AN EFFICIENT COOPERATIVE CACHE APPROACH IN MOBILE INFORMATION SYSTEM

Thu Tran Minh Nguyen and Thuy Thi Bich Dong
IS Department, Faculty of IT, University of Science, VNU, HCMC, Vietnam

Keywords: Mobile information systems, Cooperative computing, Cooperative caching, Cache discovery, Cache replacement, Mobile P2P network.

Abstract: Data availability in mobile information systems is lower than in traditional information systems due to its limitations of wireless communication such as low bandwidth, instability, and easy disconnection. Cooperative data caching is an attractive approach related to this problem because it allows sharing and coordinating the cached data among multiple mobile users in the network. So it can improve the performance of the system and the accessibility of data items. However, if we do not have a good cooperative caching approach then the cache hit ratio and response time may become ineffective. In this paper, we propose an efficient approach for cooperative caching in mobile information systems namely MIX-GROUP. We evaluate and demonstrate the experimental results on datasets by using NS2 simulator. MIX-GROUP is also proved the effectiveness by comparing it with the existing other approaches such as COCA and GROUPCACHING. The experimental results present that MIX-GROUP is more effective than those other approaches.

1 INTRODUCTION

Wireless communications and mobile devices have become more and more popular in our life. People use mobile devices such as mobile phones, pocket PCs, laptops … to retrieve information from other applications anywhere and at anytime they need. This type of applications is based on mobile information systems or mobile databases. There are many mobile clients connected to a server through wireless links in a mobile information system. However, mobile information systems raise new problems that do not exist in traditional information systems. Wireless links are unstable, which means the bandwidth of the uplink channel (mobile client to base station) is usually narrower than that of the downlink (base station to mobile clients). Mobile devices often have small storage capacity and weak power. Furthermore, a mobile user (client) is usually disconnected from the base station (server) because of unstable wireless links or out-of-battery itself. Therefore, researchers have explored several solutions for effective data management in mobile platform. Cooperative caching in mobile peer-to-peer systems is more interesting recently. In mobile peer-to-peer system, the clients will look up the required data at their own local cache or other peer's local cache before sending the data requests to the server. This means that the cooperative caching explores how to collaborate multiple caches to achieve a better performance. By cooperatively caching the frequently accessed information, mobile clients do not always have to send requests to the base station. This technique can get lower mobile host communication overhead and energy consumption as well as reduce query latency. However, mobile peer- to-peer systems have brought up new challenges for researching on routing, resource discovery, data retrieval, and data consistency control, security and privacy management.

In this paper, we will propose a cooperative cache approach in mobile information system. Our proposed approach addresses three problems including cache discovery, cache admission and cache replacement. We also evaluate the effectiveness of our approach by some simulation experiments. Especially, experiment results are compared with COCA (Chow, 2004) and GroupCaching (Ting, 2007) to prove the effectiveness of proposed approach.

2 RELATED WORKS

Cooperative caching improves the system performance because it allows sharing and coordinating the cached data among multiple mobile users in the network. A variety of popular cooperative caching strategies have already been studied such as: Zone Cooperative scheme (Chand, 2007); proactive approach for cooperative Caching (Kumar, 2010); Cache Data, Cache Path, Hybrid Cache (Yin, 2004); COCA (Chow, 2004); GroCOCA (Chow, 2004); Cluster Cooperative Caching (Chand, 2006); COOP (Yu Du, 2005 & 2009). However, none of all these researches provides a complete solution for cooperative caching. Each of them has both advantages and disadvantages. In this section, we only analyse approaches which are used and compared with our approach in more detail.

COCA (Chow, 2004) is proposed by Chow et al. In this cooperative caching protocol, the mobile node shares its cache contents with each other to reduce the number of server requests and access miss ratio. However, access latency is quite high when the mobile hosts encounter a global cache hit.

