Coding-based Cooperative Caching in On-demand Data Broadcast Environments

Houling Ji[a,b], Victor C.S. Lee[b], Chi-Yin Chow[b], Kai Liu[c,*], Guoqing Wu[a]

[a] *School of Computer, Wuhan University, Wuhan, China*
[b] *Department of Computer Science, City Univeristy of Hong Kong, Hong Kong*
[c] *College of Computer Science, Chongqing University, Chongqing, China*

Abstract

Data broadcasting has been commonly deployed in many emerging mobile applications such as intelligent transportation systems and location-based services, because it is a scalable approach to disseminating information from a mobile support station (MSS) to a large population of mobile hosts (MHs). To provide timely data access and better data availability, MHs can store data items broadcast by the MSS in their local caches and share cached data items cooperatively among neighboring peers via peer-to-peer (P2P) communication. However, if MHs are not neighbors, they cannot cooperate even if they have each other's requested data items in their own caches. Network coding is a technique, by which multiple MHs can decode out different requested data items from an encoded packet broadcast by the MSS in one broadcast time unit. In this work, we propose a network coding based solution to enable MHs which are not neighbors to cooperate indirectly. We formulate the Maximum Channel Efficiency Encoding (MCEE) problem by introducing network coding and cooperative caching techniques in on-demand data broadcast environments. We prove that MCEE is NP-hard by constructing a polynomial-time reduction from the Minimum Clique Cover (MCC) problem. Further, we propose two schemes (NCM and NCB) for on-demand data broadcasting using network coding. In each scheme, we propose two algorithms running at the MSS and MHs for making encoding decisions and decoding requested data items, respectively. We build the simulation model for performance evaluation and the simulation results demonstrate that the proposed schemes not only increase the bandwidth efficiency of the limited downlink communication channel, but also enhance the system performance by reducing the data access latency.

Keywords: On-demand Data Broadcast, Peer-to-peer Communication, Cooperative Caching, Network Coding

1. Introduction

Data broadcast is an effective way to disseminate data items in an infrastructure network due to its scalability and flexibility. A typical infrastructure network consists of multiple mobile hosts (MHs) and a mobile support station (MSS). The MSS disseminates information to MHs within its service area. MHs could be mobile devices such as laptops, mobile phones or other devices equipped

*Corresponding author
Email addresses: jihouling@whu.edu.cn (Houling Ji), csvlee@cityu.edu.hk (Victor C.S. Lee), chiychow@cityu.edu.hk (Chi-Yin Chow), liukai0807@cqu.edu.cn (Kai Liu), wgq@whu.edu.cn (Guoqing Wu)

with wireless communication modules. Information is transmitted from the MSS to MHs via the downlink communication channel. Pull-based broadcast, which is commonly known as on-demand broadcast, is one of the most promising data broadcast techniques for disseminating information to a large population of MHs [7, 17, 18, 19, 22]. In on-demand broadcast environments, MHs submit requests to the MSS via the uplink communication channel. Outstanding requests are pended in the request queue at the MSS. In each broadcast time unit, the MSS selects the most rewarding data item to broadcast via the downlink communication channel according to a certain scheduling policy, such as FCFS (First Come First Served) [11]. Requesting MHs tune in to the downlink communication channel and wait for their requested data items. However, in such an on-demand broadcast environment, bandwidth efficiency of the downlink communication channel cannot be fully exploited because only MHs requesting the same data item have the chance to be served in one broadcast time unit.

Network coding [3] is a technique in which intermediate nodes can combine and forward data packets from multiple links. It has attracted researchers' attention due to its potential to enhance the system performance for both mobile ad hoc networks [23, 24, 25, 31, 34] and infrastructure networks [2, 6, 20, 33]. In on-demand broadcast environments, with network coding, it is possible that multiple MHs requesting different data items can be satisfied simultaneously by utilizing their cached data items. Thus, network coding has the potential to further improve the bandwidth efficiency of the downlink communication channel.

With recent development of peer-to-peer (P2P) wireless communication technologies and the increasing capacity of MHs' storage, a new information sharing paradigm appears. MHs can not only retrieve information from the MSS, but also share information among neighboring peers. The neighboring peers of an MH refer to those MHs which reside in its transmission range. This kind of information sharing among peers is called cooperative caching [10]. In our recent studies [8, 9, 10], COCA (cooperative caching) and GroCoca (group-based cooperative caching) have been proposed for data dissemination in on-demand broadcast environments. In particular, COCA is a cooperative caching framework, in which P2P communication technology is used for information sharing among MHs. GroCoca extends COCA based on the concept of a tightly-coupled group (TCG), which is a group of MHs that are geographically and operationally close to each other [10]. GroCoca outperforms COCA in terms of reducing data access latency, which is the duration from the moment when an MH submits a request to the moment when the requested data item is received. However, in GroCoca, it is difficult for MHs in disjoint TCGs to collaborate and share their cached data items. Inspired from the advantage of network coding in terms of exploiting cached data items of MHs, this work is dedicated to exploring cooperative caching of MHs which are residing out of each other's transmission range, so as to further strengthen the information sharing paradigm and improve the overall system performance.

The main contributions of this paper are outlined as follows.

- We exploit the synergy between network coding, on-demand broadcast and cooperative caching in mobile data dissemination environments. First, network coding strengthens the information sharing paradigm of cooperative caching by enabling MHs residing in different groups to cooperate. In case they have each other's requested data items in their own caches, the MSS can encode multiple requested data items in a single packet for broadcasting and the MHs can retrieve their own requested data items simultaneously in one broadcast time unit. Otherwise, the MSS has to broadcast the requested data items one by one in multiple broadcast time units. Second, cooperative caching facilitates the operation of network coding in on-demand broadcasting. Consider a request submitted by an MH to the MSS. MHs which

are in the same group of this requesting MH cannot have the requested data item in their own caches. Otherwise, they can simply share the data item with the requesting MH. Therefore, the MSS does not need to consider the cache contents of MHs which are in the same group of the requesting MH in making encoding decisions.

- We introduce a novel system model by incorporating network coding into the cooperative caching framework. In particular, we outline the communication protocol of the system. Furthermore, we formulate the Maximum Channel Efficiency Encoding (MCEE) problem by introducing network coding and cooperative caching techniques in on-demand data broadcast environments. We prove that MCEE is NP-hard by constructing a polynomial-time reduction from the Minimum Clique Cover (MCC) problem.

- We propose two schemes (NCM and NCB). NCM adopts network coding only to the MSS, while NCB adopts network coding in both the MSS and MHs. Specifically, two heuristic algorithms are proposed in each scheme. The algorithm running at the MSS makes the encoding decisions to satisfy multiple requests for different data items in one broadcast time unit. To boost the effectiveness of network coding, data items to be encoded are not restricted to the data items cached by the requesting MH but also include those cached by its peers. Another algorithm running at MHs enables the requesting MHs residing in different groups to retrieve their requested data items by decoding the data packets broadcast by the MSS. The decoding can be done not only using the data items cached by the requesting MHs but also their peers. So, these schemes seek to further improve information sharing by exploiting network coding at both the MSS and MHs.

- We build the simulation model for performance evaluation. The simulation results demonstrate that incorporating network coding into cooperative caching makes the data scheduling more flexible. The proposed schemes further increase the bandwidth efficiency of downlink communication channel, and significantly improve the system performance in terms of reducing data access latency.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 describes the system architecture. In Section 4, we formulate the MCEE problem and prove that it is NP-hard. Two new coding-based cooperative caching schemes are proposed in Section 5. Section 6 presents the simulation model and the performance evaluation. Finally, we conclude this work in Section 7.

2. Related Work

Since Ahlswede *et al.* introduced the concept of network coding in [1], network coding has been widely used in wireless networks to improve the performance of multicast flows [21, 31]. Birk and Kol [4] applied network coding in on-demand broadcast environments and demonstrated its effectiveness. Zhan *et al.* [33] proposed a generalized encoding framework to incorporate network coding into data scheduling algorithms for on-demand broadcast. Chen *et al.* [6] proposed two novel coding assisted algorithms called ADC-1 and ADC-2 in which both the coding and scheduling problems are considered. Ali *et al.* [2] gave an extensive performance evaluation on applying network coding for data dissemination in on-demand broadcast environments, which demonstrated the efficiency and adaptability of coding-based scheduling algorithms. Liu *et al.* [20] incorporated network coding into vehicular networks for infrastructure-based data dissemination.

Cooperative caching in mobile environments attracts wide attention in recent decade. Several cooperative caching schemes were proposed in mobile environments [13, 16, 26, 28, 29, 32, 36]. Sailhan and Issarny [26] proposed a cooperative caching scheme in ad hoc networks. If an MH can communicate with an infrastructure support in a single hop, it directly obtains data items from the infrastructure support. Otherwise, the MH would seek help from its peers. If none of its peers cache the requested data items, the peers route these data requests to the nearest infrastructure support. Yin and Cao [32] proposed three schemes: CachePath, CacheData, and HybridCache, aiming at efficient data access in ad hoc networks. In their model, there is a special node among all nodes called data center which connects to the wired network. In CacheData scheme, an intermediate node caches popular data items requested by a number of nodes. In CachePath scheme, an intermediate node caches the path along which the requested data item can be finally found. In HybridCache scheme, CacheData and CachePath schemes are incorporated so it could make the best of both schemes. Ting and Chang [28] proposed an improved cooperative caching scheme called group-based cooperative caching (GCC) to enhance the performance of most group-based caching schemes by exchanging a bitmap data directory periodically among MHs.

In our previous work [8, 9, 10], two cooperative caching schemes, namely COCA and GroCoca have been proposed for information sharing in on-demand broadcast environments. Specifically, GroCoca is extended from COCA by incorporating the concept of tightly-coupled group (TCG). In COCA [8], each MH has two wireless network interface cards. One of them is used to communicate with the MSS, and the other is used to communicate with other MHs. There are uplink and downlink communication channels between the MSS and MHs. The uplink communication channels are P2P channels, while the downlink communication channels are broadcast channels. Every MH works together with its peers to cache data items cooperatively via P2P communication channels. The peers of an MH refer to those MHs which reside in the transmission range of the MH. There are two approaches for an MH communicating with its peers, namely P2P broadcast and P2P point-to-point communication. GroCoca [10] is proposed based on the concept of TCG, which is defined as a group of MHs sharing common mobility pattern and data affinity. Note that peers are not necessarily members of the same TCG. The MSS maintains a two-dimensional weighted average distance matrix (WADM) and a two-dimensional access similarity matrix (ASM). WADM is based on the Euclidean distance between MHs, while ASM is based on the frequency that an MH accesses a data item. A TCG discovery algorithm is executed at the MSS to update the information of TCG members. Different from the above work, we apply network coding in cooperative caching schemes to make the data scheduling more flexible, strengthen the information sharing paradigm, and further enhance the system performance.

