Hybrid Cooperative Caching in a Mobile Environment

Hong Va Leong[1], Chi-yin Chow[2]

[1] Department of Computing, The Hong Kong Polytechnic University
Hong Kong
cshleong@comp.polyu.edu.hk
[2] Department of Computing Science, University of Minnesota
Minneapolis, MN 55455
cchow@cs.umn.edu

Abstract. Caching is a key technique for improving data access performance of mobile clients. The emergence of peer-to-peer communication technologies now brings to reality "cooperative caching" in which mobile clients help one another in caching. They not only can retrieve data items from mobile support stations, but also from the cache in their peers. Traditional caching is based on an on-demand paradigm, where clients pull data to be cached. In this work, we also consider the use of the inherent broadcast mechanism from the mobile support station to complement on-demand pull-based access from the station and from peer mobile clients, to form the hybrid environment. The performance is evaluated for cooperative caching in these environments. Pull-based environments generally lead to lower access latency, while push-based and hybrid environments could effectively reduce power consumption.

Keywords: Peer-to-peer computing, hybrid mobile environment, mobile ad hoc network, cooperative caching.

1 Introduction

The recent widespread deployment of new peer-to-peer wireless communication technologies like Bluetooth, coupled with the ever-increasing processing power and storage capacity of mobile devices, have led to the peer-to-peer information sharing paradigm to take shape. Mobile clients can communicate among themselves to share information. There are two major types of architecture for a mobile system. An infrastructure-based mobile system is formed with a wireless network connecting mobile hosts (MHs) and mobile support stations (MSSs). The MHs can retrieve their desired data items from the MSS, either by requesting them over shared point-to-point channels (pull-based), catching them from scalable broadcast channels (push-based), or using a combination of both types of channels (hybrid). In an ad hoc mobile system (mobile ad hoc network or MANET), the MHs can share information without any MSS. This is referred to as the *peer-to-peer* model.

In an infrastructure-based system, the bandwidth of uplink channels is much lower than that of downlink channels, potentially becoming a scalability bottleneck [2]. Push-based and hybrid data models are scalable at the expense of longer access latency. MANET is practical when there is no fixed infrastructure support but not suitable for commercial mobile applications. In MANETs, the MHs can rove freely and disconnect from the network frequently, leading to dynamic network topology changes. Long access latency or access failure can occur, when the peers holding desired data are far way or unreachable.

The inherent shortcomings of these motivate us to develop a novel scheme for deploying mobile data access applications. In this paper, we extend the COoperative CAching scheme (COCA) that operates on conventional infrastructure-based mobile system with peer-to-peer communication technology in a pull-based environment to push-based and hybrid environments. COCA is appropriate for an environment in which a group of MHs possesses a common access interest. For instance, in a conference and a shopping mall, the MHs sitting in the same session room or wandering around in the same shop are probably interested in information related to the research topic of the paper or information pertaining to the shop respectively. When they share a common access interest, there is a higher probability for them to find the required data from the cache of their peers.

Cooperative caching is relevant to research in cooperative data retrieval and cache management. The former mainly focuses on how to search for data items and forwarding them from the source MH or the MSS to the requesting MH, while the latter focuses on how MHs can cooperatively manage their cache as a global cache or aggregate cache to improve system performance.

In [13], an intuitive cooperative caching architecture for a MANET environment is proposed. To retrieve a piece of data, an MH would obtain it from an MSS if it can directly connect to the MSS; otherwise, it has to enlist its peers at a distance less than the MSS for help to turn in the required data. If no such peer caches the data, the peers route the request to the nearest MSS. In [12], the 7DS (Seven Degrees of Separation) cooperative caching scheme is used as a complement to the infrastructure support with power conservation. When an MH fails to connect to Internet to retrieve the desired data item, it would attempt to search for it from its neighboring 7DS peers.