Yu Du & et al proposed a cooperative caching scheme called COOP for MANETs (Yu Du, 2005 & 2009). To improve data availability and access performance, COOP addresses two basic problems of cooperative caching. For cache resolution, COOP uses the cocktail approach that consists of two basic schemes: hop-by-hop resolution and zone-based resolution. By using this approach, COOP discovers data sources, which have less communication cost. For cache management, COOP uses the inter- and intra-category rules to minimize caching duplications between the nodes within a same cooperation zone. This improves the overall capacity of cooperated caches. Disadvantage of this scheme is the flooding, which introduces extra discovery overhead.

Group Caching Scheme (Ting, 2007) maintains localized caching status of one-hop neighbors for performing the tasks of data discovery, cache placement, and cache replacement when a data request is received in a mobile host (MH). Each MH and its one-hop neighbors from a group by using “Hello” message mechanism. In order to utilize the cache space of each MH in a group, the MHs periodically send their caching status to its group. Thus, when caching placement and replacement need to be performed, the MH selects appropriate group member to execute the caching task. In this scheme, the biggest concern is the energy consumption in MHs and constrain of wireless bandwidth. Therefore, in this GroupCaching scheme, each MH only maintains one – hop neighbors in a group (Radhamani, 2010).

The cooperative caching scheme in our study named MIX-GROUP. MIX-GROUP scheme is also built basing on the favourable characteristics of the other ones, such as it inherited the idea of marking data item’s label when data items are cached into MU’s local cache. This idea is proposed in COOP scheme (Yu Du, 2005 & 2009). However, MIX-GROUP resolved the problem of flooding that COOP scheme as well as other ones has met. We will present the proposition of MIX-GROUP scheme in detail in Section 3.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS

3 COOPERATIVE CACHING MODEL

3.1 Description of Model

We assume that the cooperative caching architecture has only one server and many mobile users (MU) as shown in Figure 1. The server is also called the *base station* (BS). Each BS controls MUs in its service zone. The communication between the BS and MUs is wireless link. MU is the mobile devices used to send requests to server and can move freely in one cell or from one cell to another cell. We are especially interested in cooperating data among MUs in network in this model. Each MU has a service zone to communicate with other one. The connection among MUs is P2P wireless network.

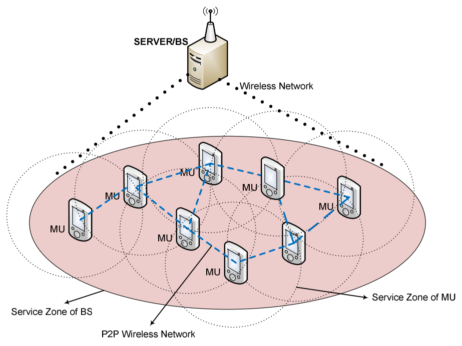

Figure 1: Proposed Cooperative Caching Architecture-MIXGROUP.

In this architecture, the BS acts as the server provide data items for MUs and it can be updated data items. MU plays the role of both the client and the server. MU acts as the client when it sends the requests to retrieve data items from the BS or the other MU and as the server when it provides data items for another MU in network. However, MU does not allow updating data items in system.

3.2 Principle of Data Search Process

When the MU resource (example as MU 2 in Figure 2) emits the data request, MU resource will look for requested data items in its local cache. Local cache is the place in which caches data items requested by previous queries. If the required data items are not found in the local cache (or cache miss), MU resource will send the data request to the MUs neighbour to seek data items. MUs neighbour are the MUs that belong to MU resource's region. MU resource links to MUs neighbour by one hop (example as MU 8 in Figure 2). Then if they can't be found out in the MU resource's region, the data request will be sent to the MUs network to find requested data items. MUs network are the MUs that do not belong to MU resource's region. MU resource links to MUs network by at least two hops (example as MU 5 in Figure 2). Lastly, if requested data items are not found at MUs neighbour or MUs network, the data request will be sent to the BS for requesting data.

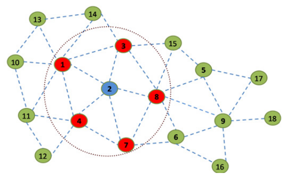

Figure 2: Illustration for cooperative caching among MUs.

