Coding-based Cooperative Caching in Data Broadcast Environments

Houling Ji[1,2], Victor C.S. Lee[2], Chi-Yin Chow[2], Kai Liu[3], Guoqing Wu[1]

[1] School of Computer, Wuhan University, Wuhan, China
jihouling@whu.edu.cn, wgq@whu.edu.cn
[2] Depart. of Comp. Science, City Uni. of Hong Kong, Hong Kong
csvlee@cityu.edu.hk, chiychow@cityu.edu.hk
[3] College of Computer Science, Chongqing University, Chongqing, China
liukai0807@cqu.edu.cn

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 the 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, MHs which are not neighbors cannot cooperate even if they have each other's requested data item in their own caches. Network coding is a technique, by exploiting which multiple MHs can retrieve different requested data items from an encoded packet which encodes a number of data items broadcast by the MSS in a broadcast time unit. In this research work, we propose to apply network coding to enabling MHs which are not neighbors to cooperate indirectly. We devise two algorithms running at the MSS and MHs, respectively, for making encoding decisions and decoding requested data from encoded packets. We build the simulation model for performance evaluation and the simulation results demonstrate that the proposed solution not only increases the bandwidth efficiency of the limited downlink communication channel from the MSS to MHs but also enhances the system performance by reducing the latency in satisfying requests.

1 Introduction

Data broadcast is an effective way to disseminate data in an infrastructure network due to its scalability and flexibility. A typical infrastructure network consists of mobile hosts (MHs) and a mobile support station (MSS). The MSS provides information access to MHs within a certain service area. Even when the number of MHs increases significantly, the data broadcast system can achieve decent performance. On-demand broadcast is one of the most promising data broadcast techniques [1]. In on-demand broadcast environments, MHs submit requests for data items 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 algorithm, such as FCFS (First Come First Served) [2], MRF (Most Requested First) [3] or RxW (Number of pending Requests Multiply Waiting time) [4], etc. Requesting MHs tune in to the downlink channel waiting for their requested data items. However, in this typical on-demand

broadcast environment, the bandwidth efficiency of the downlink communication channel cannot be fully exploited because only those MHs requesting the same data item can be satisfied in one broadcast time unit.

Network coding [5], originally proposed in information theory, is a technique in which intermediate nodes can combine and forward the received data packets from multiple links [6]. It has attracted researchers' attention due to its potential for enhancing the system performance of both mobile ad hoc networks [7,8,9,10] and infrastructure networks [11,12,13,14]. Zhan *et al.* [14] proposed a generalized encoding framework to incorporate network coding into data scheduling algorithms for on-demand broadcast. Chen *et al.* [13] proposed two novel coding assisted algorithms called ADC-1 and ADC-2 which consider data scheduling, in addition to network coding. In on-demand broadcast environments, with network coding, it is possible for multiple MHs requesting different data items to be satisfied simultaneously (i.e. in one broadcast time unit) by utilizing their cached data items.

With the recent development of peer-to-peer (P2P) wireless communication technologies, a new information sharing paradigm appears. Mobile clients can not only retrieve information from the server, but also share information among peers. The peers of an MH refer to those MHs which reside in the transmission range of the MH. This kind of information sharing among peers is called cooperative caching [15]. Several cooperative caching schemes were proposed in mobile environments in the recent decade [16,17]. Ting and Chang [18] 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 [15,19,20], 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 peers. 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. Note that peers are not necessarily the members of the same TCG. However, in any group-based cooperative caching schemes, it is difficult for MHs who are not neighbors to collaborate, even though they have in their caches the data items requested by others. Since network coding has the potential to exploit cached data items to serve different MHs, whether they are peers not, it is possible for network coding to bridge the gap between MHs residing out of each other's communication range and further improve the system performance of cooperative caching.