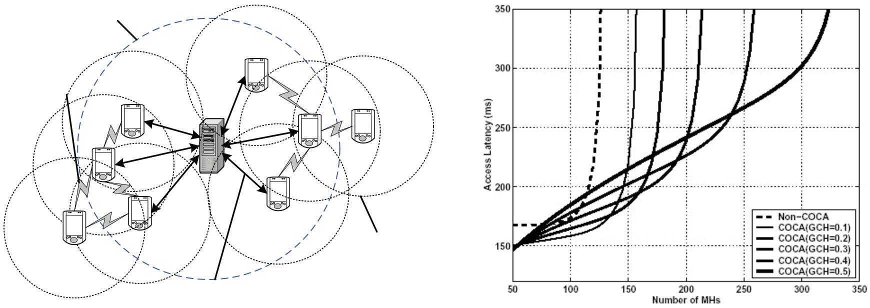
To manage a cooperative cache, one needs to consider initial data replica allocation to cache, cache admission control and cache replacement. Hara [6] proposes three replica allocation schemes, by considering the data item access probability of MHs, their connected neighborhoods and then grouping together MHs with high connection stability. These schemes are then adapted to a broadcast environment, taking into consideration of periodic data update [7]. Huang et al. [9] propose another distributed data replica allocation scheme in MANETs to improve data accessibility and reduce network traffic due to the replication mechanism, assuming that some MHs tend to roam together and share a common access range. Lim et al. [11] propose a cooperative caching scheme for Internet-based MANETs based on a simple, flooding-based searching scheme. For cache admission control, an MH determines whether to cache a data item based on the distance between itself and the data source that can be either other peers caching the item or the MSS. For cache replacement, a victim item is selected to be evicted from the cache by an MH based on the distance between itself and other peers caching the victim or the MSS.

The rest of this paper is organized as follows. The COCA model and the mobile environments are presented in Section 2. In Section 3, the simulation model of COCA is defined. We study and evaluate the system performance in Section 4. Finally, Section 5 offers brief concluding remarks to this paper.

2 COCA Model and Environments

2.1 COCA System Model

In COCA, we assume that each MH is equipped with two wireless network interface cards, one dedicated to communicate with MSS and another with other MHs. With point-to-point communication, there is only one destination MH for the message sent from the source MH, whereas with broadcast, all MHs residing in the transmission range of the source receive the message. COCA is based on the system architecture in Fig. 1*a*, in which each MH and its peers share their cached data items cooperatively.



(*a*) COCA architecture (*b*) Analysis of pull-based environment

Fig. 1. COCA.

In COCA, we distinguish cache hits and cache misses. If the required data item is found in the MH's local cache, it is a local cache hit (LCH). When the required item is not cached, the MH attempts to retrieve it from its peers. If some peer can turn in the item, it is a global cache hit (GCH). If the MH fails to achieve neither a LCH nor a GCH, it encounters a cache miss and has to contact the MSS to obtain the required item. If it still fails to access the item from the MSS, as it is residing outside of the service area or the MSS is overloaded, that is an access failure. We deploy three types of messages in COCA: *request*, *reply* and *retrieve*. A *request* contains a unique identifier formed from the user identifier and request timestamp. Each peer processes each request only once and drops all duplicates. If an MH caches the required item, it sends a *reply* to the requesting MH directly or through multi-hop routing. Otherwise, the MH decrements the hop count and propagates the request. A request with a zero hop count will be discarded. After the requesting MH receives a *reply* from its peer, it sends a *retrieve* to the peer for the required item. In multi-hop data searching, intermediate MHs propagate requests and items between requesting and source MHs.

2.2 Pull-based Environment

In a pull-based system, the storage hierarchy is generally composed of three layers: Mobile Client Cache, MSS Cache and MSS Disk. When an MH encounters a local cache miss, it sends a request to the MSS. The MSS processes the request and sends the item back to the requesting MH. In COCA, a new logical layer is inserted between the Mobile Client Cache and MSS Cache layer, called Peer Cache layer [4]. When an MH encounters a local cache miss, it first attempts to search for the desired items in the Peer Cache layer. If no peer caches the item, it obtains the item from the MSS.

