Multihop Caching-Aided Coded Multicasting for the Next Generation of Cellular Networks

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

Abstract—Next generation of cellular networks deploying wireless distributed femtocaching infrastructure proposed by Golrezaei et. al. are studied. By taking advantage of multihop communications in each cell, the number of required femtocaching helpers is significantly reduced. This reduction is achieved by using the underutilized storage and communication capabilities in the User Terminals (UTs), which results in reducing the deployment costs of distributed femtocaches.

A multihop index coding technique is proposed to code the cached contents in helpers to achieve order optimal capacity gains. As an example, we consider a wireless cellular system in which contents have a popularity distribution and demonstrate that our approach can replace many unicast communications with multicast communication. We will prove that simple heuristic linear index code algorithms based on graph coloring can achieve order optimal capacity under Zipfian content popularity distribution.

Index Terms—Cellular Networks, Index Code,Femtocache.

I. INTRODUCTION

With the recent pervasive surge in using wireless devices for video and high speed data transfer, it seems eminent that the current wireless cellular networks cannot be a robust solution to the ever-increasing wireless bandwidth utilization demand. Researchers have been recently focused on laying down the fundamental grounds for future cellular networks to overcome such problems.

Deploying home size base stations is proposed as a solution in [7]. The solution in [7] is based on the idea of femtocells in which many small cells are deployed throughout the network to cover the entire network. Deploying many small cells in the network with reliable backhaul links requires significant capital investment. Therefore, Golrezaei et. al. [14] proposed femtocaching as an alternate solution to overcome this problem. In this approach, in every cell along with the main base station, smaller base stations with low-bandwidth backhaul links and high storage capabilities are deployed to create a wireless distributed caching infrastructure. These small base stations which are called caching helpers (or simply helpers), will store popular contents in their caches and use their caches to serve User Terminal (UT) requests. Therefore, in networks with high content reuse, the backhaul utilization will be significantly reduced using this approach. If the requested content is not available in the helper’s cache, UTs can still download the content from their low-bandwidth backhaul links to the base station. The proposed technique in [14] requires deployment of large number of femtocaches in order to cover all the nodes in the network.

On the other hand, it is well-known that web content request popularity follows Zipfian-like distributions [9]. This content popularity distribution implies that few popular contents are widely requested by the network UTs. We assume UTs store their requested contents and therefore, helpers can multicast multiple requests by taking advantage of coding to reduce the total number of transmissions.

In this paper, we propose to use index coding to code the contents in helpers before transmission. Index coding is a source coding technique proposed in [3] which takes advantage of UTs’ side information in broadcast channels to minimize the required number of transmissions. In index coding, the source (e.g. base station) designs codes based on the side information stored in requesting nodes. The coded information is broadcasted to the UTs that use the information together with their cached contents to decode the desired content. Index coding improves bandwidth utilization by minimizing the number of required transmissions. We propose to extend index coding approach from broadcast one-hop communication to multihop scenarios which will be explained in details.

Our main motivation to use index coding is the high storage availability in UTs to improve the achievable throughput of the future wireless cellular networks. Current improvements in high density storage systems has made it possible to have personal devices with Terabytes of storage capability. This ever-increasing trend provides future personal wireless devices with huge under-utilized storage capabilities. Future wireless devices can use their storage capability to store the contents that they have already requested. In an index coding setting, many UTs that are requesting different contents can receive a coded file which is multicasted to them and then each UT uses the information in its cache to decode its requested content from the received coded file. There is an important equivalence between index coding and network coding as stated in [10], [11] and therefore, the results in this paper can also be stated based on a network coding terminology.

We will prove that index coding can be efficiently used to encode the contents by helpers under a Zipfian distribution model. The encoded contents can be relayed through multiple hops to all the UTs being served by that helper.

The optimal index coding solution is an NP-Hard problem [18]. However, we will show that even using linear index codes can result in order optimal capacity gains in these networks. We believe that this coding technique can serve as
a complement to the solution proposed in [14]. As clearly
articulated in [21], in any caching problem we are faced with
two phases of cache placement and cache delivery. While
[14] proposes efficient cache placement algorithms, we will
be focusing on efficient delivery methods for their solution.
We will show that the problem of delivery in [14] can be
efficiently addressed by using index coding in the helpers.

Recent discussions on standards for future 5G cellular net-
works are focused on providing high bandwidth for Device-to-
Device communications (D2D). Examples of such approaches
are the IEEE 802.11ad standard (up to 60GHz [2]) and the
millimeter-wave proposal which can potentially enable up to
300GHz of bandwidth for D2D communications [1], [4]. This
potential abundant D2D bandwidth can be utilized to relay the
coded contents inside an ad hoc network which is being served
by a helper. It is indeed such excessive storage and bandwidth
capabilities of future wireless systems that make our solution
feasible.

Deploying many femtocaching helpers is not economically
efficient. On the other hand, since UTs have significant D2D
capabilities, they can efficiently participate in content delivery
through multihop D2D communications. In our proposal, we
suggest to deploy few helpers which can deliver the contents
to neighboring UTs through multihop D2D communications.
This can reduce the network deployment and maintenance
costs.

The rest of this paper is organized as follows. Section
II reviews the related works and section III describes the
proposed network model that is similar to [14] with the
addition of using multihop communications and index coding.
In section IV we will explain the scaling laws of capacity
improvement using index coding and relaying. Section V
demonstrates that index coding algorithms can achieve order
optimal gains. Section VI describes the simulation results and
the paper is concluded in section VII.

II. RELATED WORK

The problem of caching when a server is transmitting
contents to clients was studied in [21] from an information
theoretic point of view. The authors introduced two phases
of cache placement and content delivery. For a femtocaching
solution, efficient cache placement algorithms are proposed in
[14]. In this paper, we focus on efficient content delivery
algorithms through index coding and multihop D2D commu-
nications for a femtocaching solution.