The main contributions of this paper are outlined as follows. First, we exploit the synergy between network coding and cooperative caching. On the one hand, network coding strengthens the information sharing paradigm of cooperative caching by enabling MHs residing in different groups to cooperate. Similar to the situation of cooperative caching, in case they have each other's requested data item 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. On the other hand, cooperative caching facilitates the operation of network coding. Consider a request submitted by an MH to the MSS. MHs 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 the MHs in the same group of the requesting MH in making the encoding decisions. Second, 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 and propose two algorithms. One is running at the MSS side to encode a number of requested data items in a single packet for broadcasting. The other is running at the MH side for multiple MHs to decode the packet with their cached data items and retrieve different data items to satisfy their requests in a single broadcast time unit. Last, 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, further increases the bandwidth efficiency of the downlink communication channel, and significantly improves the system performance in terms of reducing data access latency.

The rest of this paper is organized as follows. A novel coding-based cooperative caching system is introduced and two algorithms are proposed in Section 2. Section 3 builds the simulation model. Section 4 conducts the performance evaluation. Finally, we conclude this work in Section 5.

2 Coding-based Cooperative Caching

In this section, we introduce a novel cooperative caching system called Coding-based Cooperative Caching, and then propose two cooperative caching algorithms. One algorithm is running at the MSS to encode and broadcast data packets to MHs, and the other algorithm is running at MHs to receive and decode data packets broadcast by the MSS.

2.1 Architecture

The system architecture shown in Fig. 1 consists of one MSS and a number of MHs. Suppose there are $NumMH$ MHs $m_1, m_2, ..., m_{NumMH}$ and each of them has a unique identifier. 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 neighbouring MHs. In the system, both the MSS and MHs have certain communication area determined by their respective transmission range. One MH can communicate with others residing in its own communication area. 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

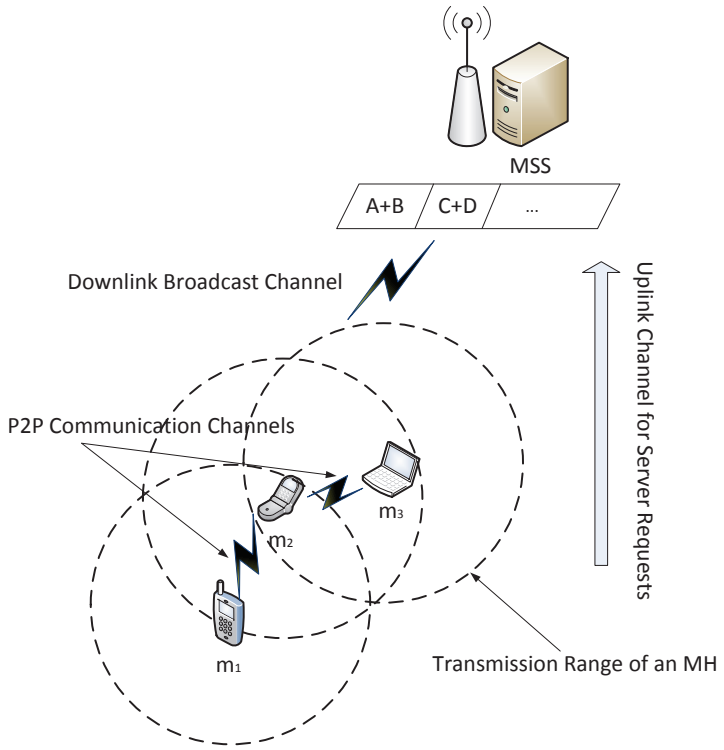


Fig. 1. System Architecture

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. IEEE 802.11 and CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) are adopted in the wireless communication channels to transmit data and avoid collisions.