Some studies also considered exploiting the synergy between network coding and cooperative caching. Wang *et al.* [30] proposed a cache management framework for information-centric network (ICN) based on software-defined networking (SDN) using a controller to determine the caching strategy and content routing via linear network coding (LNC). They developed a network coding based cache management (NCCM) algorithm to reduce the network bandwidth cost in ICNs. Not only the system topology but also the problem of their work are different from those of ours. The system architecture of their work is ICN, while our work studied an on-demanded data broadcast system with P2P communication. Their work formulated the optimal cache management problem for ICNs, while our work formulated the MCEE problem and focused on the scheduling and encoding processes in the MSS and the decoding process in MHs. We exploited the possibility for MHs who are not physical neighbors to collaborate. Zhang *et al.* [35] proposed a cooperative caching scheme combining network coding technique, called NC-COCA, to exploit

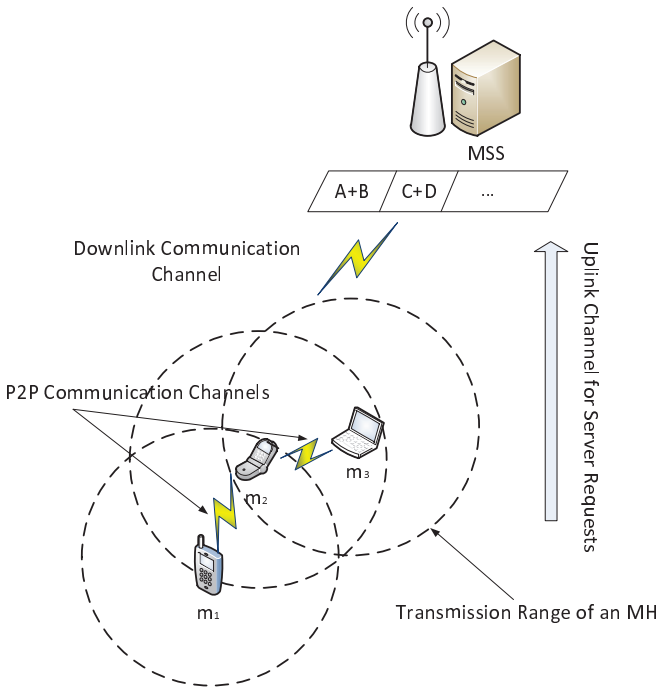


Figure 1: System Architecture

cache spaces of relatively idle MHs. In their system, they set up a strict assumption that some MHs request data items more frequently (high activity node) than others (low activity node). In our previous work [15], we proposed an initial idea of applying network coding only to the MSS, so that MHs which are not neighbors can cooperate indirectly. In this work, we extend the application of network coding to both MSS and MHs, which provides more opportunities for the MSS to encode a requested data item with any data item cached by the requesting MH and also its peers, and allows requesting MHs to retrieve data items from peers to decode their own requested data items from encoded data packets broadcast by the MSS.

3. Architecture

The system architecture is shown in Figure 1. In particular, the system consists of one MSS and a number of MHs. Suppose there are $NumMH$ MHs $m_1, m_2, ..., m_{NumMH}$ and the MSS contains a database of $NumData$ data items $d_1, d_2, ..., d_{NumData}$. Each MH has a local cache storing data items received from either the MSS or neighboring MHs (peers). In the system, all the entities, including the MSS and MHs, have certain communication areas determined by their respective transmission ranges. Therefore, one MH can communicate with others which are residing in its own transmission range. All the MHs are in the communication area of the MSS, which means that MHs can always communicate with the MSS wherever they move. Each MH does not manage its cache alone. Instead, multiple MHs share their cached data items cooperatively via P2P communication. There are two P2P communication paradigms: P2P broadcast and P2P point-to-point. All MHs within the transmission range of the source MH receive the broadcast message in P2P broadcast communication, while there is only one destination MH for the source MH in P2P point-to-point communication. The MSS maintains a request queue for outstanding requests submitted by MHs. According to a certain scheduling policy, the MSS fetches requested data items from the database and makes decision whether to encode them or not based on the

information of both cached and requested items of each requesting MH. A data packet containing the requested data items will be broadcast to MHs via downlink communication channel. The objective is to utilize limited broadcast bandwidth to serve as many requests as possible.

3.1. Communication Protocol

There are three possible situations that an MH may encounter when requesting a data item.

1. **Local Cache Hit (LCH).** If the required data item can be found in its local cache, it is a local cache hit. Otherwise, it is a local cache miss.
2. **Global Cache Hit (GCH).** If the MH encounters a local cache miss, it attempts to enlist its peers for help. If any peer caches and replies the required data item, it is a global cache hit. Otherwise, it is a global cache miss.
3. **Server Request.** If the MH encounters a global cache miss, it will send a server request to the MSS via uplink communication channel. Then, the MH will tune in to downlink communication channel to retrieve the required data item.

According to above situations, the communication protocol of the system can be divided into three steps:

1. **Local Cache Search.** Searching the local cache is the first step to find a requested data item, and the time for this operation is called the local cache search time.
2. **Global Cache Search.** If the MH cannot find the desired data item in its local cache, it broadcasts a request for the data item to its peers through P2P broadcast communication. There are two possible outcomes for the global cache search, resulting in different global cache search time. a) The peers which cache the required data item send a *reply* message back to the requesting MH through P2P point-to-point communication. Within a timeout period, if the requesting MH receives the *reply* messages from its peers, the peer from which the MH receives the first reply is selected as a target peer. Then, the requesting MH sends a *retrieve* message to inform the target peer to reply the data item through P2P point-to-point communication. When the target MH receives the *retrieve* message, it sends the required data item to the requesting MH. In this case, the global cache search time is constituted of the time for broadcasting the request from the MH to its peers, the time for sending the *reply* message from the peer to the requesting MH, the time for sending the *retrieve* message from the requesting MH to its peers, the time for transmitting the requested data item from the peer to the requesting MH and the waiting time for the P2P point-to-point channel. b) Otherwise, within the timeout period, if no peer sends back a *reply* message, the requesting MH will submit a request to the MSS for the data item. In this case, the global cache search time is equal to the timeout period.
3. **Server Request.** If the MH encounters a global cache miss, it will send a server request to the MSS via uplink communication channel. Then, the MH will tune in to downlink communication channel to retrieve the required data item. The server request time is constituted of the time for transmitting the request from the MH to the MSS, the request waiting time at the MSS and the time for broadcasting the requested data item from the MSS to the MH.

If an MH has a local cache hit for a required data item, the local cache search time is the data access latency. If the MH encounters a local cache miss and then has a global cache hit, the data access latency is the sum of local cache search time and global cache search time. Otherwise, if the MH encounters a local cache miss and a global cache miss, the data access latency equals the

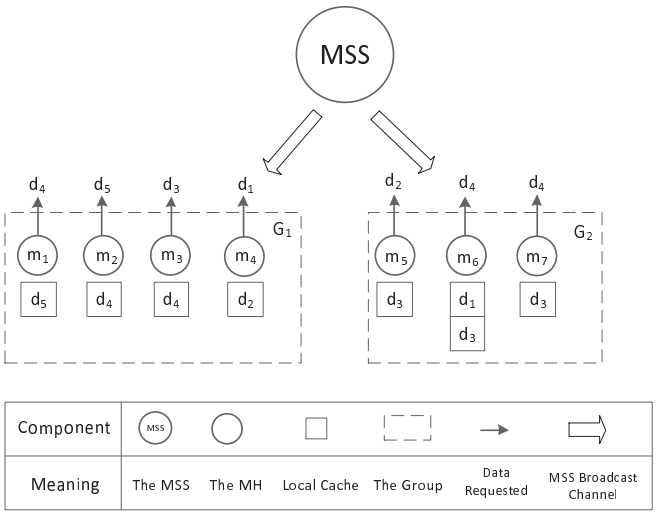


Figure 2: An Example of Data Dissemination

Table 1: Possible Solutions to the Data Dissemination Example

Solution	GCC	NC	MSS Broadcast Sequence	Number of Broadcast
1	No	No	d_1, d_2, d_3, d_4, d_5	5
2	Yes	No	d_1, d_2, d_3, d_4	4
3	No	Yes	$d_4 \oplus d_5, d_3 \oplus d_4, d_1, d_2$	4
4	Yes	Yes	$d_1 \oplus d_2, d_3 \oplus d_4$	2

sum of local cache search time, global cache search time and server request time. Note that in this paper, the local processing times at MHs and the MSS are considered negligible.

The timeout period is initially set to the round-trip time of P2P communication channel such that initial timeout is computed by: $\frac{request\ size\ +\ reply\ size}{BW_{P2P}} \times \varphi$, where BW_{P2P} is the bandwidth of P2P communication channel and φ is a congestion factor. Then, for each P2P broadcast, the requesting MH records the duration τ from the moment the requesting MH broadcasts the request to its peers to the moment a *reply* message is received for continuously adjusting the timeout period. Accordingly, the timeout period is computed by: $\overline{\tau} + \varphi'\sigma_\tau$, where $\overline{\tau}$ and σ_τ are the mean and standard deviation of τ, respectively, while φ' is the system parameter to weigh the standard deviation in computing the timeout period.

3.2. Motivation Example

Consider a data dissemination scenario in Fig. 2, which consists of one MSS and seven MHs, m_1, m_2, m_3, m_4, m_5, m_6 and m_7. MHs can be divided into two groups G_1 and G_2: m_1, m_2, m_3 and m_4 are in G_1, while m_5, m_6 and m_7 are in G_2. m_1 is requesting d_4 and has d_5 in the local cache. m_2 is requesting d_5 and has d_4 in the local cache. m_3 is requesting d_3 and has d_4 in the local cache. m_4 is requesting d_1 and has d_2 in the local cache. m_5 is requesting d_2 and has d_3 in the local cache. m_6 is requesting d_4 and has d_1, d_3 in the local cache. m_7 is requesting d_4 and has d_3 in the local cache.

We consider the problem that, if the MSS can broadcast one data packet in one broadcast time unit, then how many broadcast time units are required to satisfy all the requesting MHs. The possible solutions are given in Table 1. Solution 1 shows a situation of traditional on-demand

broadcast with neither group-based cooperative caching (GCC) nor network coding (NC). The MSS will broadcast d_1, d_2, d_3, d_4, d_5 in sequence to satisfy all MHs. Solution 2 shows a situation of on-demand broadcast with GCC but without NC. Since m_1 and m_2 can obtain their requested data items from peers, the MSS only has to broadcast d_1, d_2, d_3, d_4 in sequence to satisfy other MHs. Note that even though m_3 has d_4 in its local cache which is requested by both m_6 and m_7, while both m_6 and m_7 also has d_3 in its local cache which is requested by m_3, they can hardly share their cached data items as they do not belong to the same TCG. Solution 3 shows a situation with NC but without GCC. At the first broadcast time unit, the MSS broadcasts the data packet $d_4 \oplus d_5$. The XOR ($\oplus$) operation is commonly used in network coding for encoding and decoding due to its trivial computation overhead [2, 6, 33]. When m_1 receives the encoded packet, it can obtain d_4 by computing $d_5 \oplus (d_4 \oplus d_5)$. m_2 can obtain d_5 in a similar way. After that, the MSS broadcasts $d_3 \oplus d_4$ which can satisfy m_3, m_6, m_7, and then broadcasts d_1, d_2 in sequence to serve m_4, m_5, respectively. Solution 4 shows the situation with both GCC and NC, where not only MHs in the same TCG but also the ones in different TCGs can cooperate and share cached data items. The MSS only has to broadcast two data packets to satisfy all MHs. Note that when m_5 receives the packet $d_1 \oplus d_2$, although it cannot decode the packet directly, it can retrieve d_1 from its TCG member m_6. Finally, it obtains d_2 by computing $d_1 \oplus (d_1 \oplus d_2)$. This example illustrates that GCC and NC can help to decrease the number of MSS broadcast. Since data transmission time between peers is much shorter than server broadcast time, the average data access latency will be significantly reduced by exploiting the synergy between network coding and cooperative caching.

