Index Coding Increases Capacity of Information-Centric Networks

Mohsen Karimzadeh Kiskani†, and Hamid R. Sadjadpour†,

Abstract

The index coding problem studies transmission policies when the source node broadcasts encoded data to users with side information. This paper extends the index coding problem to cases when the source node can reach users through multi-hop communications. The new approach is called Modified Index Coding (MIC) which can be applied to both wireless and wired networks. We demonstrate the benefits of our approach by applying MIC to Information-Centric Networks (ICN). We demonstrate that the combination of ICN and MIC requires a hybrid caching scheme that includes both central and distributed caching to support two different goals. The approach results in a combination of conventional caching in ICN and a new distributed caching scheme across nodes in the network. Our analysis demonstrates that capacity improvement can be achieved by the new architecture. Simulation results compare the capacity improvement to traditional ICN architecture.

Index Terms

Information Centric Network, Index Coding, Caching, Network Coding.

I. INTRODUCTION

Information Centric Network (ICN) architectures [1]–[4] were introduced based on the premise that in most Internet applications, users are interested in accessing the content regardless of the location of delivery as long as the information is secure. ICN focuses on the content delivery without any consideration on where the content is obtained. The key question that ICN attempts to address is “how to securely deliver huge amount of contents that are distributed in different locations and requested by many users?”

M. K. Kiskani† and H. R. Sadjadpour† are with the Department of Electrical Engineering, University of California, Santa Cruz. Email: {mohsen, hamid}@soe.ucsc.edu
ICN addresses this question by utilizing a naming architecture where the content is retrieved by its name and defining a naming taxonomy that makes the content independent of its source or location. Further, it allows the contents to be cached in the network, preferably close to the destinations. This unique content recovery requires content-based routing in order to find the content in the network using appropriate name resolution infrastructure to map a name to one copy of the content. This new approach has provided significant benefits for content delivery in the network at the expense of additional overhead to keep track of content locations in different caches.

Index Coding (IC) \cite{6}, \cite{7} combines applications of coding with the nodes capable of caching some of the data which provides capacity gains. The original IC problem was based on the assumption that the channel is a broadcast channel and the source can reach all the nodes in one transmission. We will modify this concept in this paper to accommodate other classes of the channels such as wired or multihop wireless networks. Figure 1 demonstrates an example of IC problem with source having six messages and each node $N_1$ to $N_6$ needs one message and has a subset of all messages. For example, node $N_4$ needs message $m_4$ while it has prior side information $m_5$ and $m_6$. The objective is to find an optimal encoding scheme that allows all nodes to receive their required messages with minimum number of transmissions. In this paper, we only consider linear coding approach. If all the nodes in figure 1 are within transmission range of the source, then the source can send only two messages of $m_1 + m_2 + m_3$ and $m_4 + m_5 + m_6$ to allow all nodes to retrieve their requested messages. For example, node $N_4$ can add $m_5$ and $m_6$ to encoded message $m_4 + m_5 + m_6$ in order to recover message $m_4$. A question that we intend to address in this paper is that “how can we take advantage of caching in ICN and modify IC concept in order to increase the overall throughput capacity of the network?”

The rest of the paper is organized as follows. In Section II we define the modified index coding (MIC) and the new hybrid caching schemes. Section III describes proposed ICN architecture. In section IV, we prove that MIC can be efficiently utilized to increase the capacity of the ICN. Simulation results demonstrate the throughput capacity improvement in section VI. Section VII concludes the paper.
Fig. 1. An example of index coding problem.

II. MODEL AND PROBLEM FORMULATION

A. Modified Index Coding (MIC)

The IC problem [6], [7] was originally formulated for wireless broadcast channels such as in satellite communication applications. In this paper, we assume the source node can access the receiver nodes through multihop communications. The new scheme, called Modified Index Coding (MIC) technique, can be applied to different types of networks such as wireless multihop communication network, wired network or a hybrid network consisting of both wired and wireless channels. For example, in a corporate environment, we can have information transported in a wired medium while the last hop is a wireless access point with nodes connected to the infrastructure through a wireless modem. Such scenarios are becoming more common in medium and large size corporate campuses.

**Definition 1.** A Modified Index Code $\text{MIC}(M, N)$ consists of a set of $k$ messages $M = \{m_1, \ldots, m_k\}$ and a set of receiver nodes $N$. Each node $N_i$ has a subset of messages called $L_i \subseteq M$ and requests one message $m_i$ that node $N_i$ does not have, i.e., $m_i \in M$ and $L_i \subseteq M \setminus \{m_i\}$. Each message can be divided into $n$ packets, i.e., $m_i = \{m_{i1}, \ldots, m_{in}\}$. Each packet also belongs to an alphabet taken from a $q$-ary finite Field $\Gamma$. Therefore, we have $m_i = \{m_{i1}, \ldots, m_{in}\} \in \Gamma^n$. We further define $\mathcal{N} = \{m_{11}, \ldots, m_{1n}, \ldots, m_{k1}, \ldots, m_{kn}\} \in \Gamma^{nk}$. At any given time, $k_1 \leq k$ receiver nodes are requesting some messages. The source defines groups of receiver nodes $k_1^{1}, \ldots, k_1^{P}$.