Communication Protocol: If an 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. 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, which informs 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. After 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. The MSS maintains a request queue for outstanding requests submitted by MHs. According to a certain scheduling algorithm, the MSS fetches data items from the database and makes the decision whether to encode them or not based on the information of both cached items and required items of each MH. In each broadcast time unit, the MSS broadcasts either an encoded or uncoded data item via the

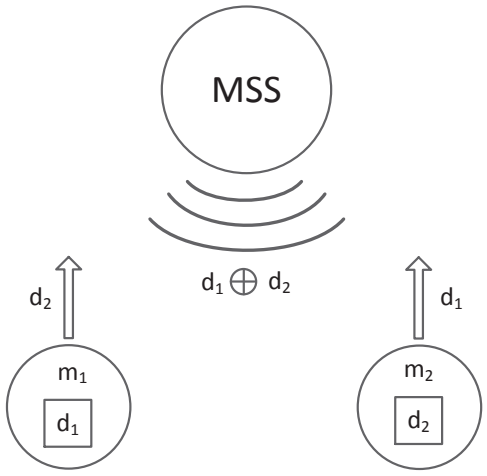


Fig. 2. A Network Coding Example

downlink channel to serve outstanding requests. The objective is to utilize the limited broadcast bandwidth efficiently.

The timeout period is initially set to the round-trip time of P2P communication channel such that timeout is computed by: $\frac{request\ size\ +\ reply\ size}{BW_{P2P}} \times \varphi$ initially, 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 weight the standard deviation in computing the timeout period.

Therefore, there are three possible situations that an MH may encounter when requesting a data item. (1) Local Cache Hit: 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: If the MH encounters a local cache miss, it attempts to enlist its neighbouring peers for help. If any peer caches and replies the required data item, it is a global cache hit. Otherwise, it is a cache miss. (3) Server Request: If the MH encounters a global cache miss, it will send a server request to the MSS via the uplink channel. Then, the MH will tune in to the downlink broadcast channel to retrieve the required data item.

2.2 Proposed Algorithms

Network coding has the potential to further improve the bandwidth efficiency of the downlink channel. We verify this claim with the following example. Assume there is one MSS and two MHs m_1 and m_2 in the system as shown in Fig. 2. m_1 and m_2 are not peers. Suppose the MSS has a database with two data items d_1 and d_2. m_1 caches d_1 and requires d_2, while m_2 caches d_2 and requires d_1. Since both m_1 and m_2 cannot find their required data items in their own caches or from their peers, they need to submit requests for their

required data items to the MSS. In the conventional broadcast strategy, the MSS needs to broadcast twice (d_1, d_2 in sequence) to satisfy both requests. In contrast, with network coding, the MSS only needs to broadcast one encoded data packet $d_1 \oplus d_2$ (bit-by-bit XOR operation) to satisfy both requests. When m_1 receives $d_1 \oplus d_2$, it can get d_2 by $d_1 \oplus (d_1 \oplus d_2)$. m_2 can get d_1 in a similar way.

Algorithm 1 MSS Scheduling and Encoding Algorithm at Each Broadcast Time Unit

1: Fetch the request, i.e., $R(m_i, d_j)$, at the head of the request queue Q, where m_i is the MH sending the request and d_j is the data item m_i requests;
2: $MaxNum \leftarrow 0$;
3: $ClientsToSatisfy \leftarrow \phi$;
4: $DataToBroadcast \leftarrow d_j$;
5: **for** each $R(m_k, d_l) \in Q$ **do**
6: **if** $d_l = d_j$ **then**
7: $MaxNum \leftarrow MaxNum + 1$;
8: $ClientsToSatisfy \leftarrow ClientsToSatisfy \cup \{m_k\}$;
9: **end if**
10: **end for**
11: Suppose the cache content of m_i is C_i in the MSS view;
12: **for** each $d_p \in C_i$ **do**
13: $ClientsToSatisfyEncode \leftarrow \phi$;
14: $MaxNumEncode \leftarrow 0$;
15: **for** each $R(m_r, d_t) \in Q$ **do**
16: Suppose the cache content of m_r is C_r in the MSS view;
17: **if** $d_j = d_t$ **and** $d_p \in C_r$ **then**
18: $MaxNumEncode \leftarrow MaxNumEncode + 1$;
19: $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
20: **else if** $d_p = d_t$ **and** $d_j \in C_r$ **then**
21: $MaxNumEncode \leftarrow MaxNumEncode + 1$;
22: $ClientsToSatisfyEncode \leftarrow ClientsToSatisfyEncode \cup \{m_r\}$;
23: **end if**
24: **end for**
25: **if** $MaxNumEncode > MaxNum$ **then**
26: $MaxNum \leftarrow MaxNumEncode$;
27: $ClientsToSatisfy \leftarrow ClientsToSatisfyEncode$;
28: $DataToBroadcast \leftarrow d_j \oplus d_p$;
29: **end if**
30: **end for**
31: Broadcast the data packet in $DataToBroadcast$ and prefix the MH identifiers in $ClientsToSatisfy$;