In a pull-based mobile environment, COCA requires that an MH should first find the required item in its local cache. If it encounters a local cache miss, it broadcasts a *request* to its peers within the distance of a predefined number of hops via peer-to-peer broadcast communication. Any peer caching the required item will return a *reply* to the requesting MH. When the MH receives a *reply*, it sends a *retrieve* to the peer. In case of no peer sending *reply* back upon timeout, the requesting MH has to obtain the item from the MSS. The timeout period is defined to be adaptive to the degree of network congestion. We have done some analytical study on the expected access latency of the MHs with number of MHs, as shown in Fig. 1*b*. The result shows that COCA is much more scalable than the conventional pull-based system and the system capability increases as GCH ratio increases.

2.3 Push-based Environment

In a push-based system, the storage hierarchy commonly consists of three layers: Mobile Client Cache, Broadcast Channel and MSS Disk. The MSS grabs the data items from the disk and allocates them to the broadcast channel. If an MH encounters a local cache miss, it tunes in to the broadcast channel and catches the required item when the item appears in the channel. In COCA, we insert the Peer Cache layer as a supplementary component of the Broadcast Channel layer. When an MH suffers from a local cache miss, the MH tunes in to the broadcast channel; meanwhile, it searches for the required item in the Peer Cache layer. There is a GCH, if some peers turn in the item to the MH before either the item appears on the broadcast channel or the timeout period elapses. In case of no peer returning the item, the MH has to wait for it to appear in the broadcast channel as usual.

We analyzed the performance study of a conventional scheme and COCA in a push-based environment with flat disk and broadcast disk [1] to structure the broadcast, employing (1,*m*)-indexing [10]. When the MHs adopt COCA in a push-based environment, they could improve access latency and tune-in time. The access latency can be reduced, if some peers could turn in the item to the requesting MHs before it appears in the broadcast channel. Likewise, the tune-in time gets shorter, if the requesting MHs receive some *reply* messages before the index is broadcast. Our analysis indicated that the push-based schemes are scalable to the number of MHs and COCA can effectively improve the access latency. However, the access latency slightly gets worse with too many MHs.

2.4 Hybrid Environment

In a hybrid mobile environment, MHs make use of both point-to-point and broadcast channels to retrieve data items from the MSS. There are three aspects to be considered: bandwidth allocation, data allocation and broadcast channel indexing.

Bandwidth allocation is concerned with the amount of channels allocated for push-based broadcasting. The MHs can access hot data items via the allocated broadcast channels to improve system scalability. To access the remaining cold data items, the MHs still have to obtain them from the MSS via point-to-point communication. Data allocation is concerned with the selection of data items to the broadcast channel, based on their access probabilities estimated by observed access frequencies. We maintain an exponentially weighted moving average estimate of the access probability. Since the hottest data items are likely to be cached by most MHs, they need not to be broadcast with the highest frequency. We would thus shift a certain percentage of hottest data items to be broadcast with least frequency.

When an MH encounters a local cache miss, it tunes in to the broadcast channel. If the required item appears in the broadcast channel before the index is broadcast, the MH catches the item. Otherwise, the MH grabs the index and looks up the required item to determine its arrival time. The MH dozes off and then wakes up to catch its desired item. If the MH cannot find the identifier of the item in the index or the latency of accessing it is longer than a latency threshold, the MH switches to retrieve the item from the MSS via the point-to-point channel [8]. Our analysis indicated that though the hybrid environment is more scalable than pull-based environment, the system becomes unstable when the number of MHs is too large.

3 Simulation Model

The simulated mobile environment is composed of an MSS and a default number of 100 MHs. There are 20 wireless channels between the MSS and the MHs, with a downlink bandwidth of 10Mbps and an uplink bandwidth of 100kbps. There is a half-duplex wireless channel for an MH to communicate with its peers with a bandwidth of 2Mbps. The power consumption measurement model is based on [5] which uses linear formulas to measure the power consumption of source MH, destination MH, and other remaining MHs residing in the transmission range of source and destination MHs. The power consumption measurement model of a broadcast environment is slightly different, since there is additional power consumption for the MHs listening to the broadcast channel until they obtain the item or the broadcast channel index.