---

1Extension of this definition to multiple nodes is straightforward.
where $k_1 + \ldots + k_p = k_1$. Modified index code for $\text{MIC}(M, N)$ is a function $f_i : \Gamma^{nk} \rightarrow \Gamma^{\ell_i}$, for an integer value of $\ell_i$ and groups of nodes $k_i^1$ that satisfies for each receiver node $N_i = (m_i, L_i) \in N$ within this group $k_i^1$ a function $\Phi_i : \Gamma^{\ell_i+n|L_i|} \rightarrow \Gamma^n$ such that the desired message can be decoded for that particular node, i.e., $\Phi_i(f_i(\mathcal{K}), L_i) = m_i, \forall \mathcal{K} \in \Gamma^{nk}$.

Note that in this new definition, we multicast different coded messages to these $p$ groups, each one consists of $k_i^1$ receiver nodes where $1 \leq i \leq p$. IC definition subsumes MIC since broadcast is a special case of multicast when the set of receiver nodes include the entire network. Under the new definition, we can apply MIC for both wireless and wired networks. MIC is utilized to improve the capacity of ICN architecture.

Figure 1 can be an example of MIC. Suppose that any two nodes can communicate only when there is an arrow between them. For instance, the source node in Figure 1 can only communicate with nodes $N_1$ and $N_2$. In this example, $p = 2$ and $k_1^2 = \{N_4, N_5, N_6\}$. The source node multicasts two encoded messages of $m_1 + m_2 + m_3$ and $m_4 + m_5 + m_6$ to $k_1^1$ and $k_1^2$ groups respectively. These two encoded messages are the minimum number of transmissions (optimum) that will achieve the desired outcome. However, in general the problem of finding the best encoding strategy is an NP-hard problem.

### B. Hybrid Caching

One of the main features of ICN is the ability of the network to cache the requested contents in the network at different locations in order to serve the users with lower latency and improve the throughput capacity in the process by bringing the contents closer to the users. This feature seems to be very attractive in separating the content from any unique source node in order to find the nearest content to the client node. However, this by itself creates certain challenges for network designers. One major challenge is how to locate the closest cached content in the network? Another challenge is to design a caching policy that will increase the throughput capacity while reducing the latency. We introduce a new hybrid caching technique that requires minimum overhead related to locating stored cache contents.

In our architecture, we use caching for two different purposes. We cache the contents in some locations in the network in order to provide it to users similar to the original approach in ICN. However, most users have significant storage capacity that is not being used. For example, it is
now common for a laptop to have Terabyte of storage mostly not being utilized by majority of the users. We use this enormous distributed storage capacity for improving the data distribution in ICN architecture. We propose that each user shall keep any data object that is requesting from the network. Therefore, each user allocates a predefined portion of its memory to keep the data objects that it has already obtained. By the discussion that we had in previous section related to MIC, it should be clear that these data stored by different users throughout the network will be used to extract the desired message when the node receives a combination of multiple messages that it stores except the current requested content. The network encoded message is multicasted by the central cache system that is serving these nodes. The distributed cache system is not used to deliver contents to other users as it is common in current ICN approach. The problem with distributed caching is the significant overhead associated to this approach. We suggest that the requested contents by different users should always come from the central cache system or from source node that has the content. In our proposed architecture, caching are used for two different purposes as described above.

Note that by taking advantage of the MIC, we need one multicast session to replace $k^1$ unicast sessions. It is easy to show that one multicast session always consumes less channel bandwidth in the network than $k^1$ unicast sessions [20]. This reduction in network resource usage can be very helpful specially when the size of contents are large. Another advantage of this architecture is the fact that since the central cache system knows what contents each node has cached before, there is no need for the extra overhead to update each cache (node) information in the network. As long as the central cache system can keep track of the cached contents in different nodes, then the overhead is very small. Note that since each node receives the content via this cache, then that information is already available to the central cache system.

In current network architecture, the communication is between source and destination. ICN architectures take advantage of caching to bring the content closer to the users. In all existing ICN architectures point-to-point concept is still used in delivering data from cache to destination. The only difference is that they replace the source with a cache that is closer to the destination. However, in our proposed ICN architecture, we no longer use point-to-point communication as a means of data delivery. In the new architecture, we use one to many communications to send multiple messages to multiple users from central cache system utilizing multicast communications. This is one of the main advantages of the new architecture that reduces the bandwidth
usage per message in the network. To the best of our knowledge, this concept has never been used in previous ICN architectures.

III. PROPOSED ICN ARCHITECTURE

In this section, we will describe how to take advantage of MIC concept in order to derive a new ICN architecture. We assume each group of nodes in the network is served by a unique router that also caches the information. In this context, if a node requests a content, this request will be directed toward that particular router. The router either sends the information directly to the node or finds the source for the requested content. Further, we assume when a node receives a content, it will keep this content in its cache. In Figure 2, all the routers that are shaded are responsible for serving different groups of nodes. The selection of these routers is based on the number of nodes that are connected to that router either directly or through multiple hops. In general, there is a trade-off between latency, speed of router and the maximum number of nodes assigned to a router.