3.3 Processes at each MU Peer

In this section, we will present the architecture of each MU in more detail. Each MU has three main layers: *User Application, Middleware* and *Network Protocol* as in Figure 3.

- *User Application layer* is responsible for receiving the data requests emitted by users. Then it transmits these requests to Middleware layer in order to be processed. *User Application layer* also receives the requested data responded from Middleware layer and then displays them to users.
- *Network Protocol layer* is used to communicate and transmit messages among the system's elements.
- *Middleware layer* is the main layer that serves for researching requested data items. *Middleware layer* has three modules: *Local Query Process, In-zone Query Process* and *Out-zone Query Process*. *Local Query Process* module is responsible for seeking data items from local cache. It also resolves for data admission and data replacement. *In-zone Query Process* module takes the part of searching data in the region that includes the neighbours of the MU resource. *Out-zone Query Process* module assigns a task to find requested data at the MUs network of the MU resource. The detail of processes and metadata will be presented in the next section.

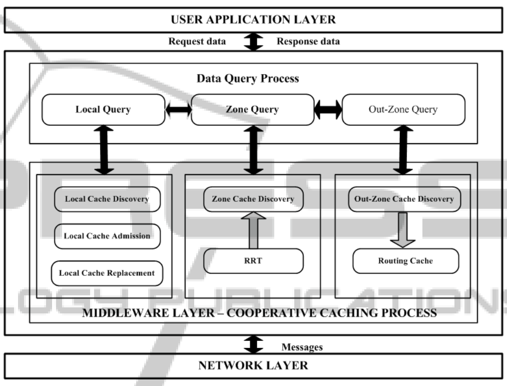

Figure 3: Detail of the middleware layer.

3.4 Detail of Processes and Metadata

3.4.1 Local Query Process

a) Local Cache Discovery

Each MU has a local cache (*LC*). *LC* stores the frequently accessed data items to serve local data search process. Caching data items in the *LC* of each MU helps reduce latency and increases accessibility. The *LC* structure includes {*id, f, t, L, TTL, D, S*} where *id* is the identification of the cached data item, *f* is the frequency of cached data item accessed by other MUs; *L* is the label of the cached data item. Basing on *L*, we will recognize a data item as the primary data or the secondary data; *t* is the timestamp of cached data item. *TTL* value is the living time of a data item, *D* is data value of data item *d, S is* a size of data item *d*. When a MU requests for data item *d*, it first will look for *d* from its local cache. A local cache hit occurs if the request is satisfied in local cache.

b) Cache Admission

When a MU receives the response for the requested data item, a cache admission control is triggered at MU to decide whether the data item should be cached. The idea of cache admission control is known as each cached data item will be

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

marked as the primary data (PD) or the secondary data (SD). This process must always ensure that there is only one primary data copy in cooperative zone. That means once a MU fetches a data item, it labels the data item as the primary copy if the data item comes from a MU beyond the zone radius. Otherwise, if a data items comes from within the zone radius, we need to consider whether the data provider labels the data item as primary or secondary. If one of the providers already labels its copy as primary, the new copy would be secondary since we do not intend to have duplicated primary copies in cooperative zone. On the other hand, if all providers tag its own copy as secondary, the new copy is primary one. The idea of classifying cached data items is the same way that Y. Du et al used in COOP scheme (Yu Du, 2005 & 2007). However, the difference between COOP and MIX-GROUP scheme is MIX-GROUP's zone has one hop whereas COOP's zone has multi hops.

SCIENCE AND TECHNOLOGY PUBLICATIONS

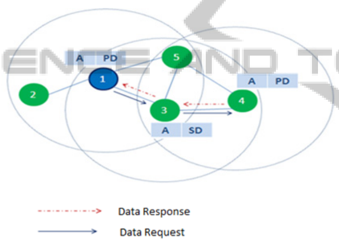

Figure 4: The Illustration for classifying cached data.