4. Problem Formulation

In the coding-based cooperative caching system, the set of data items in the database of MSS is denoted by $D = \{d_1, d_2, \ldots, d_{NumData}\}$, where $NumData$ is the number of data items. The set of MHs in the system is denoted by $M = \{m_1, m_2, \ldots, m_{NumMH}\}$, where $NumMH$ is the number of MHs.

Definition 1. *Let $Q = \{Q_1, Q_2, \ldots, Q_{NumMH}\}$ be the set of requests issued by corresponding MHs. Each request $Q_i (1 \leq i \leq NumMH)$ is associated with a set of data items, $\{d_{\alpha^i(1)}, d_{\alpha^i(2)}, \ldots, d_{\alpha^i(|Q_i|)}\}$, where $d_{\alpha^i(j)} \in D (1 \leq j \leq |Q_i| \leq NumData)$ and $|Q_i|$ is the number of data items in Q_i. Let $C_i = \{d_{\beta^i(1)}, d_{\beta^i(2)}, \ldots, d_{\beta^i(|C_i|)}\}$ be the set of data items cached in m_i, where $d_{\beta^i(j)} \in D (1 \leq j \leq |C_i| \leq NumData)$ and $|C_i|$ is the number of data items in m_i's cache.*

Definition 2. *Suppose MHs can be divided into k TCGs. Let $G = \{G_1, G_2, \ldots, G_k\}$ be the set of TCGs. Each TCG $G_i (1 \leq i \leq k)$ consists of a set of MHs, $\{m_{\gamma^i(1)}, m_{\gamma^i(2)}, \ldots, m_{\gamma^i(|G_i|)}\}$, where $m_{\gamma^i(j)} \in M (1 \leq j \leq |G_i| \leq NumMH)$ and $|G_i|$ is the number of MHs in G_i.*

Definition 3. *A group request is a set of data items requested by MHs in the same TCG. A group cache is a set of data items stored in MHs of the same TCG. Let $GQ = \{GQ_1, GQ_2, \ldots, GQ_k\}$ be the set of group requests issued by MHs in the TCGs. Let $GC = \{GC_1, GC_2, \ldots, GC_k\}$ be the set of group caches. GQ_i and GC_i $(1 \leq i \leq k)$ can be derived by performing the union operation and eliminating the duplicate items.*

In the example shown in Fig. 3, there are 3 TCGs in the system, and each TCG includes 3 MHs. Each MH requests a data item and stores some data items in its cache. From the MSS point of view, all the caches in the same TCG can be considered as a group cache and all requests in the same TCG can be considered as a group request.

Table 2: Summary of Notations

Notations	Descriptions
D	The set of data items in the database
d_i	The ith data item of D
$NumData$	Number of data items in the database
M	The set of MHs
m_i	The ith MH
$NumMH$	Number of MHs
Q	The set of requests issued by corresponding MHs
Q_i	The set of data items requested by m_i
C_i	The set of data items cached by m_i
G	The set of TCGs
G_i	The ith TCG
k	Number of TCGs
GQ	The set of group requests issued by MHs in TCGs
GQ_i	The set of data items requested by MHs in G_i
GC	The set of group caches
GC_i	The set of data items cached in the ith group cache
$\mathcal{G} = \{V, E\}$	An undirected graph with vertex set V and edge set E
v_{ij}	A vertex of $\mathcal{G}$
e	An edge of $\mathcal{G}$
$\mathcal{C}$	An arbitrary clique in $\mathcal{G}$
$G_{\mathcal{C}}$	The set of TCGs covered in $\mathcal{C}$
$D_{\mathcal{C}}$	The set of requested data items in $\mathcal{C}$
p	An encoded data packet
$\mathscr{C}$	The set of cliques in $\mathcal{G}$
$\mathscr{C}_i$	The ith clique
w	Number of cliques in $\mathcal{G}$
$V_{\mathscr{C}_i}$	The set of vertices included in $\mathscr{C}_i$
$\mathcal{P}$	The set of broadcast data packet
P_j	The jth broadcast data packet

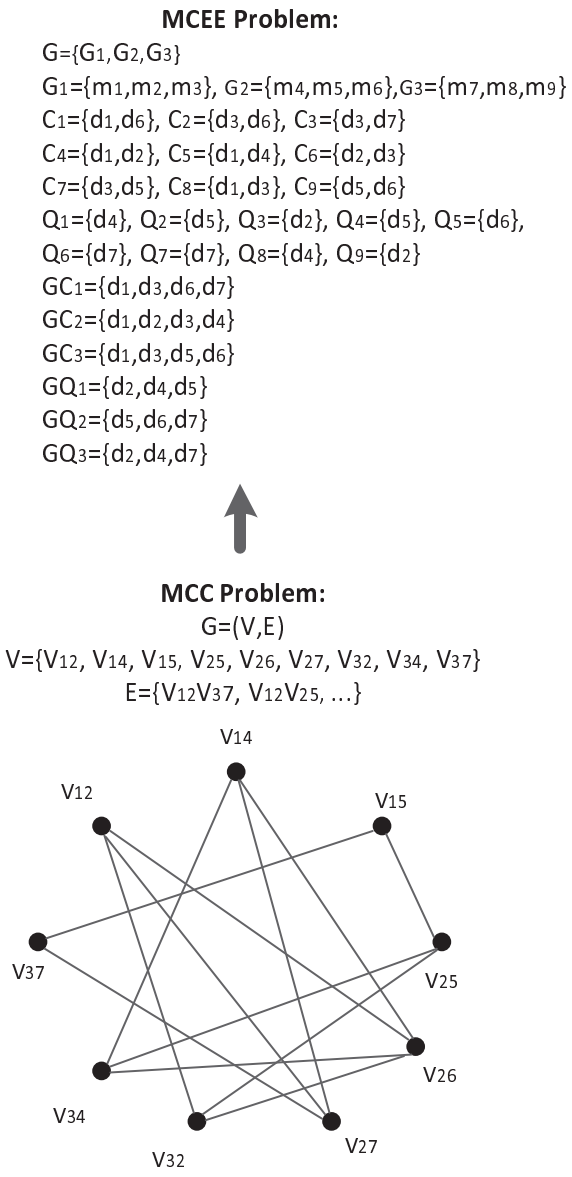


Figure 3: An Example of Problem Reduction from MCC to MCEE

Based on the definitions, the Maximum Channel Efficiency Encoding problem (MCEE) can be specified as follows.

Instance : There is one MSS and a set of group requests GQ and the corresponding group caches GC.

Question : Is there a group-based encoding scheme that can satisfy all requests in minimum number of MSS broadcast?

We prove that the MCEE problem is NP-hard by constructing a polynomial-time reduction from the Minimum Clique Cover problem (MCC, which is NP-hard) to the MCEE problem.

The MCC problem is specified as follows.

Instance : An undirected graph $\mathcal{G} = (V, E)$.

Question : Determine the minimum set of cliques in $\mathcal{G}$ such that each vertex in V is in at least one of the selected cliques.

Proof. Given an instance of $\mathcal{G}(V, E)$ for determining whether there is the minimum set of cliques in $\mathcal{G}$ such that each vertex in V is in at least one of the selected cliques, we consider an instance of the MCEE problem to complete the reduction as follows.

- We construct a one-to-one mapping from each vertex $v_{ij}(v_{ij} \in V)$ to the group request GQ_i for data item $d_j(d_j \in D)$, i.e., one or more than one MH in G_i requests d_j.
- For any relationship between two groups satisfying one of the following rules, We construct a one-to-one mapping from each edge between two vertices $v_{i_1j_1}$ and $v_{i_2j_2}$ to the relationship.
  - Rule 1: $j_1 = j_2$
  - Rule 2: $j_1 \neq j_2$, $d_{j_1} \in GC_{i_2}$ and $d_{j_2} \in GC_{i_1}$

An example of the above reduction is illustrated in Fig. 3.

Let $\mathcal{C} = \{v_{i_1j_1}, v_{i_2j_2}, \ldots, v_{i_{|\mathcal{C}|}j_{|\mathcal{C}|}}\}$ be an arbitrary clique in $\mathcal{G}$. A clique is a subset of vertices where any two vertices in the subset are connected. Let $G_{\mathcal{C}} = \{G_i | v_{ij} \in \mathcal{C}\}$ be the set of groups covered in $\mathcal{C}$. Let $D_{\mathcal{C}} = \{d_{\mathcal{C}(j)} | v_{i\mathcal{C}(j)} \in \mathcal{C}\}$ be the set of requested data items in $\mathcal{C}$. $D_{\mathcal{C}}$ can be also written as $D_{\mathcal{C}} = \{d_{\mathcal{C}(1)}, d_{\mathcal{C}(2)}, \ldots, d_{\mathcal{C}(|D_{\mathcal{C}}|)}\}$.

First, we consider a proposition: By broadcasting the encoded data packet $p = d_{\mathcal{C}(1)} \oplus d_{\mathcal{C}(2)} \oplus \cdots \oplus d_{\mathcal{C}(|D_{\mathcal{C}}|)}$, for any group $G_i \in G_{\mathcal{C}}$, it can obtain its requested data item d_j from p if $v_{ij} \in \mathcal{C}$. Suppose $e = (v_{i_1j_1}, v_{i_2j_2})$ is an edge connecting two vertices $v_{i_1j_1}$ and $v_{i_2j_2}$ in $\mathcal{C}$. (1) If $j_1 = j_2$, both group requests (GQ_{i_1} for j_1 and GQ_{i_2} for j_2) can be satisfied if the MSS broadcasts d_{j_1}. This is because one broadcast can serve all group requests for the same data item. (2) If $j_1 \neq j_2$, it means the group G_{i_1} caches d_{j_2} and G_{i_2} caches d_{j_1}. Therefore, if the MSS broadcasts $d_{j_1} \oplus d_{j_2}$, G_{i_1} can derive d_{j_1} by computing $d_{j_2} \oplus (d_{j_1} \oplus d_{j_2})$, and G_{i_2} can derive d_{j_2} in the similar way. Consider the whole clique $\mathcal{C}$. When the MSS broadcasts the encoded packet $p = d_{\mathcal{C}(1)} \oplus d_{\mathcal{C}(2)} \oplus \cdots \oplus d_{\mathcal{C}(|D_{\mathcal{C}}|)}$, according to Rules 1 and 2, each group $G_i \in G_{\mathcal{C}}$ should have cached all data items in $D_{\mathcal{C}}$ except the requested data item d_j. Therefore, G_i can obtain d_j by decoding data packet p.