Fig. 2. Example of serving nodes by routers. Router R5 serves nodes N1 and N2 and router R2 serves nodes N3 and N4. Solid lines represent one hop and dotted lines represent multiple hops.

When a content is delivered to a node, the node will keep a copy of this content in its cache. Now let’s assume each node has a subset of contents in its cache (see figure 2). When some
of these nodes request different contents from the assigned router, this router network encodes the requested contents by utilizing modified index coding technique to minimize the total number of multicast transmissions to serve all these nodes.

If the content is not available in the serving router, then the router will request the content from the source (or another router on the way toward the source). Once it receives the content, it will again apply the network coding concept for this content along with some of the available contents that are requested at that time. This router also keeps one copy of the content in its cache. As we can see, under the new architecture, we do not use an aggressive caching approach that each router or node caches the contents but rather a subset of the routers cache the contents. The assumption here is that most of the contents that are being requested are popular among nodes and will be likely requested by other nodes. This is particularly true since often the content request popularity are heavy-tailed and have a distribution close to Zipf distribution. Prior studies \cite{5}, \cite{21} have shown that multiple layer caching or cooperative caching does not provide significant improvement for Zipfian distribution. For this reason, recent study \cite{16} has suggested caching scheme that takes advantage of this distribution and caches at the edges of the network. Our approach has some similarities to the technique proposed in \cite{16} by suggesting that it is sufficient to cache in a subset of nodes at the edge of the network.

Figure \ref{fig:example} demonstrates an example for our proposed ICN architecture. Let nodes $N_1$ and $N_2$ request messages $m_{10}$ and $m_8$ respectively. Both these nodes are served by router $R_5$. When these nodes send request to this router, the router will multicast $m_{10} + m_8$ to these two nodes. Node $N_1$ can add the received encoded message with $m_8$ to obtain $m_{10}$ and node $N_2$ can similarly obtain its own requested message. Similarly, for nodes $N_3$ and $N_4$ that are served by router $R_2$ via routers $R_6$ and $R_7$ respectively, $R_2$ will send $m_6 + m_9$ and each node can retrieve its requested data. All these operations are carried in Galois Field. It is quite possible that more complicated combinations of messages are sent by routers in order for nodes to decode their requested messages.

Therefore, as long as the caching policy of the user is known by the router, the router knows for each user which contents are being stored and which contents are evicted after the user reaches its maximum caching capacity. This clearly requires more processing power for each

\footnote{Some nodes can request the same content.}
router that is involved in caching but it also reduces the overhead significantly. Users can also update their cache information off peak hours with their respective routers. The other overhead is associated to when a node requests a content. Since each user has an assigned router to serve that user, the request is always directed toward that router. Clearly, the overhead associated to finding the content in the network if it is not available locally by the router is similar to the current network architectures. Therefore, our proposed architecture simplifies the overhead and content routing challenges in ICN systems.

IV. ICN Capacity Improvement Using MIC

In this section, we study the problem of capacity improvement in the proposed ICN architecture when utilizing MIC. We assume the content popularity follows a Zipfian-like distribution which is supported by many studies [10], [11]. The use of MIC in networks that a small portion of contents are requested with higher probability provides significant capacity gains as will be demonstrated here.

In the remainder of this section, we assume that the network is a hybrid network with the last hop is between a wireless router (like 802.11) and mobile users. We observe that index coding (IC) is a special case of MIC approach in which we have only one broadcast group. As it is the case in the current study of IC [7], [12], [14] in the literature, for an instance of an IC problem, the concept of dependency graph is very helpful as described below.

**Definition 2. (Dependency Graph):** Given an instance of an index coding, the dependency graph $G(V, E)$ for this problem is defined as

- For each client $N_i$ corresponds to a vertex in $V$, $N_i \in V$, and
- There is a directed edge in $E$ from $N_i$ to $N_j$ if and only if the client $N_i$ is requesting a content that is already cached in user $N_j$.

It is known from [12] that if we choose the right encoding vectors for any IC problem, for any vertex disjoint cycle in the dependency graph we can save at least one transmission. Therefore, the number of vertex-disjoint cycles in the dependency graph can serve as a lower bound for the number of saved transmissions in any IC problem. The same result also holds for an MIC problem since MIC is similar to the IC problem and a dependency graph can be defined for each subnetwork.
Assume that we have an ICN with hybrid network that is utilizing MIC. Denote the set of contents available in the whole network by \( M = \{ m_1, m_2, ..., m_n \} \) with \( m_1 \) being the most popular content and \( m_n \) being the least popular content in the network. Also, assume that the users \( N = \{ N_1, N_2, ..., N_m \} \) are being served by a specific router \( R \) and user \( N_i \) is requesting content with popularity index \( r_i \) in the current time instant. For the sake of simplicity of calculations, assume that each user has a cache of fixed size \( \delta \) in which, the contents with indices \( C_i = \{ c_i1, ..., c_{i\delta} \} \) are stored.