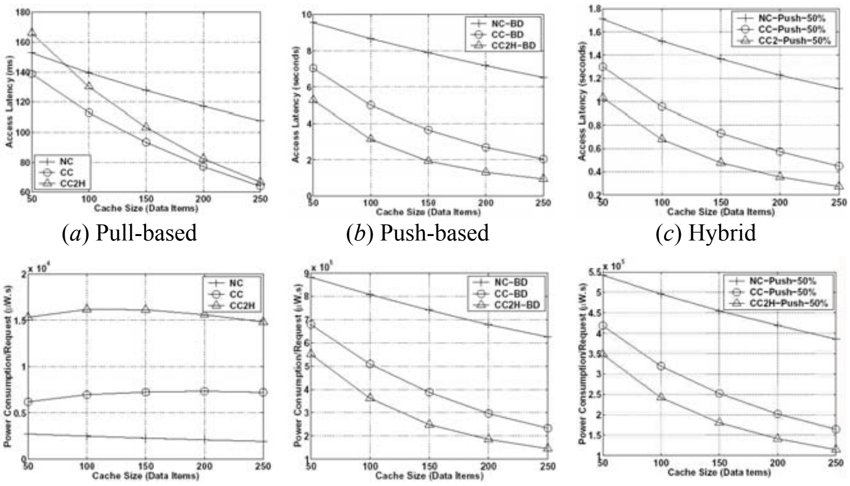
The MHs move based on “random waypoint” model [3], randomly distributed in a region of 1000×1000, which constitutes the service area of the MSS. Each MH has a cache that can hold 100 items, generating accesses to the data items following a Zipf distribution with a skewness parameter θ. The time interval between two consecutive accesses generated by an MH follows an exponential distribution with a mean of 1 sec. The effect of skewness in access pattern is studied by varying θ. To study the effect of common access pattern, we introduce the concept of *CommonAccessRange*, which specifies that a certain percentage of access range is common to all MHs and the remaining access ranges are randomly selected for each MH. If this common access range is 0%, the access ranges for all MHs are randomly selected. If the value is 100%, all MHs share the same common access range. There is a single MSS, with a database containing 10000 data items of size 4KB each. It receives and processes the requests sent by the MHs with a first-come-first-serve policy.

4 Performance Evaluation

In the simulated experiments, we compare the performance of COCA (denoted as **CC**) with a conventional caching scheme without any cooperation among MHs (denoted as non-COCA or **NC**). All schemes adopt LRU cache replacement policy. We consider COCA with two-hop communication (denoted as **CC2H**). In **CC2H**, the MHs can search the required items in their peers within a distance of two hops. We study **NC**, **CC** and **CC2H** in pull-based, push-based and hybrid environments. For the push-based and hybrid environments, broadcast disk [1] (denoted as **BD**) broadcast scheduling algorithm is adopted. All simulation results are recorded only after the system reaches a stable state, when all the caches of all MHs are fully occupied, in order to avoid a transient effect. We conduct the experiments by varying cache size, client population and access patterns. The performance metrics include access latency and power consumption. The access latency is the sum of the transmission time and the time spent on waiting for requested communication channels, if they are busy.

4.1 Experiment #1: Effect of Cache Size

Our first experiment studies the effect of cache size on system performance by varying the cache size from 50 to 250 data items. The results are shown in Fig. 2.



(*a*) Pull-based (*b*) Push-based (*c*) Hybrid

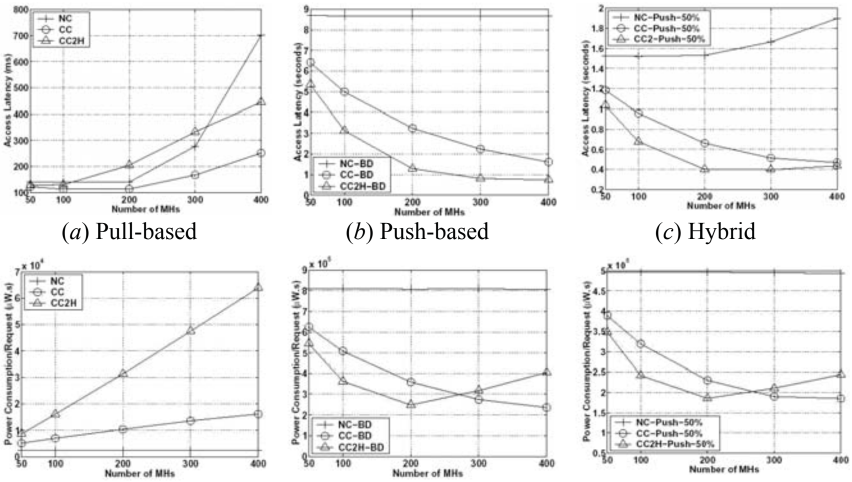
(*d*) Pull-based (*e*) Push-based (*f*) Hybrid