There has been significant research on index coding since it
was proposed in [3]. The practical implementation of index
coding for wireless applications was proposed in [8] by
proposing cycle counting methods. A dynamic index coding
solution for wireless broadcast channels is proposed in [23].
In [23], a wireless broadcast station is considered and a simple
set of codes based on cycles in the dependency graph is
provided. They show the optimality of these codes for a class
of broadcast relay problems. In this paper, we prove that codes
based on cycles can achieve order optimal capacity gains in
networks with Zipfian content request distribution.

Approximating index coding solution is proved [3] to be
an NP-Hard problem. However, efficient heuristics has been
proposed in [9] which are based on well-known graph coloring
algorithms. Other references like [11] and [10] have studied
the connections and equivalence between index coding and
network coding. Tran et al. [24] studied a single hop wireless
link from a network coding approach and showed similar
results to index coding. Ji et al. [16] studied theoretical limits
of caching in D2D communication networks.

Study of coding techniques in networks with high content
reuse has recently attracted the attention of researchers. Mont-
 Petit et al. [22] studied the applications of network coding in
Information-Centric Networks (ICN). Wu et al. [25] studied
network coding in Content Centric Networks (CCN) which
is an implementation of ICN. Leong et al. [19] proposed a
linear programming formulation to deliver contents optimally
in today’s IP based Content Delivery Networks (CDN) using
network coding. Llorca et al. [20] proposed a network-coded
caching-aided multicasting technique for efficient content de-
Iery in CDNs. This paper extends Index Coding to multihop
communication and shows that linear codes can achieve order
optimal capacity gains in such networks.

III. NETWORK MODEL

We assume a network model for future wireless cellular
networks in which femtocaching helpers with significant stor-
age capabilities are deployed throughout the network to assist
the efficient delivery of contents to UTs through reliable D2D
communications. Such helpers are characterized with low rate
backhaul links which can be wired or wireless. They will also
have localized, high-bandwidth communication capabilities.
With their significant storage capacity, in a network with high
content reuse, many of the content requests for the UTs inside
a cell can be satisfied directly by the helpers.

Each helper is serving a wireless ad-hoc network in which
the UTs are utilizing high bandwidth D2D communication
techniques such as millimeter wave and IEEE 802.11ad tech-
nologies. This high bandwidth D2D communication enables
the UTs to relay data from a nearby helper to all the UTs that
are within transmission range. We assume that the path lifetime
between the helper and UTs is longer than the time required to
transmit the content. For large files and when UTs are moving
fast, one solution is to divide the content into smaller files and
treat each file separately. The paper assumes that the network
is connected which can be justified by the large number of UTs
that will be available in the future wireless cellular networks.

Further, we take advantage of index coding and the side
information cached in UTs to significantly reduce the required
number of helpers and consequently, reduce the infrastructure
maintenance and deployment costs. To demonstrate the effec-
tiveness of using multihop communications, we consider simi-
lar assumptions as [14] with a macro base station placed in the
center of a cell with radius 400 meters serving 1000 UTs and a
transmission range of 100 meters [1] for D2D communication.
As shown in Figure [1] with only 4 helpers uniformly located
in the cell, 100% and 80% of nodes are covered with 3 and 2
hop communications respectively. Covering all the UTs in the
same cell with only one hop communication requires up to 27
helpers [14]. Clearly, for a vehicle or a mobile UT operating in
1. An encoding function \( E \) mapping inputs in \( \{0, 1\}^m \) to codewords and
2. A set of decoding functions \( D_1, D_2, \ldots, D_n \) such that
\[
D_i(E(M)), M_j = M_r, \quad \text{for } i = 1, 2, \ldots, n.
\]

In the above definition, the length of the index code \( l \) denotes the number of required transmissions to satisfy all the content requests of the \( n \) UTs. The encoding function \( E \) is applied to the contents by the helper and the decoding functions \( D_i \)s are applied individually by UTs to decode their desired contents from the encoded content using their cached information.

Figure 2 shows a helper \( H \) serving 6 UTs \( N_1, N_2, N_3, N_4, N_5, \) and \( N_6 \). Let’s assume UTs \( N_1, N_2 \) and \( N_5 \) request contents \( M_2, M_1 \) and \( M_4 \) while storing \( \{M_1, M_4\} \), \( \{M_3, M_4\} \) and \( \{M_1, M_3\} \) respectively. Using index coding requires 3 channel usages while without index coding, we need 5 channel usages. This is true since using index coding, the helper creates the XOR combination of contents \( M_1, M_3 \) and \( M_4 \) as \( M_1 \oplus M_3 \oplus M_4 \) and broadcasts this coded content to its neighboring UTs. Either of \( N_1, N_2 \) can immediately reconstruct their requested content from this coded content. For instance, \( N_1 \) which is requesting \( M_3 \) can decode its requested content by using it’s cached information and XOR operation on the encoded message to retrieve the requested content \( M_3 \), i.e., \((M_1 \oplus M_3 \oplus M_4) \oplus M_1 \oplus M_4 = M_3 \). After \( N_2 \) receives \( M_1 \oplus M_3 \oplus M_4 \), it relays it to node \( N_4 \) which simply broadcasts it to \( N_5 \). UT \( N_5 \) will again use XOR operation to decode it’s desired content \( M_4 \) by adding it’s cached contents to the coded content that it has received, i.e., \((M_1 \oplus M_3 \oplus M_4) \oplus M_1 \oplus M_3 \oplus M_4 = M_4 \). Therefore, only 3 transmissions are needed.

To satisfy the content requests through multihop D2D without index coding, 5 transmissions are needed as the helper should transmit \( M_3 \) to \( N_1, N_2 \) to \( N_4, N_2 \), \( N_4 \) to \( N_2 \), and \( N_2 \) relays \( M_4 \) to \( N_4 \) and \( N_4 \) relays it to \( N_5 \).

We assume a Zipfian distribution with parameter \( s > 1 \) for content popularity distribution in the network. This means that the probability that UT \( N_i \) requests any content with index \( r_i \) at any time instant is given by

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

where \( H_{m,s} = \sum_{j=1}^{m} \frac{1}{j^s} \) denotes the \( m^{th} \) generalized harmonic number with parameter \( s \).