As suggested by [9]–[11], we can assume a Zipfian distribution with parameter \( s \) for content popularity distribution in the network. This means that the probability that \( N_i \) requests any content with index \( r \) at any time instant can be found as

\[
\text{Pr}[N_i \text{ requests content with index } r] = \frac{r^{-s}}{H_{n,s}},
\]

where \( H_{n,s} \) denotes the \( n^{th} \) generalized harmonic number with parameter \( s \).

\[
H_{n,s} = \sum_{j=1}^{n} \frac{1}{j^s}
\]

**Lemma 1.** When \( s > 1 \), for every \( 0 < \epsilon < 1 \), there exists an integer \( h = \Theta(1) \) with respect to \( n \) such that for every \( i \),

\[
\text{Pr}[r_i \leq h] \geq 1 - \epsilon.
\]

**Proof:** Based on the Zipfian distribution assumption, this probability is qual to

\[
\text{Pr}[r_j \leq h] = \frac{H_{h,s}}{H_{n,s}}.
\]

If \( s > 1 \), we have \( H_{n,s} < H_{\infty,s} = \zeta(s) \) where \( \zeta(.) \) denotes the Reimann Zeta function. If we choose \( h \) to be the first integer such that \( H_{h,s} \geq \zeta(s)(1 - \epsilon) \), we are guaranteed to have \( \text{Pr}[r_i \leq h] \geq 1 - \epsilon \). Notice that \( h \) can be chosen independently of \( n \), i.e., \( h = \Theta(1) \).

**Remark 1.** If \( 0 \leq s \leq 1 \), to make sure that \( \text{Pr}[r_i \leq h] \geq 1 - \epsilon \), the value of \( h \) grows with \( n \) but the growth rate is so small that we can still treat \( h \) as a constant number with respect to \( n \) and use lemma for practical purposes.

---

\(^3\)Popularity decreases with index number.
Therefore, based on lemma 1 and remark 1, if $h$ is chosen to be a large enough integer, with a probability very close to one, all users are requesting contents with maximum popularity index $h$. Before going further, we will bring up the following lemma from [15].

**Lemma 2.** Let $d_c > 1$ and $m \geq 24d_c$, then any graph $G^m_{f(m,d_c)}$ with $m$ nodes and at least $f(m,d_c) = (2d_c - 1)m - 2d_c^2 + d_c$ edges contains $d_c$ disjoint cycles or $2d_c - 1$ vertices of degree $m - 1$ (its structure is then uniquely determined).

**Proof:** The proof is in [15].

Since each user is requesting some content and has a subset of contents in its cache, the use of MIC for ICN architecture seems to provide some capacity improvements. The fact that with high probability, there are strong correlation between cached contents and new content requests because of Zipfian distribution of contents, it is clear that MIC will provide capacity improvement. We will now prove the efficiency of applying MIC to the ICN in the following theorem.

**Theorem 1.** For large values of $h$ and $m$, using MIC in an ICN can save on average at least $\Omega(m p_0)$ transmissions for any router serving $m$ nodes in any time slot where

$$p_0 = \frac{h^{-s}}{H_{n,s}}.$$  \hspace{1cm} (5)

**Proof:** The dependency graph $G(V,E)$ in our problem is composed of $m$ vertices $N_{r_1}, N_{r_2}, \ldots, N_{r_m}$ which corresponds to the $m$ nodes that are served by a router. Note that the existence of an edge in dependency graph depends on the probability that a node is requesting a content and another node has already cached that content. Therefore, this is a non-deterministic graph with some probability for the existence of each edge between the two vertices. In this non-deterministic dependency graph, the probability of existence of edge $(N_{r_i}, N_{r_j})$ in $E$ is equal to the probability that $r_i$ is cached in node $N_j$ and $N_i$ is requesting content $r_i$. These two probabilities are independent and therefore, the probability that in the dependency graph $G$, there exists a directed edge $(N_{r_i}, N_{r_j})$ in $E$ is at least

$$\Pr[(N_{r_i}, N_{r_j}) \in E] \geq \frac{r_i^{-s}}{H_{n,s}}.$$  \hspace{1cm} (6)

Clearly, the theorem is valid when the number of edges is more than $f(m,d_c)$.
Using lemma 1 and remark 1 when the value of \( h \) is large enough, the probability that \( r_i \) is less than \( h \) gets close to one very rapidly. This means that with a probability close to one, the edge presence probability in equation (6) can be lower bounded by \( p_0 \). Therefore, maximum number of vertex-disjoint cycles in \( G(V, E) \) can be lower bounded by the maximum number of vertex-disjoint cycles in an Erdős-Rényi random graph \( G'(m, p_0) \) with \( m \) nodes and edge presence probability \( p_0 \). Now we can use lemma 2 to find a lower bound on the number of vertex disjoint cycles in \( G'(m, p_0) \). This in turn will give us a lower bound on the number of vertex-disjoint cycles in \( G(V, E) \).