Fig. 2. Effect of cache size.

Fig. 2*a*, 2*c* and 2*e* show that all schemes exhibit better access latency with larger cache size. Since the MHs can achieve a higher LCH ratio with larger cache, they enlist less help from the MSS. For the MHs adopting COCA schemes, they enjoy both a higher LCH ratio and a higher GCH ratio. In Fig. 2*a*, the access latency of **CC2H** is worse than **CC**, though **CC2H** yields a higher GCH ratio. When the MHs search for their required items at a distance of two hops, there are more messages generated in the wireless network than one-hop searching. In **CC2H**, the timeout is also longer, so the MHs have to spend more time in detecting a global cache miss. Furthermore, a side-effect of a higher GCH ratio is that the MHs have to consume more power to handle global cache queries, such as receiving broadcast queries from their peers and forwarding more data items to the requesting MHs. A scheme with higher GCH ratio could thus incur higher power consumption. Thus in Fig. 2*d*, the MHs adopting **CC2H** enjoy a higher GCH ratio, but consume more power than **CC**. Likewise, the power consumption of **CC** is higher than **NC**. In Fig. 2*b* and 2*e*, a broadcast channel access incurs longer access latency and higher power consumption than a global cache access in the broadcast environment, due to the doze-and-catch data access model. In the hybrid environment, the results exhibit a similar behavior as in the push-based environment, i.e., a global cache access incurring shorter access latency and less power consumption than an MSS access, as shown in Fig. 2*c* and 2*f*.

4.2 Experiment #2: Effect of Client Population

Our second experiment studies the effect of client population with number of MHs in the system varying from 50 to 400. The results are depicted in Fig. 3.



(*a*) Pull-based (*b*) Push-based (*c*) Hybrid

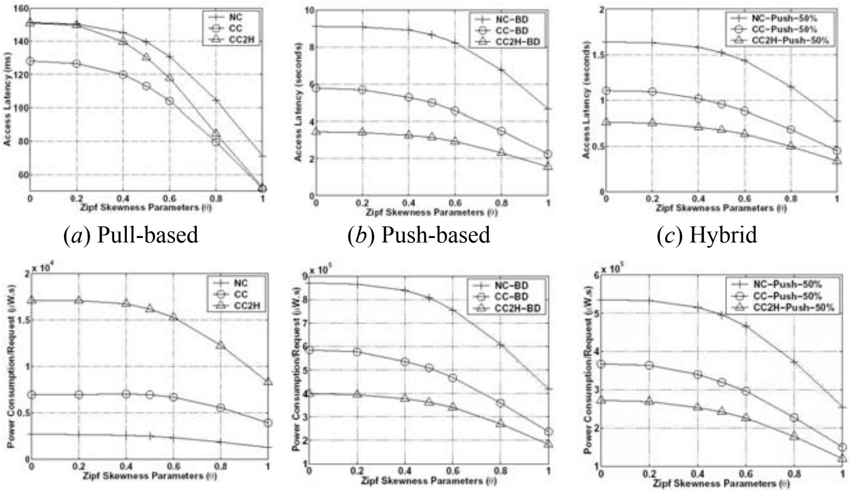
(*d*) Pull-based (*e*) Push-based (*f*) Hybrid

Fig. 3. Effect of client population.