In the following, we prove that with the above polynomial-time reduction, Graph $\mathcal{G} = (V, E)$ has a set of w cliques such that each vertex in V is in at least one of the selected cliques if and only if there is a group-based encoding scheme where all group requests can be satisfied in w MSS broadcasts.

Suppose $\mathcal{G}$ has a number of w cliques $\mathscr{C} = \{\mathscr{C}_1, \mathscr{C}_2, \ldots, \mathscr{C}_w\}$, and each vertex in V is in at least one of cliques in $\mathscr{C}$. The set of vertices included in clique $\mathscr{C}_i$ $(1 \leq i \leq w)$ is $V_{\mathscr{C}_i} = \{v_{i_1j_1}, v_{i_2j_2}, \ldots, v_{i_{|V_{\mathscr{C}_i}|}j_{|V_{\mathscr{C}_i}|}}\}$. From the above discussion, we know that if the MSS broadcasts the data packet $\mathcal{P} = d_{j_1} \oplus d_{j_2} \oplus \cdots \oplus d_{j_{|V_{\mathscr{C}_i}|}}$, all group requests in $\mathscr{C}_i$ can be satisfied. As the MSS keeps broadcasting data packets of other cliques, after w broadcasts, all group requests could be satisfied. Hence, all group requests can be satisfied in w MSS broadcasts if Graph $\mathcal{G} = (V, E)$ has a set of w cliques such that each vertex in V is in at least one of the selected cliques.

Conversely, suppose all group requests can be satisfied in w MSS broadcasts, and the broadcast data packet set is $\mathcal{P} = \{P_1, P_2, \ldots, P_{|\mathcal{P}|}\}$. Each data packet is $P_j = d_{j_1} \oplus d_{j_2} \oplus \cdots \oplus d_{j_{|P_j|}}$. We can construct a clique for each data packet in $\mathcal{G}$ based on Rules 1 and 2. Therefore we can construct a total number of w cliques. After w MSS broadcasts, if there is at least one vertex not being covered by the w cliques, it means there is at least one group request not being satisfied by MSS broadcasts, which contradicts the assumption. Hence, Graph $\mathcal{G} = (V, E)$ has a set of w cliques such that each vertex in V is in at least one of the selected cliques if all group requests can be satisfied in w MSS broadcasts.

To sum up, there is a solution for an instance of the MCC problem if and only if there is a solution for the instance of the MCEE problem. Therefore, the MCEE problem is NP-hard. □

Algorithm 1 NCM-S: MSS scheduling and encoding process at each broadcast time unit

Step 1: Calculate the most rewarding data packet at each broadcast time unit
Fetch the request, i.e., $Q_i = \{d_j\}$, at the head of the request queue $\mathcal{Q}$, where Q_i is the request issued by the MH m_i and d_j is the data item requested;
$MaxNum \leftarrow 0$;
$ClientsToSatisfy \leftarrow \phi$;
$DataToBroadcast \leftarrow d_j$;
for each $Q_k = \{d_l\} \in \mathcal{Q}$ **do**
  if $d_l = d_j$ **then**
    $MaxNum \leftarrow MaxNum + 1$;
    $ClientsToSatisfy \leftarrow ClientsToSatisfy \cup \{m_k\}$;
  end if
end for
Suppose the cache content of m_i is C_i' in the MSS view;
for each $d_p \in C_i'$ **do**
  $ClientsToSatisfyEncode \leftarrow \phi$;
  $MaxNumEncode \leftarrow 0$;
  for each $Q_r = \{d_t\} \in \mathcal{Q}$ **do**
    Suppose the cache content of m_r is C_r' in the MSS view;
    if $d_j = d_t$ **and** $d_p \in C_r'$ **then**
      $MaxNumEncode \leftarrow MaxNumEncode + 1$;
      $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
    else if $d_p = d_t$ **and** $d_j \in C_r'$ **then**
      $MaxNumEncode \leftarrow MaxNumEncode + 1$;
      $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
    end if
  end for
  if $MaxNumEncode > MaxNum$ **then**
    $MaxNum \leftarrow MaxNumEncode$;
    $ClientsToSatisfy \leftarrow ClientsToSatisfyEncode$;
    $DataToBroadcast \leftarrow d_j \oplus d_p$;
  end if
end for
Step 2: Broadcast the data packet constructed in Step 1
Broadcast the data packet in $DataToBroadcast$ and prefix the MH identifiers in $ClientsToSatisfy$;

5. Proposed Schemes

5.1. Scheme 1: Network Coding at MSS (NCM)

5.1.1. Objective

By implementing network coding at the MSS, it is possible for the MSS to broadcast an encoded data packet to satisfy multiple requests for different data items in one broadcast time unit, thereby improving the bandwidth efficiency of downlink communication channel. The system performance is expected to be improved in terms of data access latency with enhanced bandwidth efficiency of downlink communication channel. With the above motivation, we propose a scheme called NCM, which includes two algorithms NCM-S and NCM-C.

5.1.2. Algorithms

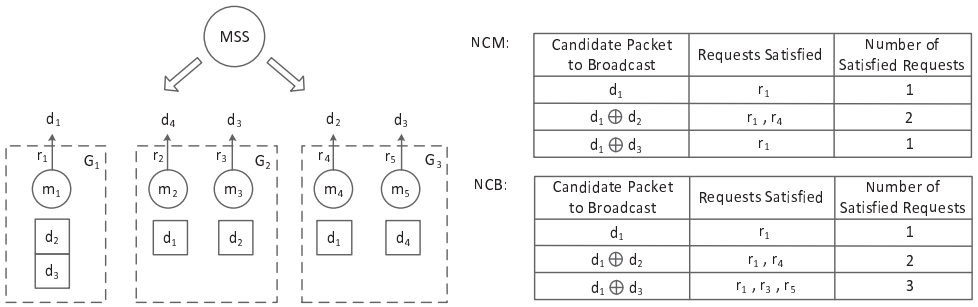
NCM-S: This algorithm is running at the MSS and is executed in every broadcast time unit to schedule, encode and broadcast requested data items to MHs. NCM-S consists of two steps. In Step 1, the MSS constructs the most rewarding data packet in terms of number of requests that can be satisfied. The data packet may consist of only one data item or multiple encoded data items. In Step 2, the MSS broadcasts the data packet constructed in Step 1. When a request arrives at the MSS, it is inserted into the request queue. In order to illustrate the impact of network coding

Algorithm 2 NCM-C: A requesting MH m_i's action when receiving data packet $DataToBroadcast$ broadcast by the MSS

1: Suppose the request of m_i is $Q_i = \{d_j\}$, pending in the request queue $\mathcal{Q}$. The cache content of m_i is C_i. MHs intended to be served by the current broadcast is $ClientsToSatisfy$;
2: **if** $DataToBroadcast$ is an unencoded packet, i.e., d_k **then**
3: **if** $d_j = d_k$ **then**
4: Obtain the requested data item d_j and and notify the MSS to remove Q_i;
5: **else**
6: Ignore d_k;
7: **end if**
8: **else if** $DataToBroadcast$ is an encoded packet, i.e., $d_k \oplus d_p$ **then**
9: **if** $d_j = d_k$ **and** $d_p \in C_i$ **then**
10: Obtain the requested data item d_j by $d_p \oplus (d_k \oplus d_p)$ and notify the MSS to remove Q_i;
11: **else if** $d_j = d_p$ **and** $d_k \in C_i$ **then**
12: Obtain the requested data item d_j by $d_k \oplus (d_k \oplus d_p)$ and notify the MSS to remove Q_i;
13: **else if** $m_i \in ClientsToSatisfy$ **then**
14: Notify the MSS to update C_i';
15: **else**
16: Ignore $d_k \oplus d_p$;
17: **end if**
18: **end if**

on the system performance, we adopt first-come-first-served (FCFS) scheduling policy at the MSS and encode at most two data items in a packet. Previous work showed that to encode more data items can provide even more performance gain [6]. The MSS seeks to maximize the number of requests to be satisfied in one broadcast time unit, on the basis of satisfying the request pending at the head of the request queue first. Note that a request is satisfied if the data packet to be broadcast consists of the requested data item in an unencoded form or the requested data item is encoded with another data item which exists in the requesting MH's cache. A requested data item will be encoded with another requested data item only if the encoded packet can serve more requests. The pseudocode of NCM-S is shown in Algorithm 1. Specifically, in Step 1 (lines 1-31), the MSS first fetches a request from the head of the request queue (line 2) and then counts the number of requests for the data item same as the fetched request (lines 2-11). Next, the MSS searches for the maximum number of requests which can be simultaneously satisfied if the MSS broadcasts an encoded data packet (lines 12-25). Lastly, the MSS compares the above two values and constructs the data packet which can satisfy more requests (lines 26-30).

NCM-C: This algorithm is running at MHs and is executed to receive and decode data packets broadcast by the MSS in downlink communication channel. To encode requested data items, the MSS has to maintain a view of the cache content of all MHs in the system. This can be done by tracing the requests submitted by MHs and data items broadcast to serve the requests because it is common for MHs to store recently requested data items in their caches for possible future retrieval. However, cache size of MHs is limited and MHs may also cache data items obtained from their peers, so the cache contents may change from time to time, leading to inconsistency between the MSS view and actual cache content of MHs. We adopt a lazy approach to minimize the maintenance cost of the view. The MSS schedules and encodes according to its own cache view. When the MSS broadcasts a packet, it prefixes a list of MH identifiers of intended recipients who should be able to retrieve the requested data items. If an MH in the list can obtain the required data item, it sends an acknowledgment to the MSS to remove its request pending in the request queue. Otherwise, if the packet is encoded and the MH cannot decode the packet, it implies that the MSS view is inconsistent with actual cache content of the MH. Then, the MH will send identifiers of its cached data items to the MSS for updating its view. The request of the MH remains in the request queue waiting for the MSS's next schedule. The pseudocode of



NCM:

Candidate Packet to Broadcast	Requests Satisfied	Number of Satisfied Requests
d_1	r_1	1
$d_1 \oplus d_2$	r_1 , r_4	2
$d_1 \oplus d_3$	r_1	1

NCB:

Candidate Packet to Broadcast	Requests Satisfied	Number of Satisfied Requests
d_1	r_1	1
$d_1 \oplus d_2$	r_1 , r_4	2
$d_1 \oplus d_3$	r_1 , r_3 , r_5	3

Figure 4: Running Examples of NCM and NCB

NCM-C is shown in Algorithm 2. Specifically, if the data packet consists of the requested data item in an unencoded form, the request is satisfied (lines 2-7). Otherwise, if the MH can decode the data packet to retrieve the requested data item, the request is also satisfied (lines 8-18).