Notice that \( G'(m, p_0) \) is an Erdős-Rényi random graph and it can have a maximum of \( m(m - 1) \) directed edges. However, since every edge in this graph exists with a probability of \( p_0 \), the expected value of the number of edges in this graph is \( m(m - 1)p_0 \). This means that if \( d_c \) is chosen to be an integer such that

\[
m(m - 1)p_0 \geq (2d_c - 1)m - 2d_c^2 + d_c,
\]

then on average, \( G'(m, p_0) \) will have \( d_c \) disjoint cycles. For the purpose of our paper we can easily verify that \( d_c = \frac{mp_0}{2} \) satisfies equation (7). Therefore, with a probability close to one (for large enough values of \( h \)) the dependency graph \( G(V, E) \), on average has at least \( \Omega(mp_0) \) vertex disjoint cycles. This can be directly applied to prove the theorem.

**Theorem 2.** The achieved lower bound in theorem 2 is a tight order bound of \( \Theta(m) \).

**Proof:** Notice that the maximum number of vertex-disjoint cycles in any graph with \( m \) vertices cannot be greater than \( \frac{m^2}{2} \). However, theorem 1 proves that the maximum number of vertex-disjoint cycles in our graph is lower bounded by \( \Omega(mp_0) \). This suggests that it is indeed a tight order bound.

We can further prove that many properties of the dependency graph are independent of the number of contents in the network. This implies that the properties of the dependency graph are mainly dominated by the most popular contents in the network. As an example of these properties, we can consider the problem of finding a clique of size \( k \) in the dependency graph. A clique of size \( k \) in the dependency graph has an interesting interpretation since such a clique means that there exist a set of \( k \) users \( N_b = \{N_{b1}, N_{b2}, ..., N_{bk}\} \) such that for every \( 1 \leq i \leq k \) and for every \( 1 \leq j \leq k \), when \( j \neq i \) we have \( r_{bi} \in C_{bj} \). This particularly means that the simple
linear index code $\sum_{i=1}^{k} m_{b_i}$ can be used by the endpoint router to send the content $m_{b_i}$ to user $N_{b_i}$ for every $1 \leq i \leq k$ in just one transmission. Each user will then be able to decode the requested message using its cached contents. The following theorem proves that the existence of a clique of size $k$ is independent of the total number of contents in the network, $n$ and only depends on the popularity index $s$.

**Theorem 3.** If the content request probability follows a Zipfian distribution and the users request independent contents in different time slots and the caching policy for network users is such that the cache content $c_{i,j} \in C_i$ is the content stored by node $i$ in its $j^{th}$ location of the cache, then the probability of finding a set of $k$ users $N_b = \{N_{b_1}, N_{b_2}, \ldots, N_{b_k}\} \subseteq N$ for which a single index code can be used to transmit the requested content $r_{b_i}$ to $N_{b_i}$ for $1 \leq i \leq k$ is independent of the total number of contents in the network.$^5$

Proof: The probability that a specific set of users $\{N_{b_1}, N_{b_2}, \ldots, N_{b_k}\}$ form a clique of size $k$ is given by

$$P_{b_1,b_2,\ldots,b_k} = \Pr[r_{b_i} \in C_{b_j} \text{ for } 1 \leq \forall i,j \leq k, j \neq i].$$

(8)

Assuming that the users are requesting contents independently of each other, this probability can be simplified as

$$P_{b_1,b_2,\ldots,b_k} = \prod_{i=1}^{k} \prod_{j=1,j\neq i}^{k} \Pr[r_{b_i} \in C_{b_j}].$$

(9)

Since the contents that are requested by each user in different time slots are independent of each other, we have

$$\Pr[r_{b_i} \in C_{b_j}] = 1 - \Pr[r_{b_i} \notin C_{b_j}] = 1 - \prod_{l=1}^{\delta} \Pr[r_{b_i} \neq c_{b_{j,l}}].$$

(10)

Notice that we assume a Zipfian distribution for the content request in the ICN network, $\Pr[r_{b_i} = c_{b_{j,l}}]$ will be equal to the probability that user $N_{b_j}$ has requested content with index $r_{b_i}$ and cached it in $l^{th}$ location of the cache. Hence

$$\Pr[r_{b_i} = c_{b_{j,l}}] = \frac{r_{b_i}^{-s}}{H_{n,s}}$$

(11)