In the pull-based environment, the access latency of all schemes gets worse with more MHs, as shown in Fig. 3*a*. When the MHs encounter local cache miss, they need not rely solely on the MSS because they can enlist help from their peers. Such nice property can be considered as an indirect load sharing technique that improves system scalability. When an MH can reach more peers with higher client population, it stands a higher chance to obtain the required items from the Peer Cache layer, and a higher GCH ratio. The power consumption would get higher, as shown in Fig. 3*d*. In the push-based environment, the performance of **NC** is not affected by number of MHs. On the other hand, the access latency of COCA schemes improves with more MHs, as depicted in Fig. 3*b*. This is because the MHs adopting COCA schemes record a higher GCH ratio with more MHs, coupled with the fact that the latency and power consumption of a global cache access is lower than a broadcast channel access. For **CC2H**, the power consumption initially drops and then rises with increasing number of MHs, as illustrated in Fig. 3*e*. When the MHs achieve a higher GCH ratio, they can conserve more power, relying more on the lower power peer than broadcast access. However, when the number of MHs further increases, the power consumption on peer-to-peer communication, such as receiving/forwarding more broadcast requests and data items and discarding unintended messages increase more rapidly, offsetting the benefit of higher GCH ratio. In the hybrid environment, the access latency of **NC** gets worse with number of MHs beyond 200. For COCA schemes, the access latency and power consumption of **CC** improve with higher client population; however, the performance of **CC2H** becomes a bit worse with too many MHs, as shown in Fig. 3*c* and 3*f*. This is due to similar reasons as in pull-based and push-based environments.

4.3 Experiment #3: Effect of Access Skewness

In our third set of simulated experiments, we study the effect of skewness in access pattern by increasing the skewness parameter value θ from zero to one. The system performance is exhibited in Fig. 4.



(*a*) Pull-based (*b*) Push-based (*c*) Hybrid

(*d*) Pull-based (*e*) Push-based (*f*) Hybrid

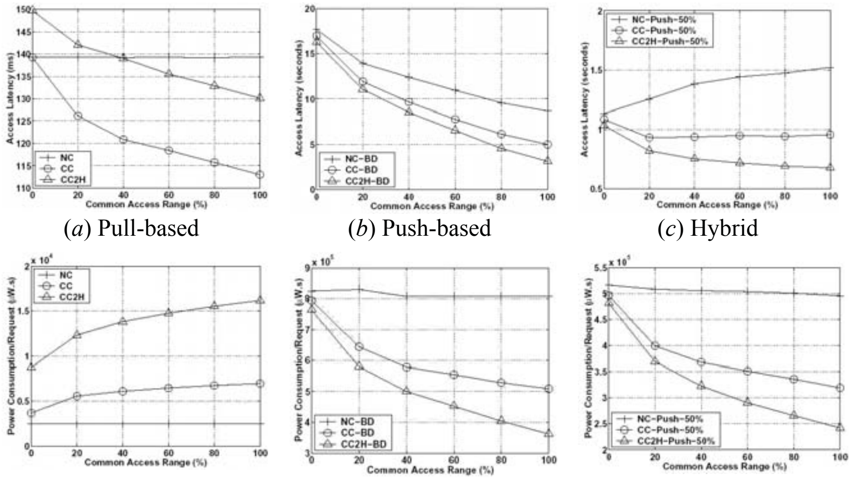
Fig. 4. Effect of access skewness.

When θ is zero, the MHs access the data items uniformly, and there is a higher chance for them to encounter local cache misses. As θ rises, the client access pattern becomes more skewed and the MHs can find more required items in their local cache. The higher the LCH ratio, the better the performance the MHs can achieve. In COCA schemes, when θ is small, the MHs need more help from their peers due to a lower LCH ratio, but the performance gradually improves with more concentrated access.

Presence of hot spots has a similar impact as increasing the effective cache size. The relative performance of the protocols therefore resembles that of Experiment #1.

4.4 Experiment #4: Effect of Common Access Pattern

Our final experiment aims to find out the effect of common access pattern for MHs on system performance by varying it from 0 to 100%. The results are illustrated in Fig. 5.