A running example of NCM is given in Figure 4. The five MHs are divided into three TCGs: m_1 is in G_1; m_2 and m_3 are in G_2; m_4 and m_5 are in G_3. At a certain broadcast time unit, five requests are pending in the request queue of the MSS, namely r_1 (m_1 is requesting d_1), r_2 (m_2 is requesting d_4), r_3 (m_3 is requesting d_3), r_4 (m_4 is requesting d_2) and r_5 (m_5 is requesting d_3). At the MHs, m_1 caches d_2 and d_3, m_2 caches d_1, m_3 caches d_2, m_4 caches d_1, and m_5 caches d_4. Algorithm 1 (NCM-S) running at the MSS searches for the most rewarding data packet to broadcast (Step 1). The MSS fetches r_1 at the head of the request queue. There are three candidate data packets, i.e., d_1, $d_1 \oplus d_2$, and $d_1 \oplus d_3$, because any one of the three packets can satisfy r_1. If d_1 is broadcast, one request (r_1) can be satisfied. If $d_1 \oplus d_2$ is broadcast, two requests (r_1 and r_4) can be satisfied. If $d_1 \oplus d_3$ is broadcast, one request (r_1) can be satisfied. So the MSS choose to broadcast $d_1 \oplus d_2$ (Step 2), which can simultaneously satisfy the maximum number of requests. Algorithm 2 (NCM-C) running at each of the requesting MHs receives the data packet, tries to decode it and, if successful, obtains the requested data item. In this example, when m_1 receives $d_1 \oplus d_2$, it obtains d_1 by computing $d_2 \oplus (d_1 \oplus d_2)$. Similar operation can be performed by m_4 to obtain its own requested data item.

5.2. Scheme 2: Network Coding at Both MSS and MH (NCB)

5.2.1. Objective

We have applied network coding to the MSS hereinbefore to improve the broadcast efficiency of downlink communication channel. Next, we seek to further improve information sharing by exploiting network coding also at MHs. Since MHs in the same TCG can share their cached data items, the caches of MHs in the same TCG constitute a logically larger group cache. We consider this information sharing among TCG members not only when the MSS encodes data packets, but also when requesting MHs decode data packets to obtain their desired data items. On this basis, we propose a scheme called NCB, which includes two algorithms NCB-S and NCB-C.

5.2.2. Algorithms

NCB-S: This algorithm is running at the MSS and is executed in every broadcast time unit to make encoding decisions. Unlike NCM-S, the MSS examines the cache information of a group of MHs rather than individual MHs. As depicted in Algorithm 3, NCB-S consists of two steps. In Step 1, the MSS constructs the most rewarding data packet at each broadcast time unit. In Step 2, the MSS broadcasts the data packet constructed in Step 1 and mark the corresponding

Algorithm 3 NCB-S: MSS scheduling and encoding process at each broadcast time unit

Step 1: Construct the most rewarding data packet at each broadcast time unit
Try from the head of the request queue $\mathcal{Q}$, and fetch the first request which is unmarked, i.e., $Q_i^u = \{d_j\}$, where Q_i^u is the request issued by the MH m_i which belongs to the group G_u, and d_j is the data item requested;
$MaxNum \leftarrow 0$;
$ClientsToSatisfy \leftarrow \phi$;
$DataToBroadcast \leftarrow d_j$;
$RequestsToMark \leftarrow \phi$;
for each $Q_k^v = \{d_l\} \in \mathcal{Q}$ **and** Q_k^v is unmarked **do**
  if $d_l = d_j$ **then**
    $MaxNum \leftarrow MaxNum + 1$;
    $ClientsToSatisfy \leftarrow ClientsToSatisfy \cup \{m_k\}$;
    $RequestsToMark \leftarrow RequestsToMark \cup \{Q_k^v\}$;
  end if
end for
Suppose the group cache content of G_u is GC'_u in the MSS view;
for each $d_p \in GC'_u$ **do**
  $ClientsToSatisfyEncode \leftarrow \phi$;
  $RequestsToMarkEncode \leftarrow \phi$;
  $MaxNumEncode \leftarrow 0$;
  for each $Q_r^w = \{d_t\} \in \mathcal{Q}$ **and** Q_r^w is unmarked **do**
    Suppose the cache content of G_w is GC'_w in the MSS view;
    if $d_j = d_t$ **and** $d_p \in GC'_w$ **then**
      $MaxNumEncode \leftarrow MaxNumEncode + 1$;
      $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
      $RequestsToMarkEncode \leftarrow RequestsToMarkEncode \cup \{Q_r^w\}$;
    else if $d_p = d_t$ **and** $d_j \in GC'_w$ **then**
      $MaxNumEncode \leftarrow MaxNumEncode + 1$;
      $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
      $RequestsToMarkEncode \leftarrow RequestsToMarkEncode \cup \{Q_r^w\}$;
    end if
  end for
  if $MaxNumEncode > MaxNum$ **then**
    $MaxNum \leftarrow MaxNumEncode$;
    $ClientsToSatisfy \leftarrow ClientsToSatisfyEncode$;
    $RequestsToMark \leftarrow RequestsToMarkEncode$;
    $DataToBroadcast \leftarrow d_j \oplus d_p$;
  end if
end for
Step 2: Broadcast the data packet constructed in Step 1 and mark the corresponding requests
Broadcast the data packet in $DataToBroadcast$ and prefix the MH identifiers in $ClientsToSatisfy$;
Mark all requests in $RequestsToMark$;

requests. Initially, all requests pending in the request queue are unmarked. Marking of a request is used to indicate that the request has been served and the MSS is waiting for the acknowledgment of the requesting MH. Unlike the NCM scheme, NCB allows the requesting MH to seek help from peers to decode the encoded packet, which may take longer time for the MSS to receive the acknowledgment. So, a marked request will not be considered for scheduling until an acknowledgment is received from the requesting MH to remove or unmark the request. Specifically, in Step 1 (lines 1-37), the MSS fetches the first unmarked request from the head of the request queue (line 2) and then counts the number of requests for the data item same as the fetched request (lines 3-13). Next, the MSS searches for the maximum number of requests which can be simultaneously satisfied if the MSS broadcasts an encoded data packet (lines 14-30). Note that a request can also be satisfied in the NCB scheme if the requested data item is encoded with another data item which exists in the group cache of the group which the requesting MH belongs to. Lastly, the MSS compares the above two values and constructs the data packet which can satisfy more requests (lines 31-36). Then, the MSS broadcasts the data packet, puts the identifiers of MHs whose requests should be satisfied by the data packet into a satisfying list, and marks all requests corresponding to this data packet (lines 38-40).

Algorithm 4 NCB-C: A requesting MH m_i's action when receiving data packet $DataToBroadcast$ broadcast by the MSS

1: Suppose the request of m_i is $Q_i^u = \{d_j\}$, pending in the request queue $\mathcal{Q}$. The group which m_i belongs to is G_u. The cache content of m_i is C_i. MHs intended to be served by the current broadcast is $ClientsToSatisfy$;
2: **if** $DataToBroadcast$ is an unencoded packet, i.e., d_k **then**
3: **if** $d_j = d_k$ **then**
4: Obtain the requested data item d_j and and notify the MSS to remove Q_i^u;
5: **else**
6: Ignore d_k;
7: **end if**
8: **else if** $DataToBroadcast$ is an encoded packet, i.e., $d_k \oplus d_p$ **then**
9: **if** $m_i \in ClientsToSatisfy$ **then**
10: **if** $d_j = d_k$ **and** $d_p \in C_i$ **then**
11: Obtain the requested data item d_j by $d_p \oplus (d_k \oplus d_p)$ and notify the MSS to remove Q_i^u;
12: **else if** $d_j = d_p$ **and** $d_k \in C_i$ **then**
13: Obtain the requested data item d_j by $d_k \oplus (d_k \oplus d_p)$ and notify the MSS to remove Q_i^u;
14: **else if** $d_j = d_k$ **and** $d_p \notin C_i$ **then**
15: Seek help from peers;
16: **if** m_i can find and then retrieve d_p from peers **then**
17: Obtain the requested data item d_j by $d_p \oplus (d_k \oplus d_p)$, and notify the MSS to remove Q_i^u;
18: **else**
19: Notify the MSS to update GC_u' and unmark Q_i^u;
20: **end if**
21: **else if** $d_j = d_p$ **and** $d_k \notin C_i$ **then**
22: Seek help from peers;
23: **if** m_i can find and then retrieve d_k from peers **then**
24: Obtain the requested data item d_j by $d_k \oplus (d_k \oplus d_p)$, and notify the MSS to remove Q_i^u;
25: **else**
26: Notify the MSS to update GC_u' and unmark Q_i^u;
27: **end if**
28: **end if**
29: **else**
30: Ignore $d_k \oplus d_p$;
31: **end if**
32: **end if**

NCB-C: This algorithm is running at requesting MHs and is executed every time when a broadcast data packet appears in the downlink communication channel. Like NCM-C, NCB-C is used for MHs to receive and decode data packets broadcast by the MSS. Specifically, if the data packet consists of the requested data item in an unencoded form, the request is satisfied (lines 2-7). Otherwise, if a requesting MH in the satisfying list receives an encoded data packet broadcast

from the MSS and it has another data item in its own cache, it decodes the packet and obtains the requested data item (lines 8-13). If it cannot decode the packet by itself, it will seek help from its peers (lines 14-28). The requesting MH broadcasts a request to its peers for the missing data item to decode the encoded packet. If it receives any reply from its peers within the timeout period, it then retrieves the missing data item, decodes the packet, and obtains the requested data item (lines 16-17 and 23-24). In any case if the request is satisfied, the requesting MH will send an acknowledgment to the MSS to remove the request from the request queue. If the timeout period expires and no peers reply to it, it implies that the MSS view of the group cache is inconsistent with the actual group cache contents. The requesting MH will send an acknowledgment to the MSS to unmark the request and broadcast a message to its peers requesting them to send identifiers of their cached data items to the MSS for updating its view (lines 19 and 26). The process is presented in Algorithm 4.