An example is illustrated in Figure 4. When MU 3 requests data item A, MU 4 replies and marks A's label as primary. Because MU 4 is in MU 3's zone, so data item A is copied at MU 3 as the secondary data. Then MU 1 requests data item A, MU 3 responds A to MU 1 as the secondary data. MU 1 will caches A as the primary data because MU 1 has the same zone with MU 3, but it has other zone with MU 4.

c) Cache Replacement

Cache replacement means that MU decides which data item should be removed from its local cache when the caching space is not enough for caching the new data items. The policy of MIX-GROUP's cache replacement is based on factors: *L*, *f*, *TTL*, *t* and *S*. These factors, as described above *f* is the access count of the data items that reflects a data item's popularity in the region; *L* is used to classify the data items as primary or secondary (primary data item is more priority than secondary one); *TTL* is the living of the data item, *t* is timestamp of data item, *S* is size of data item.

When the local cache has not enough space for storing new data item, cache replacement policy will choose the data item that has invalid *TTL value*. In the case, there is not invalid data item; we choose the secondary data item whose β value is smallest. This decision is selected because cooperating caching is not only on the behalf of the caching MU but also basing on the other MUs need. β value is evaluated as β = f/ (t*S).

3.4.2 In-zone Query Process

When the data request is missed and is not responded from *LC*, the request will be sent to the neighbors of the MU request. In this process, each MU has an information profile named *ZoneData*. *ZoneData* stores the information of the neighbors and data items that they sent to the MU request before. The *ZoneData* structure includes *{id, L, id_{sender}, t }* where *id* is the identification of requested data item; L is the label of data item cached at MU sender; *id_{sender}* is the identification of MU sender; *t* is the time that MU resource gives this data item. Based on the *ZoneData*, MU resource can find out the MU destination that is caching the request data in its zone. *ZoneData* is as a way to avoid blind broadcasting to all MUs in the region that the other cooperative caching schemes met. By using this way, our proposed scheme will reduce the communication cost and increase the system performance.

3.4.3 Out-zone Query Process

When the data request is not responded from the neighbours of the MU resource, it will be sent to MUs network. In this process, we used the information of the *OutZoneData* table to discover the MU destination. The *OutZoneData* structure includes *{id, ID_{sender}, m, t}*, where *id* is the identification of the data item *d*; ID_{sender} is the identification of MU network which is storing the data item *d;* *m* is the number of hops from MU requester to that MU network; *t* is the time that MU requester gets the data item *d*. The data discovery path selects MUs network that *m* value is minimum. This approach will increase data response with the least hops.

4 SIMULATION RESULTS

4.1 Simulation Environment and Experiments

We built a simulation scenario and implemented by

NS2 simulation (Sinh, 2011). The simulation parameters are configured as in Table 1. The goal of the simulation is the quality test of the MIX-GROUP scheme. The performance metrics used for quality testing of MIX-GROUP scheme include: *average response time for a data request, average number of messages, and cache hit ratio.* These metrics are evaluated basing on the MU's cache size and the number of MUs varying in system. The experimental results are compared with those of other schemes such as GROUPCACHING (Ting, 2007) and COCA (Chow, 2004). The results showed in the below plots prove the effectiveness of our proposed approach.

Table 1: Simulation parameters.

Parameters	Settings
Simulation area	950m x 510m
Total #MU	10 – 100
Bandwidth	2 Mb/s
Radio range	250 m
Cooperation zone radius	2-3 hops
Total # data items	1000
Client cache size	2000 – 20000 KB
Data item size	1 – 1000 KB

4.2 Simulation Experiment Results

4.2.1 Effect based on Cache Size

a) Average number of messages

The experiment result in Figure 5 showed that the average number of messages of three approaches reduces when MU's cache size increases. However, MIX-GROUP has the average number of messages is lowest. This result is archived because MIX-GROUP has used the information of ZoneData and OutZoneData to avoid flooding messages while searching data in system.