First, we propose an algorithm running at the MSS (the pseudocode is shown in Algorithm 1). This algorithm is used for the MSS to schedule, encode and finally broadcast requested data items to MHs. When a request arrives at the MSS, it is inserted into the tail of the request queue. In order to illustrate the impact of network coding on the system performance, we adopt simple first-come-first-served (FCFS) scheduling algorithm at the server and encode only two data items in a packet. The MSS seeks to maximize the number of requests to be satisfied in a broadcast unit, on the basis of satisfying the request pending at the head of the request queue first. Note that a

Algorithm 2 A Requesting MH m_i's Action When Receiving Data Packet $DataToBroadcast$ Broadcast by the MSS

Suppose the request of m_i is $R(m_i, d_j)$, pending in the request queue Q. The cache content of m_i is C_i. MHs intended to be served by the current broadcast is $ClientsToSatisfy$;
if $DataToBroadcast$ is an unencoded packet, i.e., d_k **then**
  if $d_j = d_k$ **then**
    Obtain the requested data item d_j and and notify the MSS to remove $R(m_i, d_j)$;
  else
    Ignore d_k and wait for the MSS's next service;
  end if
else if $DataToBroadcast$ is an encoded packet, i.e., $d_k \oplus d_p$ **then**
  if $d_j = d_k$ **and** $d_p \in C_i$ **then**
    Obtain the requested data item d_j by $d_p \oplus (d_k \oplus d_p)$ and notify the MSS to remove $R(m_i, d_j)$;
  else if $d_j = d_p$ **and** $d_k \in C_i$ **then**
    Obtain the requested data item d_j by $d_k \oplus (d_k \oplus d_p)$ and notify the MSS to remove $R(m_i, d_j)$;
  else if $m_i \in ClientsToSatisfy$ **then**
    Notify the MSS to update its view and wait for the MSS's next service;
  else
    Ignore $d_k \oplus d_p$ and wait for the MSS's next service;
  end if
end if

request can be assumed to be satisfied in the MSS's view if its requested data item is broadcast directly or one data item in the broadcast data packet is requested and the other data item exists in the requesting client's cache. A requested data item will be encoded with another requested data item only if the encoded packet can serve more requests in comparison to the case that the requested data item is unencoded.

Then, we propose an algorithm running at MHs (the pseudocode is shown in Algorithm 2). This algorithm is used for MHs to receive and decode data packets broadcast by the MSS. The MSS maintains a view of the cache content of all MHs in the system by tracing the requests submitted by MHs and data items broadcast to serve the requests. Since the cache size of MHs is limited and MHs may cache data items obtained from its peers, the data items in an MH's cache may change from time to time, leading to inconsistency between the MSS view and the real 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 cache view. When the MSS broadcasts an encoded packet, it prefixes a list of MH identifiers of intended recipients who should be able to decode the packet and retrieve the requested data items. If an MH in the list can decode the packet and obtain the required data item, it notifies the MSS to remove its request pending in the request queue. Otherwise, if the MH cannot decode the packet, which implies that the MSS view is inconsistent with the real cache content of MH, the MH will send the identifier of data items in its cache to the MSS to update the MSS view. The request of the MH remains in the request queue waiting for the MSS's next service.

3 Simulation Model

We construct a simulation model as described in Section 3 in C++ using CSIM [21]. In our 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 channel with a bandwidth of BW_{uplink}. The P2P communication channel is used for an MH to communicate with its peers with the bandwidth of BW_{P2P}.