A running example of NCB is given in Figure 4. The five requests pending in the request queue of the MSS are issued by five MHs. The group cache of G_1 stores d_2 and d_3, the group cache of G_2 stores d_1 and d_2, and the group cache of G_3 stores d_1 and d_4. Algorithm 3 (NCB-S) running at the MSS constructs the most rewarding data packet to broadcast (Step 1). The MSS fetches r_1 at the head of the request queue. There are three candidate data packets, i.e., d_1, $d_1 \oplus d_2$, and $d_1 \oplus d_3$, because any one of the three packets can satisfy r_1. Note that different from NCM, the MSS in NCB selects data packet to broadcast based on the group concept. In other words, the MSS considers the possibility for the requesting MH to decode the data packet using data item from its own cache and also from the group cache. If d_1 is broadcast, one request (r_1) can be satisfied. If $d_1 \oplus d_2$ is broadcast, two requests (r_1 and r_4) can be satisfied. If $d_1 \oplus d_3$ is broadcast, three requests (r_1, r_3 and r_5) can be satisfied. So, the MSS chooses to broadcast $d_1 \oplus d_3$ (Step 2), which can simultaneously satisfy the maximum number of requests. Algorithm 4 (NCB-C) running at each of the requesting MHs receives the data packet, tries to decode it and, if successful, obtains the requested data item. In this example, when m_3 receives $d_1 \oplus d_3$, although it does not cache d_1, it can get d_1 from its peer m_2 and finally obtain d_3 by computing $d_1 \oplus (d_1 \oplus d_3)$. Similar operation can be performed by m_1 and m_5 to obtain their own requested data items.

6. Performance Evaluation

6.1. Simulation Model

We construct a simulation model as described in Section 3 using CSIM [27]. In the model, there is an MSS and $NumMH$ MHs. Each MH has a cache of size $CacheSize$. MHs move in an area of 3000 m $\times$ 3000 m. The service range of the MSS covers the whole area. The MSS broadcasts data items to MHs via downlink communication channel with a bandwidth of $BW_{downlink}$. MHs send requests to the MSS via uplink communication channel with a bandwidth of BW_{uplink}. The P2P communication channel is used for an MH to communicate with its peers with a bandwidth of BW_{P2P}. The primary parameters and their default settings in the simulation are shown in Table 3.

6.1.1. Client Model

The MHs move in the service area of the MSS. MHs can communicate with each other within the transmission range of $TranRange$. MHs can be divided into several motion groups, each with $GroupSize$ MHs. The mobility of MHs in a group is based on the *reference point group mobility model* [14], in which there is a logical motion center as the reference for the whole group. The group mobility is based on the *random waypoint mobility model* [5]. The logical group center randomly

Table 3: System Parameters

Parameter	Default	Range	Description
$NumMH$	300	-	Number of MHs
$GroupSize$	3	1 - 20	Number of MHs in each motion group
$NumData$	1000	-	Number of data items in the database
$TranRange$	100 m	-	Wireless transmission range of each MH
$CacheSize$	100 data items	20 - 180	The cache size of each MH
θ	0.6	0 - 1.0	Zipf distribution skewness parameter
$DataSize$	32 kb	-	Data item size
$ReqSize, RepSize, RetSize$	512 b	-	Size of a request, a *reply* and a *retrieve* message
$BW_{downlink}$	10 Mbps	-	Bandwidth of the MSS downlink communication channel
BW_{uplink}	1 Mbps	-	Bandwidth of the MSS uplink communication channel
BW_{P2P}	5 Mbps	-	Bandwidth of the P2P channel among MHs
v_{min}	1 m/s	-	The minimum speed at which MHs move
v_{max}	5 m/s	-	The maximum speed at which MHs move
φ, φ'	10, 3	-	Timeout parameters

chooses its destination and moves towards it at a speed from a uniform distribution U(v_{min}, v_{max}). When it reaches the destination, it rests for a second and randomly chooses another destination. Then it moves towards the destination and changes the speed according to the same uniform distribution.

The request model is a closed model, in which an MH can request another data item only after its last request has been satisfied. The size of a request is $ReqSize$. The closed model is common in the reality. There are many applications that cannot decide which data item to request next until its last request is satisfied. The *reply* and *retrieve* message sizes of the COCA communication protocol are $RepSize$ and $RetSize$ respectively. The timeout parameters are φ and φ'. The data access pattern follows the Zipf distribution, with a skewness parameter θ, where $0 \leq \theta \leq 1$. When θ equals 0, it is a uniform distribution. As θ gets larger, the accesses to data items get more skewed. The duration between the time when a request is satisfied and the time when the next request is submitted follows an exponential distribution with a mean of one second.

6.1.2. Server Model

There is one MSS in the system, which serves the MHs in the area of 3000 m × 3000 m. The database in the MSS maintains $NumData$ data items and the size of each data item is $DataSize$. Outstanding requests are pended in the request queue at the MSS. In each broadcast tick, the MSS schedules the request with FCFS policy and encodes the requested data items with the purpose of satisfying as many requests as possible.

6.1.3. Power Consumption Model

Each MH is equipped with two wireless network interface cards, one of which is used to communicate with the MSS, while the other is used to communicate with its peers. In P2P communication, the power consumption measurement model uses a set of linear formulas to measure the power consumption of the source MH, S, the destination MH, D, and other *remaining* MHs residing in the transmission range of the source MH, S_R, and the destination MH, D_R [12]. In communication between MHs and the MSS, the power consumption measurement model uses a set of linear formulas to measure the power consumption of the source MH, S, when S sends

a message to the MSS, and the destination MH, D, when D receives a message from the MSS. The power consumption of P2P point-to-point communication, broadcast communication, and communication between MHs and the MSS equals $(v \times b) + f$, where f is the fixed setup cost for a transmission, and v is the variable power consumption on message size in bytes (b), and different communication approaches have different values of v and f as shown in Table 4, 5 and 6, respectively.

6.2. Performance Metrics

We use the following metrics to evaluate the performance of the coding-based cooperative caching schemes.

- *Server Request Ratio*: It is the percentage of number of server requests to total number of requests:

$$\frac{Total\ Number\ of\ Server\ Requests}{Total\ Number\ of\ Requests} \times 100\%$$

  The objectives of COCA and GroCoca are to increase global cache hit ratio and decrease server request ratio.

- *Encoded Packet Broadcast Ratio*: It is the percentage of number of encoded packets broadcast by the MSS to total number of packets broadcast by the MSS:

$$\frac{Total\ Number\ of\ Encoded\ Packets\ Broadcast\ by\ the\ MSS}{Total\ Number\ of\ Packets\ Broadcast\ by\ the\ MSS} \times 100\%$$

  It describes the effectiveness of network coding in the system. A higher encoded packet broadcast ratio means that network coding has been applied more frequently.

- *Mean Request Queue Length*: It is the mean number of requests pending in the request queue. A shorter mean request queue length reflects a higher efficiency of the MSS in scheduling and serving requests in the request queue.

- *Average Server Request Latency*: It is the waiting time from the moment an MH sends a server request to the MSS to the time when its request is finally satisfied. A shorter average server request latency means that the MSS is more responsive.

- *Average Data Access Latency*: It is the average time period from the moment an MH wants a data item to the time when the desired data item is obtained. A shorter average data access latency indicates a higher efficiency of the whole system in satisfying MHs' requests.

- *Power Consumption*: It illustrates the power consumed by MHs.

6.3. Simulation Results

In this section, we present the performance results of the two proposed coding-based cooperative caching schemes NCM and NCB. For comparison purpose, we also include performance of a number of other schemes. To better indicate the different combinations of cooperative caching and network coding in the schemes, NCM is renamed as GCC-NC (GroCoca with network coding at MSS) and NCB is renamed as GCC-NC-G (GroCoca with network coding at both MSS and MH). Other schemes are non-cooperative caching scheme (denoted as NCC), non-cooperative caching scheme with network coding (denoted as NCC-NC), COCA scheme (denoted as CC),

Table 4: Power Consumption Parameters in P2P Point-to-point Communication

Condition	v	f
$m = S$	1.9	454
$m = D$	0.5	356
$m \in S_R \wedge m \in D_R$	0	70
$m \in S_R \wedge m \notin D_R$	0	24
$m \notin S_R \wedge m \in D_R$	0	56

Table 5: Power Consumption Parameters in P2P Broadcast Communication

Condition	v	f
$m = S$	1.9	266
$m \in S_R$	0.5	56

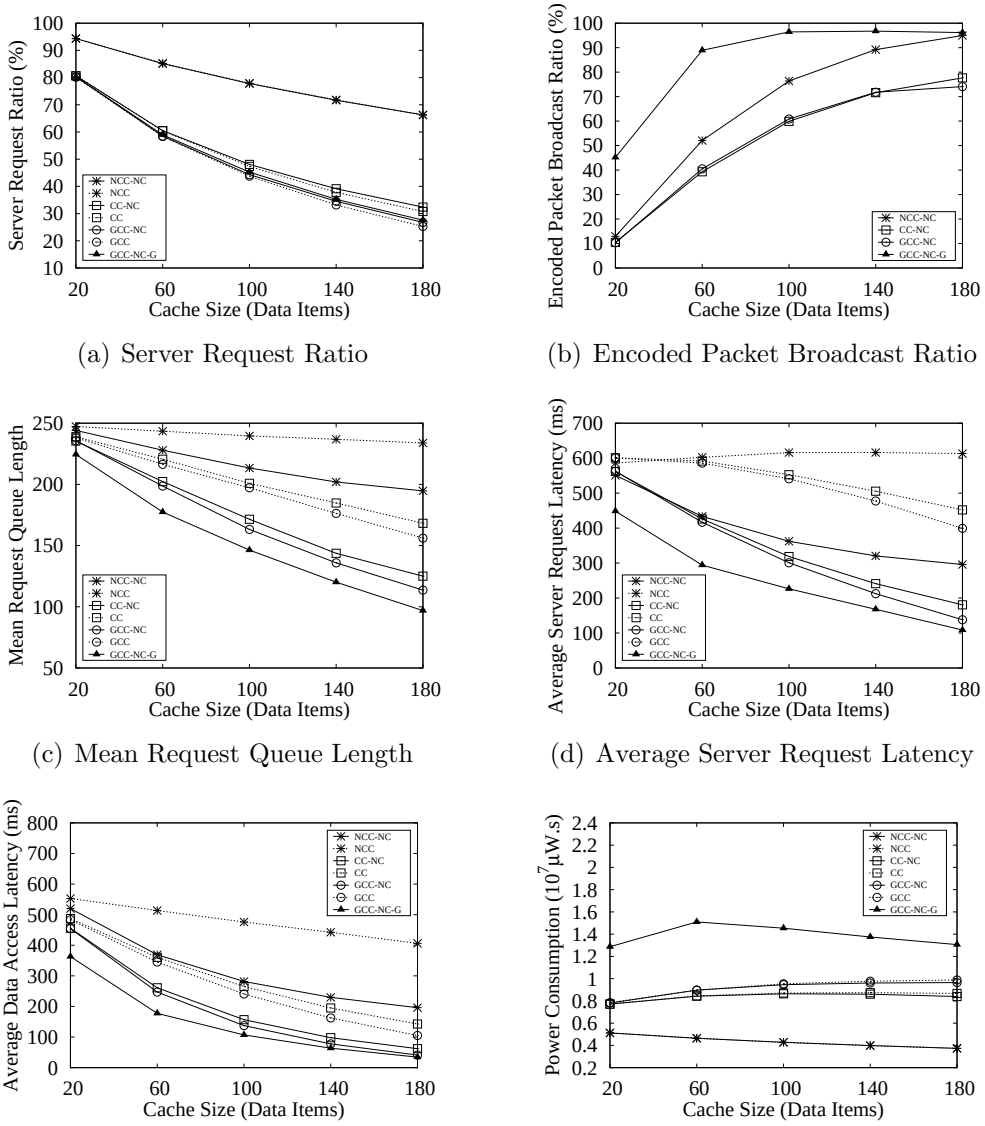
COCA scheme with network coding (denoted as CC-NC), and GroCoca (denoted as GCC). The results are collected after the system reaches a steady state, in which all caches of MHs are filled with data items. The experiment continues until each MH generates over 2000 requests after the warmup period.