Remark 1. This paper only focuses on Zipfian content request probability with parameter \( s > 1 \) which is a non-heavy-tailed probability distribution. Our results are correct for any type of non-heavy-tailed probability distribution and the extension to heavy-tailed probability distribution is the subject of future work.

Dependency graph is a useful analytical tool \( [3, 8, 11] \) that is widely used in index coding literature.

Definition 2. (Dependency Graph): Given an instance of an index coding problem, the dependency graph \( \mathcal{G}(V, E) \) is defined as

- each UT \( N_i \) corresponds to a vertex in \( V \), \( N_i \in V \),
- there is a directed edge in \( E \) from \( N_i \) to \( N_j \) if and only if \( N_i \) is requesting a content that is already cached in \( N_j \).

This dependency graph does not represent the actual physical links between UTs in the network. This is a virtual graph in which each edge represents the connection between a UT
that is requesting a content and a UT that caches this content. As discussed in [8], [23], every cycle in the dependency graph is representative of a connection between UTs and it can save one transmission. For every clique in the dependency graph, all the requesting UTs in the clique can be satisfied by a simple linear XOR index code. The complement of dependency graph is called conflict graph. This graph is of significant interest since any clique in the dependency graph gives rise to an independent set in the conflict graph. Therefore, well-known graph coloring algorithms over conflict graph can be used to find simple linear XOR index codes. The dependency and conflict graphs in our network are random directed graphs. In the next section, we will use the properties of these graphs to find the capacity gains and propose simple index coding solutions.

To prove our results in this paper, we have used Least Recently Used (LRU) or Least Frequently Used (LFU) cache policies. Similar results can be produced for other caching policies. LRU caching policy assumes that most recently requested contents are kept in the cache. In LRU caching, in each time slot the content that is requested in the previous time slot is stored in the first cache location. If this content was already available in the cache, it is moved from that location to the first cache location. If the content was not available, then the content that is least recently used is discarded and the most recently requested content is cached in the first cache location. Any other content is relocated to a new cache location such that the contents appear in the order that they have been requested.

In LFU caching policy, the contents are cached based on their request frequency. Highly popular contents are stored in the first locations of the cache and contents with lower request frequency are cached in the bottom locations of the cache.

Computing the probability of having content with index $i_r$ in cache $C_j$, $Pr[r_i \in C_j]$, turns out to be complicated for LRU or LFU caching policies. A simple lower bound on this probability for LRU can be found by noticing that $Pr[r_i \in C_j]$ is greater than the probability that UT $N_j$ has requested the content with index $i_r$ in the most recent time slot and therefore it is located at the top of the cache. This lower bound can be derived using equation (1).

$$Pr[r_i \in C_j] = Pr[c_{j1} = r_i] + \sum_{l=2}^{\delta} Pr[c_{jl} = r_i] \geq Pr[c_{j1} = r_i] = \frac{r_i^{-s}}{H_{m,s}}.$$ (2)

To prove that the same lower bound holds for LFU, notice that $Pr[r_i \in C_j]$ is larger than the same probability when the cache size is $M = 1$. Therefore,

$$Pr[r_i \in C_j] \geq Pr[r_i \in C_j \mid M = 1] = Pr[c_{j1} = r_i] = \frac{r_i^{-s}}{H_{m,s}}.$$ (3)

In the subsequent sections, we will use these lower bounds to prove our results.

In this paper, we will state our results in terms of order bounds. To avoid any confusion, we use the following order notations [17]. We denote $f(n) = O(g(n))$ if there exist $c > 0$ and $n_0 > 0$ such that $f(n) \leq cg(n)$ for all $n \geq n_0$, $f(n) = \Omega(g(n))$ if $g(n) = O(f(n))$, and $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

### IV. Order Optimal Capacity Gain

In this section, we will prove that index coding can significantly decrease the number of transmissions in the network. We will specifically use the Zipfian content distribution in the underlying content distribution network. To do so, we will first state and prove the following lemma.

**Lemma 1.** Let’s consider a Zipfian content distribution with parameter $s > 1$ and parameter $h_\epsilon = c^{\frac{1}{s+1}}$ where $0 < \epsilon < 1$. For every $i$ with popularity index $r_i$ that is less than $h_\epsilon$, the request probability is at least $1 - \epsilon$.

**Proof:** Based on the Zipfian distribution assumption and equation (1), the probability that the requested content has a popularity of at most $h_\epsilon$ is equal to

$$Pr[r_j \leq h_\epsilon] = \frac{H_{h_\epsilon,s}}{H_{m,s}}.$$ (4)

In order to satisfy $Pr[r_j \leq h_\epsilon] \geq 1 - \epsilon$, we should have

$$\epsilon \geq 1 - Pr[r_j \leq h_\epsilon] = 1 - \frac{H_{h_\epsilon,s}}{H_{m,s}} = \frac{1}{H_{m,s}} \sum_{j=h_\epsilon+1}^{m} j^{-s}.$$ (5)

If we have

$$\epsilon \geq \frac{1}{H_{m,s}} \sum_{i=1}^{\lfloor \frac{n}{m} \rfloor} \frac{(i+1)h_\epsilon}{\sum_{j=ih_\epsilon+1}^{m} j^{-s}},$$ (6)

then the inequality in (5) will certainly hold since the right hand side in (6) is larger than the right hand side in (5). Note that, for $ih_\epsilon + 1 \leq j \leq (i+1)h_\epsilon$ we have $j^{-s} < (ih_\epsilon)^{-s}$. Hence,

$$\sum_{j=ih_\epsilon+1}^{(i+1)h_\epsilon} j^{-s} < \sum_{j=ih_\epsilon+1}^{(i+1)h_\epsilon} (ih_\epsilon)^{-s} = h_\epsilon(ih_\epsilon)^{-s} = i^{-s}h_\epsilon^{-s}.$$ (7)