3.1 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* [22], 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* [23]. The logical group center randomly 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 *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 time period between two consecutive requests by an MH follows an exponential distribution with a mean of one second. Note that in this paper, the local processing time at MHs is considered negligible, because the cost of decoding using bit-by-bit XOR operation is low [13].

3.2 Server Model

There is one MSS in the system, which serves the MHs in the area of 3000 $m \times 3000$ 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. The key parameters and their default values in the simulation are shown in Table 1.

Table 1. Default parameter settings

Parameters	Values
$NumMH$	300
$GroupSize$	3
$NumData$	1000
$TranRange$	100 m
$CacheSize$	100 data items
θ	0.6
$DataSize$	32 kb
$ReqSize, RepSize, RetSize$	512 b
$BW_{downlink}$	10 Mbps
BW_{uplink}	1 Mbps
BW_{P2P}	5 Mbps
v_{min}	1 m/s
v_{max}	5 m/s
φ, φ'	10, 3

4 Performance Evaluation

4.1 Performance Metrics

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

Server Request Ratio Server Request Ratio is the percentage of the number of service requests to the total number of requests:

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

Server Broadcast Productivity Server Broadcast Productivity is the percentage of the number of server requests satisfied to the total number of packets broadcast by the MSS:

$$\frac{Total\ Number\ of\ Server\ Requests\ Satisfied}{Total\ Number\ of\ Packets\ Broadcast\ by\ the\ MSS}$$

It describes the average number of requests satisfied by one data packet.

Encoded Packet Broadcast Ratio Encoded Packet Broadcast Ratio is the percentage of the number of encoded packets broadcast by the MSS to the 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.

Mean Request Queue Length It describes the mean number of requests pending in the request queue.

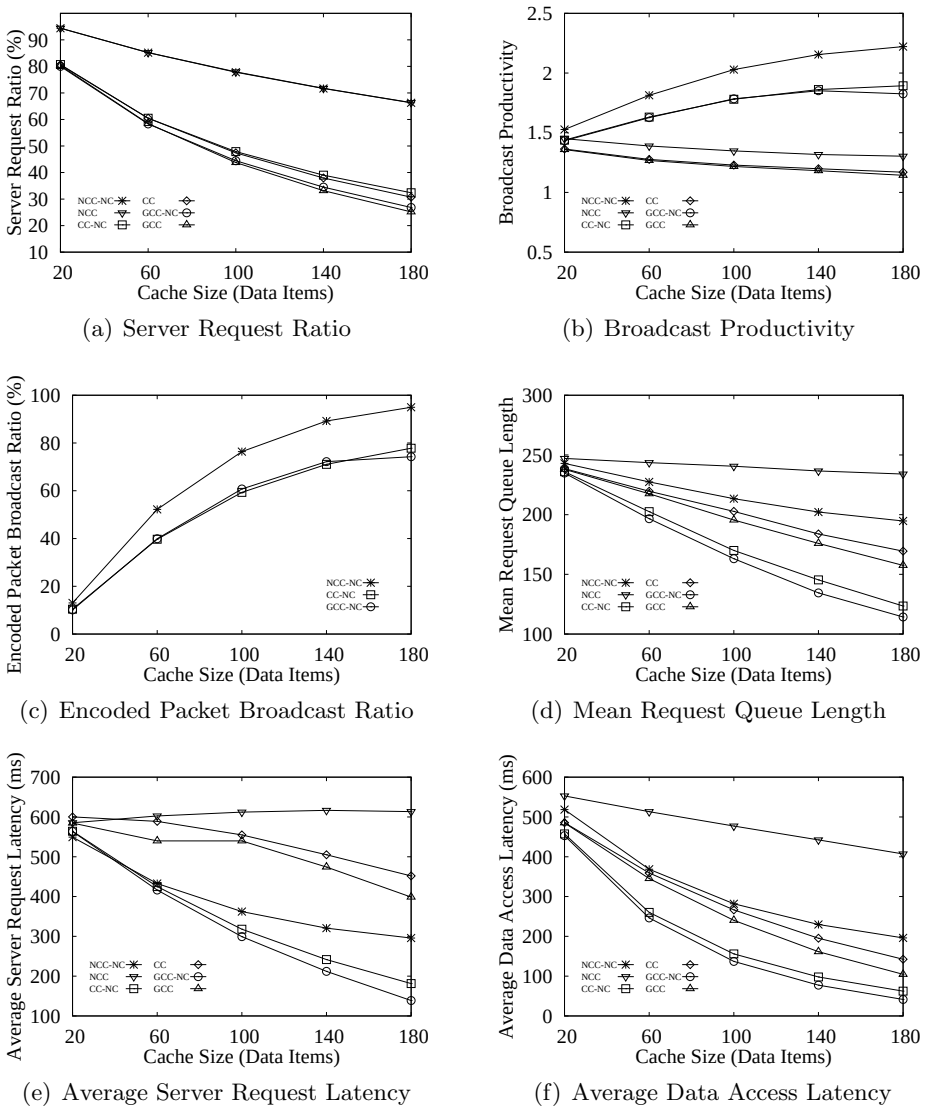
Average Server Request Latency It describes the waiting time from the moment a client sends a server request to the MSS to the moment its request is finally satisfied.