6.3.1. Effect of Cache Size

First, we study the effect of cache size on system performance by increasing the cache size from 20 to 180 data items. Figure 5(a) shows that the server request ratio decreases as the cache size increases. This is because both local cache hit ratio and global cache hit ratio increase. The cache hit ratios increase because a larger cache provides a higher chance for MHs to obtain data items locally or from peers. NCC-NC and NCC have the highest server request ratio because all requests go to the MSS in case of local cache misses. GCC-NC, GCC-NC-G and GCC have lower server request ratio than CC-NC and CC because GCC-NC and GCC can further improve data availability within TCGs. In Figure 5(b), the encoded packet broadcast ratio increases because larger cache provides more opportunities for network coding. In particular, the encoded packet broadcast ratio of GCC-NC-G is the highest because the size of a group cache is much larger than the cache size of individual MHs. As depicted in Figure 5(c), the schemes with network coding have shorter request queue length than their non-coding versions because network coding can satisfy more requests in one broadcast time unit and thus abate the broadcast pressure. This also leads to less average server request latency of the schemes with network coding as shown in Figure 5(d). In Figure 5(e), the average data access latency decreases, because the time cost is much lower for an MH to obtain a data item from its local cache or from its peers than from the MSS. CC utilizes cooperative caching, so it has less average data access latency than NCC. GCC outperforms CC because GCC increases the chance for an MH to obtain the required data

Table 6: Power Consumption Parameters in Communication Between MHs and MSS

Condition	v	f
$m = S$	1.9	454
$m = D$	0.5	56



(a) Server Request Ratio

(b) Encoded Packet Broadcast Ratio

(c) Mean Request Queue Length

(d) Average Server Request Latency

(e) Average Data Access Latency

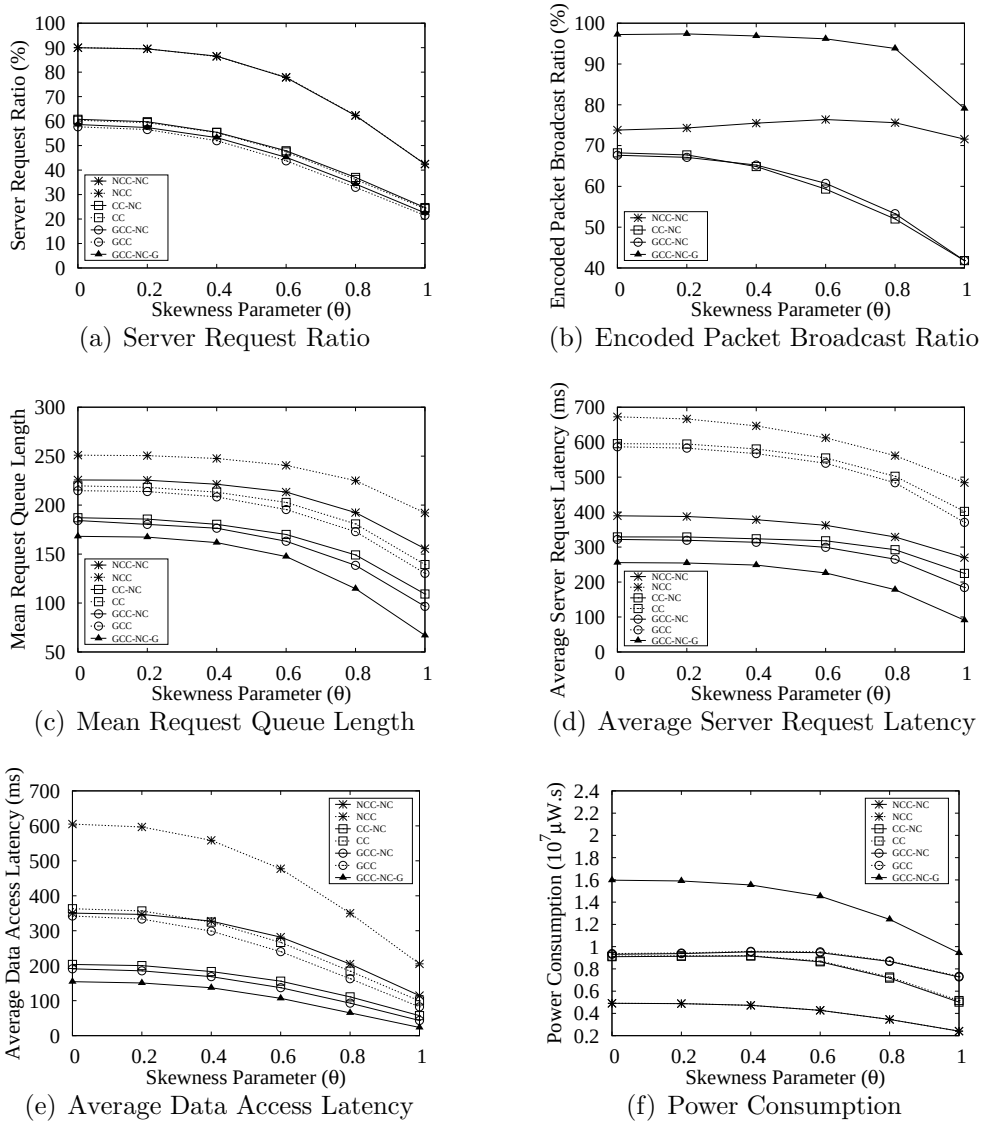
(f) Power Consumption

Figure 5: Effect of Cache Size on System Performance

item in its TCG. All the schemes with network coding have better average data access latency than the non-coding versions because network coding can significantly reduce the server request time. GCC-NC-G further reduces the average data access latency because both encoding and decoding processes are more flexible. On one hand, GCC-NC-G provides more opportunities for the MSS to encode a requested data item with any data item in a large group cache. On the other hand, GCC-NC-G allows requesting MHs to retrieve data items from peers to decode their requested data items. However, as shown in Figure 5(f), since additional searches for data items from peers for decoding is quite power consuming, GCC-NC-G consumes more power. Note that other schemes with network coding consume no higher power than the non-coding versions.

6.3.2. Effect of Access Pattern

Next, we study the effect of data access pattern on system performance by increasing the skewness parameter θ from 0 to 1. As θ gets larger, MHs have more common and narrower interest



(a) Server Request Ratio (b) Encoded Packet Broadcast Ratio

(c) Mean Request Queue Length (d) Average Server Request Latency

(e) Average Data Access Latency (f) Power Consumption

Figure 6: Effect of Access Pattern on System Performance

so the chance for an MH to find desired data items in its local cache and its peers' caches becomes higher. Therefore, the number of requests sent to the MSS decreases, leading to decrease of the server request ratio as shown in Figure 6(a). GCC-NC, GCC-NC-G and GCC have the lowest server request ratio because they can further improve data availability within TCGs. Figure 6(b) shows that the encoded packet broadcast ratio decreases as θ gets larger, which is because skewed data access patterns degrades the efficiency of network coding. Figure 6(c) shows that the schemes with network coding have shorter request queue length than their non-coding versions. So the schemes with network coding have less average server request latency as depicted in Figure 6(d). In Figure 6(e), the average data access latency decreases as θ increases, because the server request ratio decreases and the time cost is much lower for an MH to obtain a data item from its local cache or from its peers than from the MSS. All the schemes with network coding have better average data access latency than the non-coding versions because network coding can significantly reduce the server request time. Particularly, GCC-NC-G has the best performance in terms of average data access latency among all the schemes. Figure 6(f) shows that the power consumption

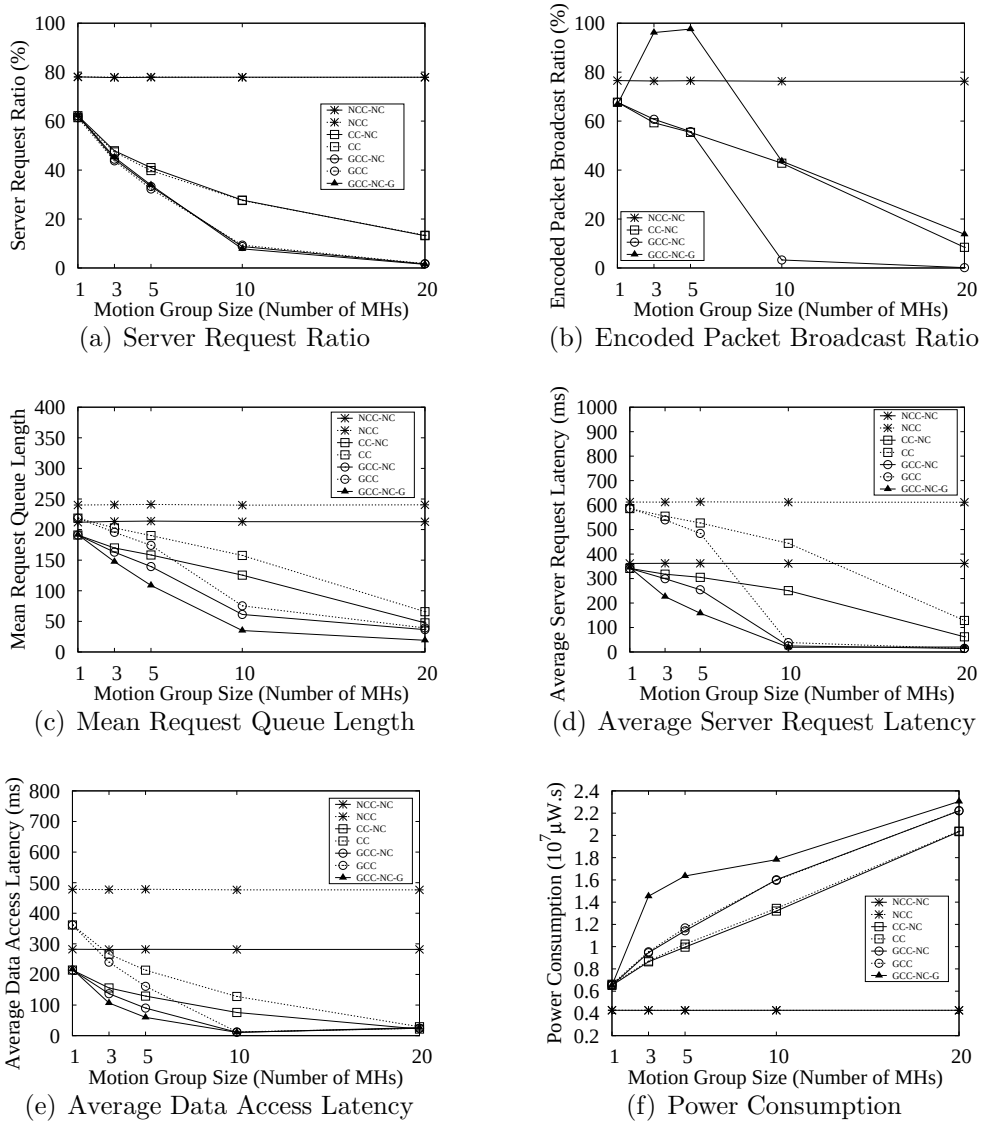


Figure 7: Effect of Motion Group Size on System Performance

drops as θ increases. As the data access pattern gets more skewed, MHs have a higher chance to find their desired data items in local caches and save some power to communicate with their peers. Again, GCC-NC-G requires more power for enabling decoding among peers and schemes with network coding consume no higher power than the non-coding versions.