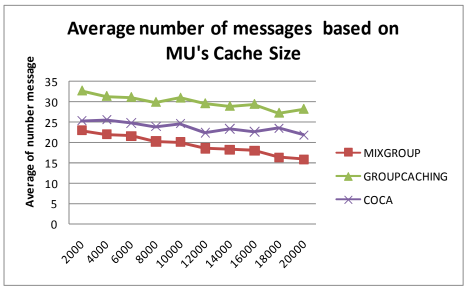

Figure 5: Effect of average number of messages based on MU's cache size.

b) Average response time

The experimental result of average response time is shown in Figure 6. Average response time is steadily decreased in each scheme when MU's cache size increases. The plot showed that GROUPCACHING's average response time rapidly decreases but still higher than MIX-GROUP.

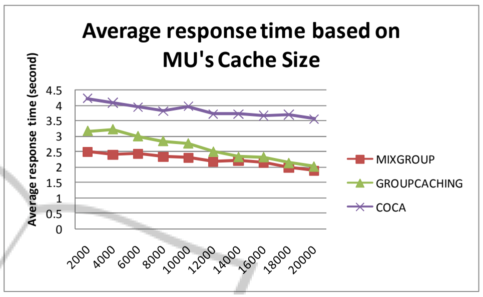

Figure 6: Effect of average response time based on MU's cache size.

c) Ratio of cache hit

Cache hit ratio result is shown in Figure 7. Cache hit means the data request is responded from cooperative zone, while cache miss means the data request is responded from BS. The plot of experiment results presented GROUPCACHING's cache hit ratio is quite effective. However, cache hit ratio of MIXGROUP is more effective when MU's cache size is large. Achieving this result based on effective cache admission and replacement strategy.

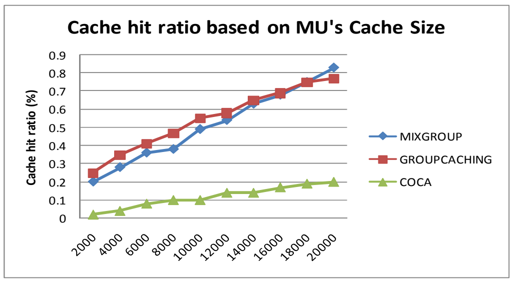

Figure 7: Effect of cache hit ratio based on MU's cache size.

4.2.2 Performance Results based on Number of MUs

a) Average number of messages

The average number of messages of three schemes that evaluated based on the different MU number shown in Figure 8. The result plot presented the average number of messages of COCA and GROUPCACHING steadily increases when the number of MU increases while MIX-GROUP's average number of message reduces. This result proves MIX-GROUP is very efficient energy consumption when the MU number increases in system.

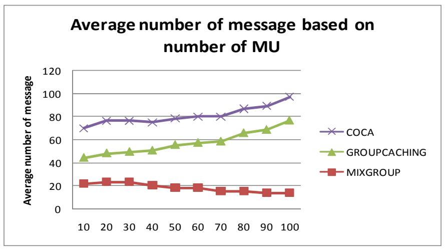

Figure 8: Effect of average number of messages based on MU number.

b) Average response time

The larger the MU number of system is, the more cooperative cache the MUs get. The average response time results are shown in Figure 9. The plot is shown that MIX-GROUP has average response time is lowest. This result has got because MIX-GROUP used the OutZoneDate talbe for searching data at the MUs network.

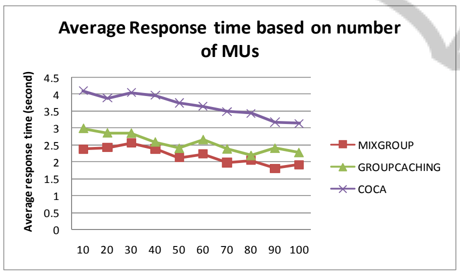

Figure 9: Effect of average response time based on MU number.

c) Ratio of cache hit

The experimental result is shown in Figure 10. Similar to cache hit ratio based on cache size, plot of cache hit ratio based on the different MU number in three schemes increases when the MU number increases. GROUPCACHING is quite effectiveness but when the MU number highly increases MIX-GROUP is more effective than GROUPCACHING. This result shows that our proposed cooperative caching scheme is very efficient.