Average Data Access Latency It describes the average time period from the moment an MH wants a data item to the moment the desired data item is obtained.

4.2 Simulation Results

In this section, we present the performance results of the coding-based cooperative caching. We compare the performance of Coding-based GroCoca (denoted as GCC-NC) with non-cooperative caching scheme (denoted as NCC), non-cooperative caching scheme with network coding (denoted as NCC-NC), COCA scheme (denoted as CC), COCA scheme with network coding (denoted as CC-NC), and GroCoca (denoted as GCC). The results are collected obtained 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.

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. Fig. 3(a) shows that the server request ratio decreases. This is because the local cache hit and global cache hit ratio both increase. The cache hit ratios increase because a larger cache size leads to a higher chance for MHs to obtain data items locally or from peers. As shown in Fig. 3(b), the schemes with network coding have better performance in broadcast productivity than their non-coding versions. In Fig. 3(c), the encoded packet broadcast ratio increases in NCC-NC and CC-NC because larger cache size leads to higher chance for network coding. As depicted in Fig. 3(d), 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 Fig. 3(e). In Fig. 3(f), 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 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 latency.



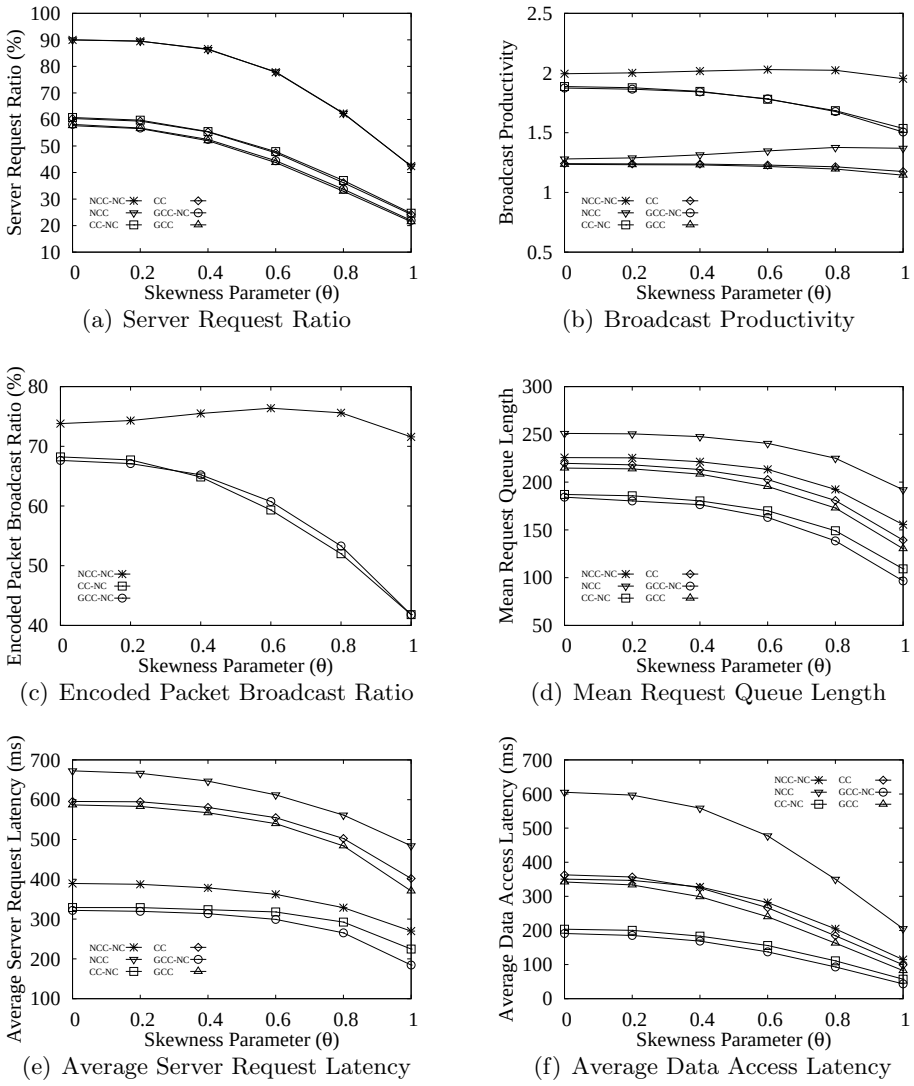
(a) Server Request Ratio (b) Broadcast Productivity

(c) Encoded Packet Broadcast Ratio (d) Mean Request Queue Length

(e) Average Server Request Latency (f) Average Data Access Latency

Fig. 3. Effect of cache size on system performance

Effect of Access Pattern Next, we study the effect of access pattern on system performance by increasing the skewness parameter θ from 0 to 1. As θ gets larger, MHs have more common interest and 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 the decrease of the server request ratio as shown in Fig. 4(a). The schemes with network coding have better performance