This means that if

$$\epsilon \geq h_\epsilon^{-s} \frac{1}{H_{m,s}} \sum_{i=1}^{\lfloor \frac{n}{m} \rfloor} i^{-s},$$ (8)

then (6) holds since the right hand side of inequality (6) is smaller than the right hand side of inequality (5) as shown by (7). Notice that since,

$$\frac{1}{H_{m,s}} \sum_{i=1}^{\lfloor \frac{n}{m} \rfloor} i^{-s} = \frac{\sum_{i=1}^{\lfloor \frac{n}{m} \rfloor} i^{-s}}{\sum_{i=1}^{\lfloor \frac{n}{m} \rfloor} i^{-s} + \sum_{i=\lfloor \frac{n}{m} \rfloor+1}^{\infty} i^{-s}} < 1,$$

if $h_\epsilon$ is chosen such that

$$\epsilon \geq h_\epsilon^{-s},$$ (9)
then all of the inequalities in equations (5), (6), (7) and (8) will be valid and hence $\Pr[r_j \leq h_e] \geq 1 - \epsilon$. Therefore, in order to have $\Pr[r_j \leq h_e] \geq 1 - \epsilon$, it is enough to choose $h_e$ such that (9) is valid. Hence, if $h_e$ is chosen to be at least equal to

$$h_e = \epsilon^{\frac{1}{1-\epsilon}},$$

(10)

then we have $\Pr[r_j \leq h_e] \geq 1 - \epsilon$. Notice that the choice of $h_e$ in (10) is such that it only depends on $\epsilon$ and is independent of $m$.

For instance for $s = 2$ and $\epsilon = 0.01$, $h_e$ can be chosen as 100 (regardless of the size of $m$). This implies that for a Zipfian distribution with $s = 2$, 100 highly popular contents among any large number of contents would account for 99% of the total content requests. Therefore, if $h_e$ is chosen as in equation (10), with a probability of at least $1 - \epsilon$ all content requests have popularity index of at most $h_e$. Now define $p_e$ as

$$p_e \triangleq \frac{h^{-s}_{e}}{H_{m,s}} = \frac{\epsilon^{\frac{s}{\epsilon-1}}}{H_{m,s}}.$$  

(11)

Based on above discussion, in our instance of index coding dependency graph, with a probability of at least $1 - \epsilon$ edges are present with a probability of at least $p_e$. We will discuss this in more details later in the proof for Theorem 1. Notice that for large values of $m$, $p_e$ is also independent of $m$ since in that case $H_{m,s} \approx \zeta(s)$ and $p_e$ only depends on $\epsilon$ and $s$.

As stated in (9), if we choose the right encoding vectors for any index coding problem, for any vertex disjoint cycle in the dependency graph we can save one transmission. Therefore, the number of vertex-disjoint cycle $c$ in the dependency graph can serve as a lower bound for the number of saved transmissions in any index coding problem. Number of vertex disjoint cycles is also used in [23] as a way of finding the lower bound for index coding gain. To count the number of vertex-disjoint cycles in our random dependency graph, we will use the following lemma originally proved as Theorem 1 in [12].

** Lemma 2.** Let $d > 1$ and $v \geq 24d$ be integers. Then any graph $G_{f(v,d)}$ with $v$ vertices and at least $f(v,d) = (2d-1)v - 2d^2 + d$ edges contains $d$ disjoint cycles or $2d - 1$ vertices of degree $v - 1$.

Note that the dependency graph is a directed graph and in order to use Lemma 2 we need to construct an undirected graph. Let's denote the directed and undirected random graphs on $n$ vertices and edge presence probability $p_e$ by $G(n,p_e)$ and $\overline{G}(n,p_e)$, respectively. In a directed graph, $\overline{G}(n,p_e)$, the probability that two vertices are connected by two opposite directed edges is $p_e^2$. Therefore, we can build an undirected graph $\overline{G}(n,p_e^2)$ with the same number of vertices and an edge between two vertices if there are two opposite directed edges in the directed graph $\overline{G}(n,p_e)$ between these two UTs. Hence, $\overline{G}(n,p_e)$ essentially contains a copy of $\overline{G}(n,p_e^2)$. Note that there are some edges between UTs in $\overline{G}(n,p_e)$ that do not appear in $\overline{G}(n,p_e^2)$. This fact was also observed in [15]. Therefore, a lower bound on the number of disjoint cycles for $\overline{G}(n,p_e^2)$ implies a lower bound on the number of disjoint cycles for $\overline{G}(n,p_e)$.

In the following theorems, we will use Lemma 2 to prove that using index coding to code the contents can be very efficient.

**Theorem 1.** Assume all UTs are utilizing LRU or LFU caching policies for a Zipfian content request distribution with parameter $s > 1$. Index coding can save $\Omega(np_e^2)$ transmissions for any helper serving $n$ UTs with a probability of at least $1 - \epsilon$ for any $0 < \epsilon < 1$.

**Proof:** Consider a Zipfian distribution with parameter $s > 1$ and let $0 < \epsilon < 1$ be fixed. The dependency graph $\overline{G}(V,E)$ in our problem is composed of $n$ vertices $N_1,N_2,...,N_n$ which correspond to the $n$ UTs that are served by a helper. Note that the existence of an edge in dependency graph depends on the probability that a UT is requesting a content and another UT 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 $(N_i,N_j)$ in $E$ is equal to the probability that content $r_i$ requested by $N_i$, is already cached in $N_j$. Therefore, with LRU or LFU caching policy assumption and using equations (2) and (3), we arrive at

$$\Pr[(N_i,N_j) \in E] = \Pr[r_i \in C_j] \geq \frac{r_i - s}{H_{m,s}}.$$  

(12)

Using Lemma 1 for any $0 < \epsilon < 1$, if $h_e$ is chosen as $h_e = \epsilon^{\frac{s}{1-\epsilon}}$, then with a probability of at least $1 - \epsilon$, any requested content has a popularity index $r_i$ less than $h_e$. This means that with a probability of at least $1 - \epsilon$, the edge presence probability in equation (12) can be lower bounded by $p_e$. Therefore, with a probability of at least $1 - \epsilon$, maximum number of vertex-disjoint cycles in our directed dependency graph $\overline{G}(V,E)$ can be lower bounded by the maximum number of vertex-disjoint cycles in an Erdős-Rényi random graph $\overline{G}(n,p_e)$ with $n$ vertices and edge presence probability $p_e$. Now we can use Lemma 2 and undirected graph $\overline{G}(n,p_e^2)$ to find a lower bound on the number of vertex disjoint cycles in $\overline{G}(n,p_e)$. This in turn, will give us a lower bound on the number of vertex-disjoint cycles in $\overline{G}(V,E)$.