$^5$This theorem is proved assuming Least Frequently Used (LRU) caching policy for the nodes.
Therefore, combining equations (11) and (10), we arrive at
\[
\Pr[r_{b_i} \in C_{b_j}] = 1 - \left(1 - \frac{r_{b_i} - s}{H_{n,s}}\right)^\delta. \tag{12}
\]
Equation (9) can be simplified as
\[
P_{b_1,b_2,\ldots,b_k} = \prod_{i=1}^{k} \left(1 - \left(1 - \frac{r_{b_i} - s}{H_{n,s}}\right)^\delta\right)^{k-1}. \tag{13}
\]
In order to have a clique of size \(k\), we need to include the possibility over all \(\binom{m}{k}\) groups of \(k\) users so the probability of having a clique of size \(k\) (which we denote by \(P\)) should be summed up over all of these choices. Therefore,
\[
P = \sum_{b_1,b_2,\ldots,b_k \subseteq N} P_{b_1,b_2,\ldots,b_k} = \sum_{b_1,b_2,\ldots,b_k \subseteq N} \prod_{i=1}^{k} \left(1 - \left(1 - \frac{r_{b_i} - s}{H_{n,s}}\right)^\delta\right)^{k-1}. \tag{14}
\]
Note that for any \(0 \leq x \leq 1\) and positive integer \(\delta\) we have
\[
1 - (1 - x)^\delta \geq x. \tag{15}
\]
Hence,
\[
P \geq \sum_{b_1,b_2,\ldots,b_k \subseteq N} \prod_{i=1}^{k} \left(\frac{r_{b_i} - s}{H_{n,s}}\right)^{k-1} = \sum_{b_1,b_2,\ldots,b_k \subseteq N} \prod_{i=1}^{k} \frac{r_{b_i} - s(k-1)}{H_{n,s}^{k-1}}. \tag{16}
\]
In order to simplify this expression, we use the elementary symmetric polynomial notation. If we have a vector \(V_m = (v_1, v_2, \ldots, v_m)\) of length \(m\), then the \(k\)-th degree elementary symmetric polynomial of these variables is denoted as
\[
\sigma_k(V_m) = \sigma_k(v_1, \ldots, v_m) = \sum_{1 \leq i_1 < i_2 < \ldots < i_k \leq m} v_{i_1} \ldots v_{i_k}. \tag{17}
\]
Using this notation and by defining \(Y_m = (r_1^{-s(k-1)}, r_2^{-s(k-1)}, \ldots, r_m^{-s(k-1)})\), we can write
\[
P \geq \frac{\sigma_k(Y_m)}{H_{n,s}^{k-1}}. \tag{18}
\]
Notice that since the content request probability follows a Zipfian distribution, we have
\[
\Pr[r_j \leq h] = \frac{H_{h,s}}{H_{n,s}}. \tag{19}
\]
Therefore, for a specific group of users \(N_{b_1}, N_{b_2}, \ldots, N_{b_k}\), the probability that they all request contents from the top \(h\) most popular contents is given by
\[
\Pr[r_{b_1} \leq h, \ldots, r_{b_k} \leq h] = \prod_{j=1}^{k} \Pr[r_{b_j} \leq h] = \left(\frac{H_{h,s}}{H_{n,s}}\right)^k. \tag{20}
\]
Using lemma 1, we can verify that for values of \( h = \Theta(1) \) with respect to \( n \), the ratio \( \frac{H_{h,s}}{H_{n,s}} \) can be very close to one. This, along with the fact that \( m \) is most likely much larger than \( k \), means that with a very high probability, for each set of users \( \{N_{b_1}, N_{b_2}, ..., N_{b_k}\} \), the requests come only from the \( h \) most popular contents. This implies that with a very high probability, 

\[
\sigma_k(Y_m) \geq \left( \binom{m}{k} \right) h^{-ks(k-1)}.
\]

Also, notice that

\[
H_{n,s} < H_{\infty,s} = \zeta(s) < \infty. \tag{21}
\]

Therefore, with a very high probability, we obtain the lower bound of (18) as

\[
P \geq \left( \binom{m}{k} \right) \left( \frac{h^{-ks}}{\zeta(s)} \right)^{k-1}. \tag{22}
\]

This lower bound does not depend on \( n \) and only depends on \( m, h, s \) and \( k \). The result means that regardless of the number of contents in the network, there is always a constant lower bound for the probability of finding a clique of size \( k \). This implies that MIC (or IC) can be used in large networks.

V. ASYMPTOTIC INDEX CODING GAIN

Both optimal and approximate solutions [7], [19] for the general index coding problem are NP-hard problems. Some efficient heuristic algorithms for the index coding problem were proposed [13] which can provide near optimal solutions. In some of these heuristic algorithms, the authors reduce the index coding problem to the graph coloring problem.

Notice that every clique in the dependency graph of a specific index coding problem, can be satisfied with only one transmission which is a linear combination of all packets that the users corresponding to the clique nodes are requesting. Therefore, solving the clique partitioning problem, which is the problem of finding a clique cover of minimum size for a graph [17], yields a simple index coding solution and therefore, the minimum number of cliques required to cover a graph can be regarded as an upper bound on the minimum number of index codes required to satisfy the users (known as the index coding rate). Since lower index coding rates translate into higher values of transmission savings\(^6\) as discussed in [12], the number of transmission savings found in the clique partitioning problem is in fact a lower bound on the total number of transmission savings found from the optimal index coding scheme.