in broadcast productivity than their non-coding versions as shown in Fig. 4(b). Fig. 4(c) describes the encoded packet broadcast ratio decreases as θ gets larger. Skewed data access patterns degrades the efficiency of network coding. As depicted in Fig. 4(d), the schemes with network coding have shorter request queue length than their non-coding versions. So the schemes with network coding have less average



(a) Server Request Ratio (b) Broadcast Productivity

(c) Encoded Packet Broadcast Ratio (d) Mean Request Queue Length

(e) Average Server Request Latency (f) Average Data Access Latency

Fig. 4. Effect of access pattern on system performance

server request latency as depicted in Fig. 4(e). In Fig. 4(f), the average data access latency decreases, 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 latency. GCC-NC has the best performance in terms of average data access latency among all the schemes.

5 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 neighbors can collaborate indirectly by utilizing their own cached data items. In particular, we have outlined the communication protocol of the system model and proposed two algorithms running at the MSS side and MH side. The MSS encodes multiple data items to satisfy as many MHs as possible in a single broadcast time unit, while MHs retrieve different requested data items from a single encoded packet simultaneously. Finally, the experimental results show that coding-based non-cooperative caching scheme (NCC-NC), coding-based COCA scheme (CC-NC) and coding-based GroCoca scheme (GCC-NC) all outperform their respective non-coding counterparts in terms of enhancing the bandwidth efficiency of the downlink communication channel and reducing the data access latency. Above all, coding-based GroCoca has the best performance among all the schemes, which confirms the synergy between network coding and group-based cooperative caching.

Acknowledgement

The work described in this paper was substantially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 115312] and the Fundamental Research Funds for the Central Universities (Grant No.106112015CDJZR185518).