Note that $\overline{G}(n,p_e^2)$ is an undirected Erdős-Rényi random graph on $n$ vertices and edge presence probability $p_e^2$. This graph has a maximum of $n(n-1)$ undirected edges. However, since every undirected edge in this graph exists with a probability of $p_e^2$, the expected value of the number of edges in graph $G(n,p_e^2)$ is $n(n-1)p_e^2$. This means that if $d$ in Lemma 3 with $v = n$ is chosen to be an integer such that

$$n(n-1)p_e^2 \geq (2d-1)n - 2d^2 + d,$$

(13)

then on average, $G(n,p_e^2)$ will have either $d$ disjoint cycles or $2d - 1$ vertices of degree $n - 1$. For the purpose of our

3This edge has no relationship with the actual physical link between two UTs.
paper we can easily verify that for large enough values of \( n \),
\[
d^* = \left\lceil \frac{np^2}{24} \right\rceil
\]
satisfies equation (14) (Notice that the condition \( 2d^* \leq n \) in Lemma 2 is also met). Therefore based on Lemma 2, the graph \( G(n, p^2) \) either has at least \( d^* = \left\lceil \frac{np^2}{24} \right\rceil \) disjoint cycles or \( 2d^* - 1 = 2\left\lceil \frac{np^2}{24} \right\rceil - 1 \) vertices with degree \( n - 1 \). As mentioned before, \( G(n, p_c) \) essentially contains a copy of \( G(n, p^2) \). Consequently, \( G(n, p_c) \) either has at least \( d^* = \left\lceil \frac{np^2}{24} \right\rceil \) disjoint cycles or \( 2d^* - 1 = 2\left\lceil \frac{np^2}{24} \right\rceil - 1 \) vertices with degree \( n - 1 \). The number of vertices in graph \( G(n, p_c) \) is \( n \). Therefore, the latter case gives rise to a situation where there are \( 2d^* - 1 = 2\left\lceil \frac{np^2}{24} \right\rceil - 1 \) vertices which are connected to any other vertex in \( G(n, p_c) \) through undirected edges. This condition results in having a clique of size \( 2\left\lceil \frac{np^2}{24} \right\rceil - 1 \) in \( G(n, p_c) \).

In summary, \( G(n, p_c) \) has either \( d^* \) disjoint cycles or it contains a clique of size \( 2d^* - 1 \). Hence, with a probability of at least \( 1 - \epsilon \), the dependency graph \( \tilde{G}(V, E) \) on average has either \( d^* \) disjoint cycles or it contains a clique of size \( 2d^* - 1 \). In either of these cases \( d^* = \left\lceil \frac{np^2}{24} \right\rceil \) transmissions can be saved using index coding. This proves the theorem.

**Theorem 2.** Index coding through cycle counting and clique partitioning can save \( \Theta(n) \) transmissions in a network with \( n \) UTs and Zipfian content request distribution with parameter \( s > 1 \).

**Proof:** In Theorem 1 we proved that in a network with Zipfian content request distribution and for a fixed \( 0 < \epsilon < 1 \), with a probability of at least \( 1 - \epsilon \), the dependency graph either has \( d^* = \left\lceil \frac{np^2}{24} \right\rceil \) disjoint cycles or it contains a clique of size \( 2d^* - 1 \). Consider the following situations:

1. The dependency graph has \( d^* = \left\lceil \frac{np^2}{24} \right\rceil \) disjoint cycles. In this case, for a fixed \( \epsilon, p_c \) is a constant which does not depend on \( n \). Hence, cycle counting can result in at least \( d^* = \left\lceil \frac{np^2}{24} \right\rceil = \Omega(n) \) transmission savings. This is a lower bound on the number of transmission savings. On the other hand, notice that the maximum number of vertex-disjoint cycles in any graph with \( n \) vertices cannot be greater than \( \frac{n^2}{2} \) as shown in Figure 3. Therefore, the maximum number of transmission savings using cycle counting is \( \frac{n^2}{2} = O(n) \). This is an upper bound on the number of saved transmissions. Since the order of upper and lower bounds are the same, it can be concluded that the number of saved transmissions scales as \( \Theta(n) \).

2. The dependency graph contains a clique of size \( k^* = 2d^* - 1 = 2\left\lceil \frac{np^2}{24} \right\rceil - 1 \). Through clique partitioning we will be able to save at least \( k^* - 1 \) transmissions by sending only one transmission. Hence, through clique partitioning, we will be able to save at least \( k^* - 1 = \Omega\left(\frac{np^2}{12}\right) = \Omega(n) \) transmissions by sending only one transmission to the UTs forming that specific clique. This is a lower bound on the number of saved transmissions. On the other hand, if the dependency graph is a perfectly complete graph on \( n \) nodes which means that every

We can further prove that many properties of the dependency graph are independent of the total number of contents and only depends on 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 all the requests in this clique can be satisfied with one multicast transmission. The following theorem proves that the probability of existence of a clique of size \( k \) is lower bounded by a value which is independent of the total number of contents in the network, \( m \), and only depends on the popularity index \( s \).

**Theorem 3.** If LRU or LFU caching policy is used and the content request probability is Zipfian distribution, then the probability of finding a set of \( k \) UTs \( N_b = \{N_{b_1}, N_{b_2}, ..., N_{b_k}\} \subseteq \mathbb{N} \) for which a single linear index code (XOR operation) can be used to transmit the requested content \( r_{b_i} \) to \( N_{b_i} \) for \( 1 \leq i \leq k \) can be lower bounded by a value that with a probability close to one is independent of the total number of contents in the network.

**Proof:** The probability that a specific set of UTs \( \{N_{b_1}, N_{b_2}, ..., N_{b_k}\} \) form a clique of size \( k \) is

\[
\Pr[r_{b_i} \in C_{b_j} , \text{ for } 1 \leq i,j \leq k, j \neq i] = \prod_{i=1}^{k} \prod_{j=1, j \neq i}^{k} \Pr[r_{b_i} \in C_{b_j}].
\]

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

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

Using equations (2) and (3), we arrive at

\[
\Pr[r_{b_i} \in C_{b_j}] \geq \frac{r_{b_i}^{-s}}{H_{m,s}}.
\]

Equation (15) can be lower bounded as

\[
P_{b_1, b_2, ..., b_k} \geq \prod_{i=1}^{k} \left( \frac{r_{b_i}^{-s}}{H_{m,s}} \right)^{-k-1}.
\]
The probability to have a clique of size \( k \) is computed by considering all \( \binom{n}{k} \) groups of \( k \) UTs. Hence, the probability of having a clique of size \( k \) denoted by \( P_k \) is given by

\[
P_k = \sum_{b_1, b_2, \ldots, b_k \subseteq N} P_{b_1, b_2, \ldots, b_k} \geq \sum_{b_1, b_2, \ldots, b_k \subseteq N} \prod_{i=1}^{k} \left( \frac{r_{b_i}}{Y_{b_i}} \right)^{k-1}
= \frac{\sum_{b_1, b_2, \ldots, b_k \subseteq N} \prod_{i=1}^{k} r_{b_i}^{-s(k-1)}}{H_{m,s}^{k-1}}.
\]  

(18)

In order to simplify this expression, we use the elementary symmetric polynomial notation. If we have a vector \( V_n = (v_1, v_2, \ldots, v_n) \) of length \( n \), then the \( k \)-th degree elementary symmetric polynomial of these variables is denoted as

\[
\sigma_k(V_n) = \sigma_k(v_1, \ldots, v_n) = \sum_{1 \leq i_1 < i_2 < \cdots < i_k \leq n} v_{i_1} \cdots v_{i_k}.
\]  

(19)

Using this notation and by defining \( Y_n = (r_1^{s(k-1)}, r_2^{s(k-1)}, \ldots, r_n^{s(k-1)}) \), we have \( P_k \geq \frac{\sigma_k(Y_n)}{H_{m,s}^{k-1}} \).

Since the content request probability follows a Zipfian distribution, we have \( Pr[r_j \leq h_e] = \frac{H_{m,s}}{H_{m,s}^{j}} \).

Therefore, for a specific group of UTs \( N_{b_1}, N_{b_2}, \ldots, N_{b_k} \), the probability that they all request contents from the top \( h_e \) most popular contents is given by

\[
Pr[r_{b_1} \leq h_e, \ldots, r_{b_k} \leq h_e] = \prod_{j=1}^{k} Pr[r_{b_j} \leq h_e] = \left( \frac{H_{m,s}}{H_{m,s}^{j}} \right)^{k}.
\]  

(20)

We have already proved in Lemma 1 that for large values of \( m \) and \( h_e = e^{1-\epsilon} \), the ratio \( \frac{H_{m,s}}{H_{m,s}^{j}} \) is greater than \( 1 - \epsilon \).

Besides this, the fact that \( n \) is most likely much larger than \( k \), means that with a very high probability, for each set of UTs \( \{N_{b_1}, N_{b_2}, \ldots, N_{b_k}\} \), the requests come only from the \( h_e \) most popular contents. This implies that with a high probability, \( \sigma_k(Y_n) \geq \binom{n}{k} h_e^{-k \cdot s(k-1)} \).

Also, notice that \( H_{m,s} < \zeta(s) < \infty \).

Therefore, with a probability close to one, \( P_k \) can be lower bounded as

\[
P_k \geq \binom{n}{k} \left( \frac{h_e^{-k \cdot s}}{\zeta(s)} \right)^{k-1}.
\]  

(21)

This lower bound does not depend on \( m \) and only depends on \( n, \epsilon, s \) and \( k \).

Theorem 3 states 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 \( k \). The result hints the potential use of linear index coding in these networks. In the next section, we will prove that linear index coding can indeed be very useful and can be used to construct codes achieving order optimal capacity gains.

**Remark 2.** The above capacity improvement is found for a traditional single hop index coding scenario. For our proposed multihop setup, similar gains still hold. In our proposed setting, we consider communications for a small number of hops and therefore multihop communication can only affect the capacity gain by a constant factor and the order bound results will not be affected.

V. HEURISTICS ACHIEVING ORDER OPTIMAL CAPACITY

Both optimal and approximate solutions \( [3], [18] \) for the general index coding problem are NP-hard problems. Some efficient heuristic algorithms for the index coding problem were proposed \( [2] \) 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 contents requested by the UTs corresponding to the clique. Therefore, solving the clique partitioning problem, which is the problem of finding a clique cover of minimum size for a graph \( [13] \), yields a simple linear index coding solution. 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 all the UTs. Index coding rate is defined as the minimum number of required index codes to satisfy all the UTs. Since lower index coding rates translate into higher values of transmission savings (or index coding gains) \( [3] \), as discussed in \( [8] \), 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 (or the optimal index coding gain).

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 \( \overline{G}(V,E) \) 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 \( \chi \) in the dependency graph, we have \( \chi \) distinct independent sets in the complement graph. In other words, the chromatic number of the complement graph is \( \chi \). 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 running over the conflict graph can be directly used to obtain an achievable index coding 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 rate of \( 1 - \chi \). 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 \( n - \chi \).

\[\text{In a dependency graph of } n \text{ UTs with the index coding rate of } \chi, \text{ the number of saved transmissions, } n - \chi, \text{ is called the index coding gain.} \]

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

$$\chi_{G(n, p)} = -\left(\frac{1}{2} + o(1)\right) \log(1 - p) n \log n$$  \hspace{1cm} (22)

We will now use Theorem 4 and the designed undirected graph $G(n, p_c^2)$ to find the number of transmission savings using a graph coloring based heuristic in our network.