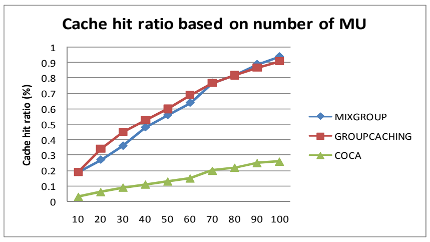

Figure 10: Effect of cache hit ratio based on MU number.

5 CONCLUSIONS

Cooperating caching is more interesting problem in mobile environment. In this paper, we proposed a cooperative caching model named MIX-GROUP, which addresses three basic problems of cooperative caching: cache discovery, cache admission and cache replacement. Experimental results show that MIX-GROUP improved data availability and access performance. A supplemental evidence for our success is the average number of messages, the average response time and cache hit ratio that MIX-GROUP has archive is more effective than COCA and GROUPCACHING. However, we still have not taken care about cache consistency in MIX-GROUP model yet. This will be eventually considered in our future work.

REFERENCES

N. Chand, R. C Joshi, M. Misra, 2007.*"Cooperative Caching in Mobile Ad hoc Networks based on Data Utility"*. International Journal of Mobile Information System 3(1), 19-37.

Prashant Kumar, Naveen Chauhan, L. K. Awasthi, Narottam Chand, 2010. "*Proactive Approach for Cooperative Caching in Mobile Adhoc Networks*". In IJCSI 2010, IJCSI International Journal of Computer Science Issues, Vol.7, Issuse 3, No 8.

Yin, L. Cao, G., 2004. *"Supporting Cooperative Caching in Ad hoc Networks"*. In Infocomm 2004, The 23rd Conference on the IEEE Communications Society.

C. Y. Chow, H. V. Leong and A. Chan, 2004. *"Peer to Peer Cooperative Caching in Mobile Environments"*. In ICDCSW 2004, The 24th International Conference on Distributed Computing Systems, Workshops, pp.528-533.

C. Y. Chow, H. V. Leong, and A. Chan, 2004. "*Cache Signatures for Peer to Peer Cooperative Caching in Mobile Evironments"*. In AINA 2004, The 18th Conference on Advanced Information Networking and Applications, pp.96-101.

C. Y. Chow, H. V. Leong, and A. Chan, 2004. "*Group based Cooperative Cache Management for Mobile Clients in Mobile Environments*". In ICPP 2004, The 33rd Conference on Parallel Processing, pp.83-90.

N. Chand, R. C Joshi, M. Misra, 2006. "*An efficient Caching Strategy in Mobile Ad hoc networks Based on Clusters*". In WCON 2006, The 3rd International Conference on Wireless and Optical Communications Networks IEEE, Los Alamitos.

Yu Du, Sandeep K. S. Gupta, 2005. "*COOP – A cooperative caching service in MANETs*", In ICAS/ICNS 2005, Autonomic and Automous System and International Conference on Networking and Service.

Yu Du, Sandeep K. S. Gupta, 2009. "*Improving on-demand data access effeciency in MANETs with cooperative caching*". In Journal Ad hoc Network, vol 7, issue 3, pp.579-598.

G. Radhamani and S. Umamaheswari, 2010. "*Comparison of Cooperative Caching Strategies in Ad-Hoc Network (MANET)*". In Communications in Computer and Information Science, Vol 90, part 2, Springer.

Y. W. Ting, Y. K. Chang, 2007. "*A Novel Cooperative Caching Scheme for Wireless Ad hoc Networks: Group Caching*", In NAS 2007, The International Conference on Networking, Architecture, an Storage), IEEE, Los Alamitos.

Sinh V. Vo, Quang X. Pham, 2011. *"A Group Cooperative Caching Approach for Mobile Information Systems and Building some Experiments*", Student Final Project Report, Unversity of Science, Vietnam National University - Ho Chi Minh (report in vietnamese).

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS