It follows that the sequence of vertices shared by P and the looping outer path (in the order of the direction of P) must be in ascending order (see Figure 3(b)); otherwise, a cycle forms outside C_1.

See Figure 3(c). Consider only the following segments of P: (i) from b to the vertex where P first touches the looping outer path, denoted by d; and (ii) the vertex where P leaves the looping outer path, denoted by e, to c. It follows that $d \leq e$. By construction, all paths in Figure 3(c) do not share inner vertices, i.e., they touch only at end points. Finally, re-draw Figure 3(c) to get Figure 3(d), which is isomorphic to Figure 2.

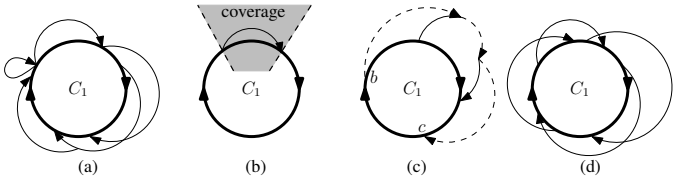

Fig. 4. We can always draw G_{sub} as in subfigure (a), i.e., a center cycle C_1 and outer paths from C_1 and back to C_1. Subfigure (b) shows the coverage of an outer path, i.e., vertices in C_1 in the gray area *excluding* the two end points. Subfigure (c) shows that when multiple outer paths originate from one vertex, we consider only the outer path with the largest coverage, i.e., the dotted path from b to c. The outer paths in subfigure (d) provide full coverage.

Note that vertices 1, b, and d must be unique. We have shown that if there is a looping outer path, then we have the configuration in Figure 2, where path I has zero arc, paths W and U possibly have zero arc (if $b = c$ and/or $d = e$), and all other paths must contain at least one arc.

D. No Looping Outer Path

For a non-looping outer path from vertex $b \in V(C_1)$ to $c \in V(C_1) \setminus \{b\}$, we say that the vertices in C_1 from b to c (in the direction of the arcs in C_1) but excluding b and c is *covered* by this outer path. See Figure 4(b) for an example.

For the purpose of this paper, we exclude outer paths with strictly smaller coverage, or multiple outer paths with equal coverage. Referring to Figure 4(c), consider an outer path that originates from b. Suppose that it has multiple paths back to C_1. We consider only the path (back to C_1) that has the *largest coverage*. Similarly, for any path that terminates at c, we consider only the path (leaving C_1) that has the largest coverage. By doing this, each path that we consider has a unique originating vertex and a unique terminating vertex.

We now show the following property:

Proposition 6: If there is no looping outer paths in G_{sub}, then all largest-covering outer paths must, together, provide full coverage for the cycle C_1. In other words, every vertex in C_1 must be covered by some outer path(s).

Proof: Consider any vertex $a \in V(C_1)$. Re-label a as vertex 1, and other vertices $V(C_1)$ in ascending order in the arc direction. Remove vertex 1 from G_{sub}. There must exist another cycle. It follows from Proposition 5 that an outer path P from b to c must exist, where $1 < c < b \leq |V(C_1)|$ ($c \neq b$ since there is no looping path), meaning that this outer path must cover vertex 1. We can safely ignore other outer paths that provide smaller or equal coverage, because if P does not cover vertex 1, then none of the ignored outer paths does. Since the choice of a is arbitrary, we have Proposition 6. ■

For example, the outer paths in Figure 4(d) provides full coverage for C_1, but the outer paths in Figures 4(b)–(c) do not. Removing one uncovered vertex from C_1 makes G_{sub} acyclic.

Now, we consider G_{sub} that consists of the cycle C_1 and all outer paths that provide the largest coverage (i.e., we remove all other arcs have gives smaller or equal coverage). We are ready to proceed with Cases 2 and 3:

- There is no looping outer path, and no two outer paths have any common inner vertex (Case 2).
- There is no looping outer path, and there exist two outer paths sharing the same inner vertex (Case 3).

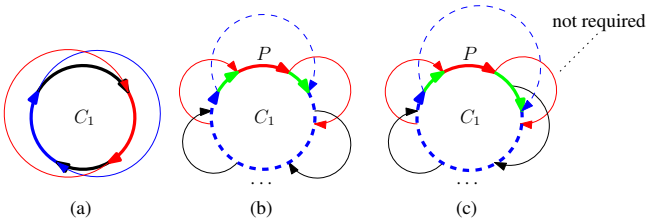

Fig. 5. (a) If two outer paths (drawn using thin lines) provide full coverage, we can always form two disjoint cycles, one marked with red and the other blue. (b) G_{sub} with $K \geq 4$ outer paths providing full coverage can be converted to $K - 2$ outer paths providing full coverage. (c) If the blue and the right black outer paths (non-adjacent) give overlapping coverage, then the right red outer path is actually redundant, i.e., $K - 1$ outer path is sufficient to give full coverage, instead of K.

E. Case 2: No Looping Outer Path, and All Outer Paths Do Not Share Inner Vertices

We will show that we can always choose three outer paths to provide full coverage.

First, note that one outer path cannot provide full coverage. Now, suppose that we can find two outer paths providing full coverage. We show in Figure 5(a) that we can always form two vertex-disjoint cycles. So, this also cannot happen.

Next, suppose that we can find three outer paths providing full coverage, we have Figure 2. As there is no looping outer path, the nine paths in Figure 2 each have one or more arcs.

Finally, we show that if we can find $K \geq 4$ outer paths providing full coverage, we can always modify the cycles such that $(K - 2)$ outer paths provide full coverage. We illustrate this in Figure 5(b). We do the following:

1) Combine the dashed blue arrows to be the new C_1.
2) Combine the two adjacent (red) outer paths, and the red arc in C_1 that connects the two red outer paths (i.e., P, which can be of zero length) into a new outer path.
3) Remove the two green paths in C_1. Each green path must contain at least one arc; otherwise, the outer paths cannot provide full coverage.

Note that by doing this, the new graph still retains the structure of a cycle with outer paths covering it. The new graph has $K - 2$ outer paths providing full coverage. This reduction is always possible as the coverage of two non-adjacent outer paths does not overlap, illustrated in Figure 5(c).

By repeating this step, starting from any $K \geq 4$ outer paths, we can find a graph with $K = 2$ or $K = 3$ outer paths. As $K = 2$ is not possible, we will always get a graph with $K = 3$ outer paths providing full coverage, i.e., Figure 2.

F. Case 3: No Looping Outer Path and Two Outer Paths Share Some Inner Vertices

Let the two outer paths that share some common inner vertex be P and Q, and one of the shared inner vertices be z. Further, let the originating and terminating vertices of P be p_1 and p_{last} respectively, and those of Q be q_1 and q_{last}. Here, $p_1 \neq p_{\text{last}}$ and $q_1 \neq q_{\text{last}}$ as there is no looping outer path, and

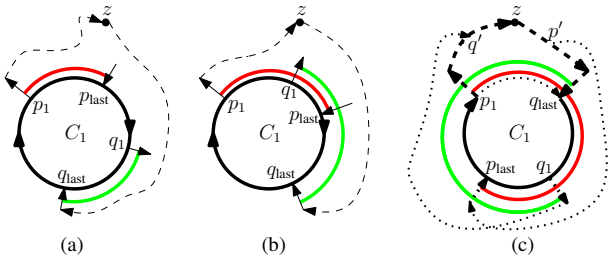

Fig. 6. The overlapping of the coverage of two outer paths, where the red line represents the coverage of the outer path P ($p_1 \to \cdots \to p_{\text{last}}$), and the green line the outer path Q ($q_1 \to \cdots \to q_{\text{last}}$)

$p_1 \neq q_1$ and $p_{\text{last}} \neq q_{\text{last}}$ as no two outer paths have the same originating or terminating vertices.

Now, the coverage of P and Q can be either (a) non-overlapping, (b) overlapping once, or (c) overlapping twice, as shown in Figure 6. The red line shows the coverage of P, and the green line Q. By definition, there is a subpath from p_1 to z along P and another subpath from z to p_{last} along P. The two subpaths must be vertex-disjoint, except z, as there is no cycle in P. Similarly, we have two vertex-disjoint paths from q_1 to z, and from z to q_{last}, both along Q. This means, there is an subpath from p_1 to q_{last} through z, and another from q_1 to p_{last} through z. So, $p_1 \neq q_{\text{last}}$, $q_1 \neq p_{\text{last}}$, as there is no looping outer path, and hence p_1, p_{last}, q_1, and q_{last} are distinct.

Suppose that we have Figure 6(a). The largest-covering outer path from p_1 should terminate at q_{last}, and that from q_1 at p_{last}. The outer path from p_1 to q_{last} and that from q_1 to p_{last} should have been chosen. This means the largest-covering paths actually overlap twice, i.e., we should have Figure 6(c).

Suppose that we have Figure 6(b). The outer path from p_1 to q_{last}, through z, gives the largest coverage, and it would have been chosen.

So, we can only have the configuration in Figure 6(c), where the coverage overlaps twice. The coverage from p_1 to q_{last} is smaller than that from p_1 to p_{last}. So, the largest-covering outer path from p_1 was correctly identified. Similarly, the largest-covering outer path from q_1 terminates at q_{last}.

We will now show that we can always get Figure 2 from Figure 6(c). Recall that there is a subpath from p_1 to z and another subpath from z to q_{last}, and these two subpaths are vertex-disjoint, except z. We denote the outer path from p_1 to q_{last} (through z) by Z (drawn with a thick dashed line).

Next, recall that there is a subpath from q_1 to z, and another from z to p_{last}. So, the subpath from q_1 to z must meet Z. Denote the vertex it first meets Z as q'. Similarly, the subpath from z to p_{last} must share some common vertices with Z (at least vertex z). Let the last shared vertex be p'. With this construction, Z, the subpath from q_1 to q', and the subpath from p' to p_{last} are vertex-disjoint, except at p' and q'.

We now re-draw Figure 6(c) as follows: Let the path from q_{last} to p_1 along C_1 (drawn with a thick solid line) plus path Z (drawn with a thick dashed line) be the center cycle, and let the subpaths (drawn with dotted arrows) (i) from p_1 to q_{last} along C_1, (ii) from p' to p_{last}, and (iii) from q_1 to q' be the three outer paths. Note that only p' and q' can co-locate. This is isomorphic to Figure 2, with path I possibly having zero arc (if $p' = q' = z$).

Combining the Cases 1–3, we have Lemma 2. ■

VI. Conclusion

We have solved a new class of index-coding problems, characterized by their side-information graphs. We have shown that for any side-information graph whose maximum acyclic induced subgraph (MAIS) can be formed by removing two or fewer vertices, the optimal index codelength (i) equals the order of the MAIS, and (ii) is achievable by linear index codes. We proved this by constructing a special subgraph that the side-information graph must contain, and design a linear index code on it. We then show that the liner index code achieves the MAIS lower bound.

We have incidentally characterized a subset of directed graphs whose minrank equals $|V(G)|-1$ or $|V(G)|-2$, where $|V(G)|$ is the order of the graph G.

The result of this paper has led to another recent result: for any side-information graph of up to five vertices, the optimal index codelength is achievable using linear index codes [13].

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. (The conference version of this paper appeared in *INFOCOM* 1998)

[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] M. Effros, S. El Rouayheb, and M. Langberg, "An equivalence between network coding and index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, July 7–12 2013, pp. 967–971.

[4] M. F. Wong, M. Langberg, and M. Effros, "On a capacity equivalence between network and index coding and the edge removal problem," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, July 7–12 2013, pp. 972–976.

[5] 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.

[6] M. J. Neely, A. S. Tehrani, and Z. Zhang, "Dynamic index coding for wireless broadcast networks," in *Proc. 31st IEEE Conf. Comput. Commun. (INFOCOM)*, Orlando, USA, Mar. 25–30 2012, pp. 316–324.

[7] A. S. Tehrani, A. G. Dimakis, and M. J. Neely, "Bipartite index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, USA, July 1–6 2012, pp. 2256–2260.

[8] K. Shanmugam, A. G. Dimakis, and M. Langberg, "Local graph coloring and index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, July 7–12 2013, pp. 1152–1156.

[9] H. Yu and M. J. Neely, "Duality codes and the integrality gap bound for index coding," in *Proc. 51st Allerton Conf. Commun. Control Comput. (Allerton Conf.)*, Monticello, USA, Oct. 2–4 2013.

[10] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, New York-London, 1972, pp. 85–104.

[11] R. Peeters, "Orthogonal representations over finite fields and the chromatic number of graphs," *Combinatorica*, vol. 16, no. 3, pp. 417–431, Sept. 1996.

[12] 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.

[13] L. Ong, "Linear codes are optimal for index-coding instances with five or fewer receivers," in *accepted and to be presented at ISIT*, 2014. [Online]. Available: http://arxiv.org/abs/1401.7369

[14] S. H. Dau, V. Skachek, and Y. M. Chee, "Optimal index codes with near-extreme rates," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, USA, July 1–6 2012, pp. 2241–2245.