**Theorem 5.** Using a graph coloring algorithm, in a network with $n$ UTs almost surely gives us a linear index code with gain

$$l = \Theta\left(n + \left(\frac{1}{2} + o(1)\right) n \log n \log p^2\right).$$  \hspace{1cm} (23)

**Proof:** Assume that a helper is serving $n$ UTs where $n$ is a large number. As discussed in Theorem 4 the index coding gain is lower bounded by $n - \chi$ where $\chi$ is the chromatic number of the conflict graph. However, notice that on average the chromatic number of our non-deterministic conflict graph is upper bounded by the chromatic number of an undirected random graph with edge existence probability of $1 - p_c^2$. To prove this, notice that in the dependency graph, the probability of edge existence between two vertices is at least $p_e$ which implies that the number of edges in the dependency graph is on average greater than or equal to the number of edges in a directed Erdos-Reyni random graph $\overrightarrow{G}(n, p_e)$. However, we know that the number of edges in $\overrightarrow{G}(n, p_e)$ is at least equal to the number of edges in an undirected Erdos-Reyni random graph $G(n, p_c^2)$. Therefore, the conflict graph which is the complement of dependency graph, on average has less edges compared to a random graph with edge existence probability of $1 - p_c^2$ and consequently, its chromatic number cannot be greater than the chromatic number of $G(n, 1 - p_c^2)$. Given these facts, the index coding gain is lower bounded by $n - \chi_{G(n, 1 - p_c^2)}$. Since $1 - p_c^2$ is fixed, Theorem 4 implies that the chromatic number of the conflict graph is equal to

$$\chi_{G(n, 1 - p_c^2)} = -\left(\frac{1}{2} + o(1)\right) \log p^2 n \log n$$  \hspace{1cm} (24)

This proves that the index coding gain is lower bounded by $\Omega(n \times \{1 + \frac{1}{2} + o(1)\} \frac{1}{\log n} \log p^2)$ which asymptotically tends to $n$. However, the maximum index coding gain of $n$ UTs is also $n$. Therefore, this coding gain is also a tight bound.

**Remark 3.** Theorem 5 presents the index coding gain using a graph coloring algorithm which only counts the number of cliques in the dependency graph. The gain in Theorem 4 counts the number of disjoint cycles in the dependency graph. Theorem 2 proves that index coding gain in Theorem 1 is $\Theta(n)$ which means that it is order optimal. Theorem 5 is also proving the same result. Therefore, a graph coloring algorithm can achieve order optimal capacity gains.

**Remark 4.** We have shown our proposed algorithm in pseudocode in Algorithm 1. As we mentioned earlier, in our solution we only focus on content delivery. For cache placement, we assume that the greedy approximate algorithm proposed in [14] is used to populate helper caches. For helper assignment, the greedy algorithm is used. In other words, we suggest that the closest helper which has the content in its cache be assigned to the UT. During the content delivery phase, we use our proposed heuristic for index coding. We use a graph coloring heuristic for the conflict graph and find independent sets in the dependency graph. Then for each independent set only one multicast transmission is needed which will be sent by the helper. Our main contribution is to propose better content delivery algorithm compared to the baseline. In the decoding phase, UTs use their cached contents and the received content to decode their desired contents.

**Algorithm 1** Multihop Caching-Aided Coded Multicasting

1: **procedure** CACHE PLACEMENT
2: Use greedy cache placement algorithm in [14] to populate helper caches.
3: **end procedure**

4: **procedure** HELPER ASSIGNMENT
5: Use a greedy algorithm to assign the closest helper which has the requested content, to the UT.
6: **end procedure**

7: **procedure** CONTENT DELIVERY
8: Form the dependency graph and conflict graph.
9: Color the conflict graph by a graph coloring heuristic to find independent sets in dependency graph.
10: for every independent set $S$ found do
11: Helper multicasts coded content $\oplus_{k \in S} M_{r_k}$ to decode $M_{r_j}$.
12: **end for**
13: **end procedure**

14: **procedure** CONTENT DECODING
15: UT $j$ XORs some of its cached contents with the received coded content $\oplus_{k \in S} M_{r_k}$ to decode $M_{r_j}$.
16: **end procedure**

VI. SIMULATIONS

In this section, we will show our simulation results. To show the performance of our coding technique, we have plotted the simulation results for five different sets of parameters in Figure 4. In this simulation, we assume that index coding is done in the packet level. We plotted the average packets sent in each transmission. We have assumed that the UTs are requesting contents based on a Poisson distribution with an average rate of $\frac{1}{7}$. This way we can assure that the average total request rate is one and we are efficiently using the time resource without generating unstable queues. The optimum solution is an NP-hard problem. However, we used a very simple heuristic algorithm to count the number of cliques and cycles of maximum size 4. 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, optimal index coding or more sophisticated algorithms can achieve better results compared to what we obtained by our simple algorithm.

For small values of $s$, the content request distribution is close to uniform, the dependency graph is very sparse and there is little benefit of using index coding. For large values of $s$, most UTs are requesting similar contents which results in broadcasting the same content to all nodes which is equal to one content per transmission. The main benefit of index coding happens for values of $s$ between 0.5 and 2 which is usually the case in practical networks. Note that a wireless distributed caching system with no index coding, will always have one content per transmission.

Figure 4 compares the average number of contents transmitted in each time slot when using index coding as a function of the Zipfian parameter $s$ using a suboptimal search.

| $s$ | Number of contents sent per transmission |
|-----|----------------------------------------|
| 0.1 | 2                                      |
| 0.2 | 1                                      |
| 0.3 | 0                                      |

Fig. 4. Average number of contents transmitted in each time slot when using index coding as a function of the Zipfian parameter $s$.