\(^6\)In a dependency graph of \( n \) nodes with the index coding rate of \( k \), the number of saved transmissions is \( n - k \).
On the other hand, solving the clique partitioning problem for any graph \( G(V, E) \) is equivalent to solving the graph coloring problem for the complement graph \( \bar{G}(V, \bar{E}) \) which is a graph on the same set of vertices \( V \) but containing only the edges that are not present in \( E \). This is true because every clique in the dependency graph, gives rise to an independent set in the complement graph. Therefore, if we have a clique partitioning of size \( k \) in the dependency graph, we have \( k \) distinct independent sets in the complement graph. In other words, the chromatic number of the complement graph is \( k \).

This leads us to consider the complement graph of the dependency graph of an index coding problem. This graph is also referred [18] to as conflict graph. The above argument allows us to use the rich literature on the chromatic number of graphs to study the index coding problem. In fact, any graph coloring algorithm runs over the conflict graph can be directly used to obtain an achievable index code rate. If running such an algorithm over the conflict graph results in a coloring of size \( \chi \), this coloring gives rise to a clique cover of size \( \chi \) in the dependency graph and an index coding of rate \( \chi \) with a transmission saving of \( n - \chi \) which is a lower bound for the total number of transmission savings using the optimal index code[7].

Considering the chromatic number of the conflict graph, we can find a lower bound on the asymptotic index coding gain[8] in information centric networks. To do so, we use the following theorem from [8],

**Theorem 4.** For a fixed probability \( p, 0 < p < 1 \), almost every random graph \( G_{n,p} \) (a graph with \( n \) nodes and the edge presence probability of \( p \)) has chromatic number,

\[
\chi_{G_{n,p}} = -\left( \frac{1}{2} + o(1) \right) \log(1 - p) \frac{n}{\log n} \tag{23}
\]

Now assume that an endpoint router is serving \( m \) users where \( m \) is a large number. As discussed before, the index coding gain is lower bounded by \( m - \chi \) where \( \chi \) is the chromatic number of the conflict graph. However, notice that on average the chromatic number of our

---

[7] Notice that since the optimal index coding rate is upper bounded by the size of the minimum clique cover (which is equal to the chromatic number of the conflict graph), the value of transmission savings that we can achieve using the optimal index code is lower bounded by \( n - \chi \).

[8] Index coding gain is defined as the number of saved transmissions when using index coding.
non-deterministic conflict graph is upper bounded by the chromatic number of a random graph with edge existence probability of $1 - p_0$. Therefore, the index coding gain is lower bounded by $m - \chi_{G_{m,1-p_0}}$. Since $1 - p_0$ is fixed, theorem 4 implies that the chromatic number of the conflict graph is equal to

$$\chi_{G_{m,1-p_0}} = -\left(\frac{1}{2} + o(1)\right) \log p_0 \frac{m}{\log m} \quad (24)$$

Since the index coding gain is equal to $m - \chi_{G_{m,1-p_0}}$, we have proved the following lemma.

**Lemma 3.** Using a graph coloring-based algorithm, in a network with $m$ nodes we can achieve an index coding gain of

$$IC_G = m + \left(\frac{1}{2} + o(1)\right) \log p_0 \frac{m}{\log m}. \quad (25)$$

In the next theorem, we will prove that in a perfect tree network structure, in line with the experimental results in [16], graph coloring based algorithms can be near optimal in terms of index coding gain when caching is done at the edges of the network.

**Theorem 5.** Assuming a perfect tree network structure with depth $\eta$ and expansion degree $d_e$ (like in figure 3 with $\eta = 3$ and $d_e = 2$) and assuming each edge router $R_{\eta i}$ for $i = 1, 2, ..., (d_e)^\eta$ serves $m$ wireless mobile users, then caching in depth $i$ of the network for any $i = 0, 1, ..., \eta - 1$ and index coding using a graph coloring based algorithm does not have a significant benefit in terms of index coding gain over caching on the edge (depth $\eta$) of network for large enough values of $m, n$.

**Proof:** We can apply lemma 3 to prove this theorem. Notice that in depth $i$ of the network for $i = 0, 1, ..., \eta$, we have $(d_e)^i$ routers each responsible for all the traffic that is going to or from the downlink routers. Any depth $i$ router has $(d_e)^{\eta-i}$ edge routers in the set of its downlink routers. Therefore, such a router is connecting $m \times (d_e)^{\eta-i}$ wireless mobile users to each other and to the outside world. Lemma 3 implies that in such a network, the index coding gain for

\footnote{This is true because in the dependency graph, the probability of edge existence for each edge is at least $p_0$ which implies that the probability of edge existence in the conflict graph is at most $1 - p_0$. Therefore, the conflict graph on average has less edges compared to a random graph with edge existence probability of $1 - p_0$ and so its chromatic number cannot be greater than the chromatic number of $G_{m,1-p_0}$.}
each router $R_{ij}$ where $1 \leq j \leq (d_e)^i$ is

$$IC_G(R_{ij}) = m d_e^{\eta-i} + \left(\frac{1}{2} + o(1)\right) \log p_0 \frac{m d_e^{\eta-i}}{\log(m d_e^{\eta-i})}$$