(*a*) Pull-based (*b*) Push-based (*c*) Hybrid

(*d*) Pull-based (*e*) Push-based (*f*) Hybrid

Fig. 5. Effect of common access range.

In the pull-based environment, the performance of **NC** is not affected by varying the common access range, as depicted in Fig. 5*a* and 5*d*, since there is no cooperation among the MHs and the size of common access range does not have any influence on individual MHs. However, the access latency of the MHs adopting COCA schemes improves with increasing common access range because they enjoy a higher GCH ratio at the expense of higher power consumption. In the push-based environment, the access latency and power consumption of all schemes improve with increasing commonality, as shown in Fig. 5*b* and 5*c*. When the MHs are interested in more common items, the hot items can be allocated to the broadcast channel spinning faster, so the access latency reduces. **CC** and **CC2H** perform better than **NC** because the MHs adopting COCA schemes have a chance to obtain their required items from the peers. In the hybrid environment, Fig. 5*c* shows that the MHs adopting **NC** experience higher access latency, with a larger common hot spot. More data items qualify as being hot, making a less effective use of the limited broadcast channels. Thus, the access latency would increase slightly. Meanwhile, peer cache could still contribute to cache hit and access latency in COCA schemes. There is a higher GCH ratio, when MHs share more common hot data items. The higher GCH ratio is, the better performance the MHs can achieve in terms of power consumption, which is lower for a global cache access than a broadcast channel access, as shown in Fig. 5*f*.

5 Conclusion

In this paper, we extended the COCA mobile cooperative caching scheme for MHs over pull-based, push-based and hybrid environments, adopting multi-hop searching. The performance is evaluated through a number of experiments. The results show that COCA with single-hop searching always performs better than traditional caching scheme in the pull-based environment in terms of access latency. However, the cost of MHs adopting COCA is higher power consumption. In the broadcast and hybrid mobile environments, COCA yields better performance in access latency and power consumption compared with conventional caching scheme. COCA with multi-hop searching further improves system performance in the push-based and hybrid environments. It also shows that COCA can improve the system scalability by sharing system workload among MHs.

References

1. S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: Data management for asymmetric communication environments. In *Proceedings of SIGMOD*, pages 199–210, 1995.
2. S. Acharya, M. Franklin, and S. Zdonik. Balancing push and pull for data broadcast. In *Proceedings of SIGMOD*, pages 183–194, 1997.
3. T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, **2**(5):483–502, August 2002.
4. C.Y. Chow, H.V. Leong, and A.T.S. Chan. GroCoca: group-based peer-to-peer cooperative caching in mobile environment. *IEEE Journal on Selected Areas in Communications*, **25**(1):179–191, January 2007.
5. L.M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *Proceedings of INFOCOM*, pages 1548–1557, 2001.
6. T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. In *Proceedings of INFOCOM*, pages 1568–1576, 2001.
7. H. Hayashi, T. Hara, and S. Nishio. Cache invalidation for updated data in ad hoc networks. In *Proceedings of International Conference on Cooperative Information Systems*, pages 516–535, 2003.
8. Q. Hu, D.L. Lee, and W.C. Lee. Performance evaluation of a wireless hierarchical data dissemination system. In *Proceedings of MobiCom*, pages 163–173, 1999.
9. J.L. Huang, M.S. Chen, and W.C. Peng. Exploring group mobility for replica data allocation in a mobile environment. In *Proceedings of CIKM*, pages 161–168, 2003.
10. T. Imielinski, S. Viswanathan, and B.R. Badrinath. Data on air: Organization and access. *IEEE Transactions on Knowledge and Data Engineering*, **9**(3):353–372, May 1997.
11. S. Lim, W.C. Lee, G. Cao, and C. R. Das. A novel caching scheme for internet based mobile ad hoc networks. In *Proceedings of IEEE International Conference on Computer Communications and Networks*, pages 38–43, 2003.
12. M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *Proceedings of MobiHoc*, pages 117–127, 2001.
13. F. Sailhan and V. Issarny. Cooperative caching in ad hoc networks. In *Proceedings of International Conference on Mobile Data Management*, pages 13–28, 2003.