6.3.3. Effect of Motion Group Size

Finally, we study the effect of motion group size on system performance by increasing the motion group size from 1 to 20. When the motion group size is equal to one, the mobility model is equivalent to an individual *random waypoint mobility model*. The values of all metrics of NCC and NCC-NC remain unchanged as the motion group size gets larger, as shown in Figure 7. This is because these two schemes do not exploit cooperative caching, and thus the motion group size has no effect on their performance. The performance of CC, CC-NC, GCC, GCC-NC and GCC-NC-G is better with increasing motion group size in terms of server request ratio, mean request queue length, average server request latency and average data access latency, as shown

in Figure 7(a), 7(c), 7(d) and 7(e) respectively. Larger motion group size provides higher chance for MHs to obtain their desired data items from their peers. When the group size increases, the number of groups decreases. Since network coding will only encode a requested data item with a data item exists in another group, fewer number of groups will reduce the opportunities for network coding as shown in 7(b). The schemes with network coding have better performance than their non-coding versions, because network coding enhances broadcast efficiency of the MSS. Particularly, GCC-NC-G has the best performance in terms of average data access latency among all the schemes. As mentioned in Section 6.1.3, MHs residing in the transmission range of the source MH and the destination MH also consume certain power even if they are not the sending or the receiving MHs. Figure 7(f) shows that as the motion group size gets larger, more MHs are involved in the process of global cache search, leading to the rise of power consumption. Again, schemes with network coding consume no higher power than the non-coding versions.

7. Conclusion

In this paper, we have introduced a novel cooperative caching architecture by incorporating network coding into on-demand broadcast environments. MHs who are not neighboring peers can collaborate indirectly by utilizing their own cached data items. In addition, we formulate the MCEE problem and prove that it is NP-hard. Further, two cooperative caching schemes NCM and NCB are proposed. In NCM, the MSS can encode a requested data item with another data item to maximize the number of requests that can be served simultaneously. NCB allows more flexibility in both encoding and decoding. In particular, the MSS can encode a requested data item with another data item even it is not cached by the requesting MH. The requesting MH can retrieve a data item from its peers for decoding. An extensive experimental results have demonstrated that coding-based schemes outperform their respective non-coding versions in terms of enhancing bandwidth efficiency and reducing data access latency. Specifically, NCB, which exploits the synergy between network coding and cooperative caching at both MSS and MH, has the best performance among all the schemes.

Acknowledgement

The work described in this paper was partially supported by a grant from City University of Hong Kong (Project No. 7004412), and by the National Natural Science Foundation of China under Grant No. 61572088.

References

[1] R. Ahlswede, N. Cai, S.-Y. Li, R. W. Yeung, Network information flow, IEEE Trans. Information Theory 46 (4) (2000) 1204–1216.

[2] G. Ali, Y. Meng, V. Lee, K. Liu, E. Chan, Performance improvement in applying network coding to on-demand scheduling algorithms for broadcasts in wireless networks, in: International Multi-Conference on Computing in the Global Information Technology (ICCGI), 2014.

[3] R. Bassoli, H. Marques, J. Rodriguez, K. W. Shum, R. Tafazolli, Network coding theory: A survey, Communications Surveys & Tutorials 15 (4) (2013) 1950–1978.

[4] Y. Birk, T. Kol, Coding on demand by an informed source (iscod) for efficient broadcast of different supplemental data to caching clients, IEEE/ACM Trans. Networking 14 (6) (2006) 2825–2830.

[5] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, in: Proc. Annual ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom), 1998.

[6] J. Chen, V. Lee, K. Liu, G. M. N. Ali, E. Chan, Efficient processing of requests with network coding in on-demand data broadcast environments, Information Sciences 232 (2013) 27–43.

[7] J. Chen, V. C. Lee, K. Liu, On the performance of real-time multi-item request scheduling in data broadcast environments, Journal of Systems and Software 83 (8) (2010) 1337–1345.

[8] C.-Y. Chow, H. V. Leong, A. Chan, Cache signatures for peer-to-peer cooperative caching in mobile environments, in: Proc. Int'l Conf. Advanced Information Networking and Applications (AINA), 2004.

[9] C.-Y. Chow, H. V. Leong, A. T. Chan, Group-based cooperative cache management for mobile clients in a mobile environment, in: Proc. Int'l Conf. Parallel Processing (ICPP), 2004.

[10] C.-Y. Chow, H. V. Leong, A. T. Chan, Grococa: Group-based peer-to-peer cooperative caching in mobile environment, IEEE J. Selected Areas in Comm. 25 (1) (2007) 179–191.

[11] H. Dykeman, M. H. Ammar, J. Wong, Scheduling algorithms for videotex systems under broadcast delivery., in: Proc. IEEE Int'l Conf. Communications (ICC), 1986.

[12] L. M. Feeney, M. Nilsson, Investigating the energy consumption of a wireless network interface in an ad hoc networking environment, in: Proc. IEEE INFOCOM, 2001.

[13] W. Gao, G. Cao, A. Iyengar, M. Srivatsa, Cooperative caching for efficient data access in disruption tolerant networks, IEEE Trans. Mobile Computing 13 (3) (2014) 611–625.

[14] X. Hong, M. Gerla, G. Pei, C.-C. Chiang, A group mobility model for ad hoc wireless networks, in: Proc. ACM Int'l Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 1999.

[15] H. Ji, V. C. Lee, C.-Y. Chow, K. Liu, G. Wu, Coding-based cooperative caching in data broadcast environments, in: Proc. Int'l Conf. Algorithms and Architectures for Parallel, Springer, Berlin, Germany, 2015, pp. 273–287.

[16] N. Kumar, J.-H. Lee, Peer-to-peer cooperative caching for data dissemination in urban vehicular communications, IEEE Systems Journal 8 (4) (2014) 1136–1144.

[17] C.-M. Liu, T.-C. Su, Broadcasting on-demand data with time constraints using multiple channels in wireless broadcast environments, Information Sciences 242 (2013) 76–91.

[18] K. Liu, V. Lee, On-demand broadcast for multiple-item requests in a multiple-channel environment, Information Sciences 180 (22) (2010) 4336–4352.

[19] K. Liu, V. C. Lee, Simulation studies on scheduling requests for multiple data items in on-demand broadcast environments, Performance Evaluation 66 (7) (2009) 368–379.

[20] K. Liu, J. K.-Y. Ng, J. Wang, V. C. Lee, W. Wu, S. H. Son, Network-coding-assisted data dissemination via cooperative vehicle-to-vehicle/-infrastructure communications, IEEE Transactions on Intelligent Transportation Systems 17 (6) (2016) 1509–1520.

[21] D. S. Lun, M. Médard, R. Koetter, Efficient operation of wireless packet networks using network coding, in: Proc. of International Workshop on Convergent Technologies (IWCT), 2005.

[22] G. Md Nawaz Ali, V. C. Lee, E. Chan, M. Li, K. Liu, J. Lv, J. Chen, Admission control-based multichannel data broadcasting for real-time multi-item queries, IEEE Trans. Broadcasting 60 (4) (2014) 589–605.

[23] A. Mohapatra, N. Gautam, S. Shakkottai, A. Sprintson, Network coding decisions for wireless transmissions with delay consideration, IEEE Trans. Communications 62 (8) (2014) 2965–2976.

[24] M. Mohseni, D. Zhao, Time and power scheduling in an ad hoc network with bidirectional relaying and network coding, Wireless Communications and Mobile Computing 15 (2013) 459–474.

[25] H.-T. Roh, J.-W. Lee, Network coding-aware flow control in wireless ad-hoc networks with multi-path routing, Wireless networks 19 (5) (2013) 785–797.

[26] F. Sailhan, V. Issarny, Cooperative caching in ad hoc networks, in: Proc. Int'l Conf. Mobile Data Management (MDM), 2003.

[27] H. Schwetman, Csim19: A powerful tool for building system models, in: Proc. Conf. Winter Simulation (WSC), 2001.

[28] I.-W. Ting, Y.-K. Chang, Improved group-based cooperative caching scheme for mobile ad hoc networks, Journal of Parallel and Distributed Computing 73 (5) (2013) 595–607.

[29] B. Wang, Y. Zhang, X. Zhou, S. Ci, Y. Qi, C-lgv: A novel cooperative caching scheme for p2p caches, in: Proc. IEEE Int'l Conf. Communications (ICC), 2013.

[30] J. Wang, J. Ren, K. Lu, J. Wang, S. Liu, C. Westphal, An optimal cache management framework for information-centric networks with network coding, in: Proc. IFIP Netw. Conf., 2014.

[31] Y. Wu, P. A. Chou, S.-Y. Kung, Minimum-energy multicast in mobile ad hoc networks using network coding, IEEE Trans. Communications 53 (11) (2005) 1906–1918.

[32] L. Yin, G. Cao, Supporting cooperative caching in ad hoc networks, IEEE Trans. Mobile Computing 5 (1) (2006) 77–89.

[33] C. Zhan, V. C. Lee, J. Wang, Y. Xu, Coding-based data broadcast scheduling in on-demand broadcast, IEEE Trans. Wireless Communications 10 (11) (2011) 3774–3783.

[34] P. Zhang, C. Lin, Y. Jiang, Y. Fan, X. Shen, A lightweight encryption scheme for network-coded mobile ad hoc networks, IEEE Trans. Parallel and Distributed Systems 25 (9) (2013) 2211–2221.

[35] Y. Zhang, X. Xu, X. Wang, Z. Fang, J. Tang, Nc-coca: Network coding-based cooperative caching scheme, in: Proc. IEEE Int'l Conf. Computational Science and Engineering (CSE), 2014.

[36] Y. Zhang, X. Zhou, Y. Liu, B. Wang, S. Ci, A novel cooperative caching algorithm for massive p2p caches, Peer-to-Peer Networking and Applications 6 (4) (2013) 425–433.