(26)

The result implies that considering all the routers in depth $i$, the aggregate index coding gain in depth $i$ is equal to

$$IC_G(i) = m d_e^{\eta} \left(1 + \frac{(1 + o(1)) \log p_0}{\log m + (\eta - i) \log d_e}\right).$$

(27)

Since $\log p_0$ is negative, $IC_G(i)$ is decreasing in $i$. Therefore, the maximum index coding gain using a graph coloring based algorithm happens in depth 0 and the minimum happens in depth $\eta$. The ratio of the maximum to minimum index coding gain is then equal to

$$\frac{IC_G(0)}{IC_G(\eta)} = \frac{1 + \frac{(1 + o(1)) \log p_0}{\log m + \eta \log d_e}}{1 + \frac{(1 + o(1)) \log p_0}{\log m}} = \frac{\log m}{\log m + \eta \log d_e} \left(1 + \frac{\eta \log d_e}{\log m + \eta \log d_e + o(1) \log p_0}\right).$$

(28)

This equation asymptotically tends to 1 when $m$ (and obviously $n$) tends to infinity. This means that maximum index coding gain in such a network (happening at depth 0 of the network) is not much higher than the minimum index coding gain which happens at the edge of network.

---

**VI. SIMULATIONS**

In this section, we compare the performance of the new architecture to the original ICN architecture. Figure 4 shows the probability that a specific content that the users are requesting be available in the endpoint router. As this plot suggests, the probability of content availability
in the endpoint router approaches one as the Zipfian parameter is increased. Notice that this probability goes to one regardless of the number of available contents in the network, number of users and other factors. However, figure 4 suggests that this probability is slightly higher when the number of contents is smaller and/or the router has a larger cache size. The fact that many requested contents have been already cached in the router implies that some nodes also store them in their caches. Therefore, we predict that the introduction of MIC to ICN architecture will be very useful. For these simulations, \( R \) denotes the size of endpoint router cache and \( U \) denotes the size of user’s cache.

![Probability of content availability in endpoint router versus the Zipfian parameter](image)

Fig. 4. Probability of requesting a content that is already available in the edge router cache versus the Zipfian parameter \( s \). \( R, U, l \) and \( k \) denote the size of endpoint router cache, the size of user’s cache, the number of users served by each endpoint router and the total number of contents in the network, respectively.

Figure 5 shows the simulation results for five different set of parameters. In this figure, we have plotted the average packets sent in each transmission. We have assumed that the users are requesting contents based on a Poisson distribution with an average rate of \( \frac{1}{m} \). This way we can ensure that the average total request rate is one and we are efficiently using the time resource without generating unstable queues. In each time slot, the endpoint router chooses to send to as many users as possible using index coding. In fact, to find the actual benefit that we can
achieve by using index coding, we need to find the optimal rate for the index coding problem. For instance, in each time slot, if we can find a clique of size $k$ in the dependency graph, we can save $k - 1$ transmissions by transmitting a simple index code to all the specified users. For any cycle of size $k$, we only need to transmit $k - 1$ encoded packets. However, finding a clique of maximum size in the dependency graph or the optimal index coding rate is an NP-hard problem.

For the purpose of our simulation, however, we used a very simple heuristic algorithm to count the number of cliques and cycles of maximum size 4. We first search for all cliques and then, look for cycles. Even with this simple algorithm, we were able to show that the index coding can double the average number of packets per transmission in each time slot for certain values of the Zipfian parameter. Clearly, the optimal index coding rate is larger than the results obtained by our simple algorithm.

When $s$ is a small value, then the distribution of content request is close to uniform distribution. Under this condition, the index coding dependency graph is very sparse because there is a small probability that a user requests a content that is already available in another user's cache. Clearly, there is no benefit for using MIC in this case. Similarly, when $s$ is a very large number, most users are asking for the same content and therefore, the router broadcasts the content to all of them which is equivalent of an average of one content per transmission. The main benefit of index coding happens for medium values of $s$ between 0.5 and 2 which is usually the case in practical networks. Note that an ICN with no MIC, will have always one content per transmission which is the baseline.

VII. Conclusion

This paper introduced a new architecture for ICN based on an approach for broadcast communications called index coding (IC). In order to use index coding in multihop communications, we introduced Modified Index Coding (MIC) approach. MIC is more general than original IC problem which can be applied to networks with nodes communicating using multi-hop communications or in wired networks. We also introduced a hybrid caching scheme that consists of a central caching at the router and distributed caching at the nodes. The purpose of central caching is similar to the original concept of caching in ICN but we transmit encoded messages instead of the data itself. The purpose of distributed caching is to have replicas of the popular messages so that when we transmit new requested messages, we can combine multiple messages in order
to serve multiple users with a single transmission. In this paper, we proved that for perfect tree network structures, index coding in the edge of network is almost as good as index coding inside the network.

There are many research problems that were not addressed in this paper. For example, security, content routing and caching policies for these two types of caching are some problems that can be investigated in the future.

**REFERENCES**