References

1. Liu, K., Lee, V.: On-demand broadcast for multiple-item requests in a multiple-channel environment. Information Sciences **180**(22) (2010) 4336–4352
2. Dykeman, H., Ammar, M.H., Wong, J.: Scheduling algorithms for videotex systems under broadcast delivery. In: Proc. IEEE Int'l Conf. Communications (ICC). (1986)
3. Dykeman, H., Wong, J.: A performance study of broadcast information delivery systems. In: Proc. IEEE INFOCOM. (1988)
4. Aksoy, D., Franklin, M.: R× w: a scheduling approach for large-scale on-demand data broadcast. IEEE/ACM Trans. Networking **7**(6) (1999) 846–860
5. Ahlswede, R., Cai, N., Li, S.Y., Yeung, R.W.: Network information flow. IEEE Trans. Information Theory **46**(4) (2000) 1204–1216
6. Lun, D.S., Médard, M., Koetter, R.: Efficient operation of wireless packet networks using network coding. In: Proc. of International Workshop on Convergent Technologies (IWCT). (2005)
7. Mohseni, M., Zhao, D.: Time and power scheduling in an ad hoc network with bidirectional relaying and network coding. Wireless Communications and Mobile Computing **15** (2013) 459–474
8. Roh, H.T., Lee, J.W.: Network coding-aware flow control in wireless ad-hoc networks with multi-path routing. Wireless networks **19**(5) (2013) 785–797
9. Wu, Y., Chou, P.A., Kung, S.Y.: Minimum-energy multicast in mobile ad hoc networks using network coding. IEEE Trans. Communications **53**(11) (2005) 1906–1918
10. Zhang, P., Lin, C., Jiang, Y., Fan, Y., Shen, X.: A lightweight encryption scheme for network-coded mobile ad hoc networks. IEEE Trans. Parallel and Distributed Systems **25**(9) (2013) 2211–2221
11. Birk, Y., Kol, T.: 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
12. Ali, G., Meng, Y., Lee, V., Liu, K., Chan, E.: 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)

13. Chen, J., Lee, V., Liu, K., Ali, G.M.N., Chan, E.: Efficient processing of requests with network coding in on-demand data broadcast environments. Information Sciences **232** (2013) 27–43
14. Zhan, C., Lee, V.C., Wang, J., Xu, Y.: Coding-based data broadcast scheduling in on-demand broadcast. IEEE Trans. Wireless Communications **10**(11) (2011) 3774–3783
15. Chow, C.Y., Leong, H.V., Chan, A.T.: Grococa: Group-based peer-to-peer cooperative caching in mobile environment. IEEE J. Selected Areas in Comm. **25**(1) (2007) 179–191
16. Sailhan, F., Issarny, V.: Cooperative caching in ad hoc networks. In: Proc. Int'l Conf. Mobile Data Management (MDM). (2003)
17. Yin, L., Cao, G.: Supporting cooperative caching in ad hoc networks. IEEE Trans. Mobile Computing **5**(1) (2006) 77–89
18. Ting, I.W., Chang, Y.K.: Improved group-based cooperative caching scheme for mobile ad hoc networks. Journal of Parallel and Distributed Computing **73**(5) (2013) 595–607
19. Chow, C.Y., Leong, H.V., Chan, A.: Cache signatures for peer-to-peer cooperative caching in mobile environments. In: Proc. Int'l Conf. Advanced Information Networking and Applications (AINA). (2004)
20. Chow, C.Y., Leong, H.V., Chan, A.T.: Group-based cooperative cache management for mobile clients in a mobile environment. In: Proc. Int'l Conf. Parallel Processing (ICPP). (2004)
21. Schwetman, H.: Csim19: A powerful tool for building system models. In: Proc. Conf. Winter Simulation (WSC). (2001)
22. Hong, X., Gerla, M., Pei, G., Chiang, C.C.: 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)
23. Broch, J., Maltz, D.A., Johnson, D.B., Hu, Y.C., Jetcheva, J.: 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)