Figure 5 compares the average number of requests satisfied per time slot per helper between our proposed scheme and the baseline approach. For this simulation, a Zipfian content request distribution with parameter $s = 2$ is assumed. The cell radius is 400 meters, D2D transmission range is assumed to be 100 meters and 1000 UTs are considered in this figure similar to the simulations in [4] and Figure 1. Notice that with the baseline approach at least 27 helpers are required to cover the entire network and 24, 20, 14 and 10 helpers can cover 97%, 93%, 78% and 62% of the network, respectively. Using multihop D2D communications with a maximum of three hops, 4 helpers can cover the entire network while 3 helpers can cover 95% of the network nodes. We have shown that even with our very simple heuristic algorithm, multihop D2D can significantly improve the helper utilization ratio. Note that in baseline approach, each helper can at most transmit one content per transmission, however, our approach can satisfy more than one request per transmission by taking advantage of the side information that is stored in nodes’ caches. As the number of requests per user increases, there are more possibilities of creation of cliques of large size which results in increasing the efficiency of helper nodes. The simulation is carried for content request probability of up to 0.5 since realistically, no more than one third of nodes at any given time, request contents in the network.

**VII. Conclusion**

An efficient order optimal content delivery approach is proposed for future wireless cellular systems. We take advantage of femtocaches [14] and multihop communication using index coding. We proved that using index coding is very efficient technique by taking advantage of side information stored in UTs. Further, it was shown under Zipfian content distribution, linear index coding could be order optimal. A heuristic graph coloring algorithm is proposed that achieves order optimal capacity bounds. Our simulation result demonstrates the gains that can be achieved with this approach.

Our proposed scheme improves the efficiency of femtocaches by taking advantage of multihop communications and index coding. One challenge of multihop communication is connectivity when nodes are moving fast. One solution is to divide the contents into equal smaller chunks and treat each chuck as a content. Another challenge is the additional delay imposed by using multihop communications. However, it is not readily clear that baseline approach provides lower delays. The reason is the fact that by using multihop communication, larger portion of the cell can be covered when the same number of femtocaches are used in both approaches as shown in Figure 1. A more detailed comparison of mobility and delay in single hop and multihop communication is the subject of future study. Security, overhead, routing are other issues that should be investigated in future works.

**References**

[1] http://spectrum.ieee.org/telecom/wireless/millimeter-waves-may-be-the-future-of-5g-phones

[2] Amendments in IEEE 802.11ad™ enable multi-gigabit data throughput and groundbreaking improvements in capacity. https://standards.ieee.org/news/2013/082/11ad.html. 2013. [Online; accessed 9-October-2015].

[3] Ziv Bar-Yossef, Yitzhak Birk, TS Jayram, and Tomer Kol. Index coding with side information. Information Theory, IEEE Transactions on, 57(3):1479–1494, 2011.

[4] Federico Boccardi, Robert W Heath, Aurelie Lozano, Thomas L Marzetta, and Petar Popovski. Five disruptive technology directions for 5G. Communications Magazine, IEEE, 52(2):74–80, 2014.

[5] Bélá Bollobás. The chromatic number of random graphs. Combinatorica, 8(1):49–55, 1988.
[6] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. Web caching and Zipf-like distributions: Evidence and implications. In INFOCOM’99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, volume 1, pages 126–134, New York, NY, 1999. IEEE.

[7] Vikram Chandrasekhar, Jeffrey G Andrews, and Alan Gatherer. Femtocell networks: a survey. Communications Magazine, IEEE, 46(9):59–67, 2008.

[8] Mohammad Asad R Chaudhry, Zakia Asad, Alex Sprintson, and Michael Langberg. On the complementary index coding problem. In Proceedings of Information Theory (ISIT). IEEE International Symposium on, pages 244–248, Saint Petersburg, Russia, 2011. IEEE.

[9] Mohammad Asad R Chaudhry and Alex Sprintson. Efficient algorithms for index coding. In INFOCOM Workshops, IEEE, pages 1–4, Phoenix, AZ, 2008. IEEE.

[10] Michelle Effros, Salim El Rouayheb, and Michael Langberg. An equivalence between network coding and index coding. Information Theory, IEEE Transactions on, 61(5):2478–2487, 2015.

[11] Salim El Rouayheb, Alex Sprintson, and Costas Georghiades. On the index coding problem and its relation to network coding and matroid theory. Information Theory, IEEE Transactions on, 56(7):3187–3195, 2010.

[12] P Erdős and L Pósa. On the maximal number of disjoint circuits of a graph. Publ. Math. Debrecen, 9:3–12, 1962.

[13] Michael R Garey and David S Johnson. Computers and intractability. W. H. Freeman, San Francisco, 2002.

[14] Negin Golrezaei, Karthikeyan Shanmugam, Alexandros G Dimakis, Mingyue Ji, Giuseppe Caire, and Andreas F Molisch. Fundamental limits of caching in wireless D2D networks. Information Theory, IEEE Transactions on, 62(2):849–869, 2016.

[15] Donald E Knuth. Big omicron and big omega and big theta. ACM Sigact News, 8(2):18–24, 1976.

[16] Marie-Jose Montpetit, Cedric Westphal, and Dirk Trossen. Network coded caching-aided multicast for efficient content delivery. In Proceedings of the 1st ACM workshop on Emerging Name-Oriented Mobile Networking Design-Architecture, Algorithms, and Applications, pages 31–36, Hilton Head, SC, 2012. ACM.

[17] Hong Kong, 2013. ACM.

Mohsen Karimzadeh Kiskani received his bachelors degree in mechanical engineering from Sharif University of Technology in 2008. He got his Masters degree in electrical engineering from Sharif University of Technology in 2010. He got a Masters degree in computer science from University of California Santa Cruz in 2016. He is currently a PhD candidate in electrical engineering department at University of California, Santa Cruz. His main areas of interest include wireless communications and information theory. He is also interested in the complexity study of Constraint Satisfaction Problems (CSP) in computer science.

Hamid Sadjadpour (S’94-M’95-SM’00) received his B.S. and M.S. degrees from Sharif University of Technology and Ph.D. degree from University of Southern California, respectively. After graduation, he joined AT&T as a member of technical staff and finally Principal member of technical staff until 2001. In fall 2001, he joined University of California, Santa Cruz (UCSC) where he is now a Professor. Dr. Sadjadpour has served as technical program committee member and chair in numerous conferences. He has published more than 170 publications and has awarded 17 patents. His research interests are in the general area of wireless communications and networks. He is the co-recipient of best paper awards at 2007 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2008 Military Communications (MilCOMM) conference, and 2010 European Wireless Conference.