International Journal of Engineering Research and Development
e-ISSN: 2278-067X, **p**-ISSN: 2278-800X, www.ijerd.com
Volume 6, Issue 11 (April 2013), PP. 23-30

Group Caching: A novel Cooperative Caching scheme for Mobile Ad Hoc Networks

Suhani N. Trambadiya[1], Pinaki A. Ghosh[2], J. N. Rathod[3]
[1,2,3] Atmiya Institute of Technology & Science, Rajkot, Gujarat, India.

Abstract:- A mobile host can communicate with other mobile host from anywhere and at any time in the mobile ad hoc network. The accessibility of data objects can be improved with the help of cooperative caching scheme. However, Access latency becomes longer significantly and cache hit ration is reduces due to mobility of data objects while considering factors like energy consumption in battery and limited bandwidth in wireless network. This paper introduces novel cooperative caching scheme called as group caching in mobile ad hoc networks which allows each mobile host and its 1-hop neighbours form a group. In a group, caching status is exchanged and maintained periodically. In this group caching, cache space of mobile host utilized efficiently and therefore there is reduction of redundancy of cached data and reduction of average access latency. We evaluate the performance of this group caching scheme with the help of simulator and compare the results of it with existing schemes like CacheData and ZoneCooperative. Experimental results show that average latency is reduced by about 5% to 25% and cache hit ration is increased by about 3% to 30% while comparing with other schemes.

Keywords:- Include at least 5 keywords or phrases.

I. INTRODUCTION

All manuscripts must be in English. These guidelines include complete descriptions of the fonts, spacing, and related information for producing your proceedings manuscripts.

In the last years, research studies in MANETs mostly focus on designing efficient multi-hop ad hoc routing protocols for packet forwarding between two nodes. However, the ultimate goal of a routing protocol is to create a routing path for data communication. Therefore, how to improve the accessibility of data object by using caching techniques is another important issue.

Caching techniques are an efficient solution for increasing the performance in message or data communication. It has been widely used in different fields such as CPU design, multi-processor, memory architecture or router design. Furthermore, Internet uses cache placement and replacement in proxy servers [4] and cooperative caching architecture [5] to reduce the network traffic and average latency of data query significantly. The original idea of caching is that the data accessed by MHs has the properties of temporal and spatial locality. Higher temporal and spatial locality ensures that most accesses will go to the data that were accessed recently in the past and that reside in the cache. Therefore, caching frequently requested data can improve the performance of data communication.

In a MANET, if the mobile hosts cache some frequently requested data, the cached data can be served for others later. The requester needs not to retrieve the data from the remote server (data source) and then requests the data from the caching node. As a result, the data accessibility is enhanced. Furthermore, cooperative caching techniques allow several caching nodes to coordinate the caching tasks such as the redirection of data requests, cache placement, or cache replacement. Existing cooperative caching schemes [1] [3] [6] [7] [8] [11] [12] allow a data object can be cached by multiple hosts for enhancing the accessibility of data objects.

When we employ caching techniques in MANETs for data communication, there are some issues and challenges like mobility of mobile hosts, limited wireless bandwidth and power consumption in battery. MANETs may be partitioned into many independent networks due to movement of mobile hosts. Thus, the requester can’t retrieve desired data from the remote server called as data source in another network. Therefore entire data accessibility will be reduced and caching node may be disconnected from the network for saving power too. Thus, cached data in mobile host may not be retrieved by other mobile host and then usefulness of the cache is reduced.

According to caching status of other mobile hosts, the mobile hosts also decide the caching policy. However, existing cooperative caching schemes in a MANET lack an efficient protocol among mobile hosts to exchange their localized caching status for caching tasks. Thus, we propose here novel cooperative caching scheme called group caching (GC), which maintains localized caching status of 1-hop neighbours for performing tasks of data discovery, caching placement and replacement whenever data request is received in mobile host. With the help of “Hello” message mechanism each mobile host and its 1-hop neighbour form a group. The mobile hosts periodically send their caching status in a group in order to utilize cache space of each mobile host in a group. Thus, Mobile host selects appropriate group member to execute caching task in the group when caching placement and replacement needs to be performed. Mobile hosts know caching status of their neighbours in this proposed cooperative caching algorithm. The redundancy of cached data objects can be reduced based on the proposed group caching because of the mobile hosts which can check caching status of other group members for deciding the cache placement and replacement. Each mobile host can store more different data objects in a group because more cache space can be utilized and then increases data accessibility.

Rest of this paper is organized as follows: Section 2 represents reviews of the related works for cooperative caching schemes in mobile ad hoc networks. Section 3 presents proposed group caching scheme. The experimental results of group caching scheme are shown by Section 4 which are compared with existing schemes. Section 5 represents conclusion and future work.

II. RELATED WORKS

A. CacheData and CachePath

The CacheData [1][12] scheme considers the cache placement policy at intermediate nodes in the routing path between the source and the destination. The node caches a passing-by data item locally when it finds that the data item is popular, i.e., there were many requests for the data item, or it has enough free cache space. Since CacheData needs extra space to save the data, it should be used prudently. A conservative rule is proposed as follow: A node does not cache the data if all requests for the data are from the same node. However, there is no cooperative caching protocol among MHs. Each MH independently performs the caching tasks such as placement and replacement. CachePath [1][12] is also proposed for redirecting the requests to the caching node. In MANETs, the network topology changes fast and thus, the cached path may become invalid due to the movement of MHs.

B. ZoneCooperative

The ZoneCooperative [11] scheme considers the progress of data discovery. Each client has a cache to store the frequently accessed data items. The data items in the cache satisfy not only the client’s own requests but also the data requests passing through it from other clients. For a data miss in the local cache, the client first searches the data in its zone before forwarding the request to the next client that lies on a path towards server. However, the latency may become longer if the neighbours of intermediate nodes do not have a copy of the requested data object for the request.

C. NeighborCaching

The concept of Neighbour Caching (NC) [13] is to utilize the cache space of inactive neighbours for caching tasks. The basic operations of Neighbour Caching are as follows. When a node fetches a data from a remote node, it puts the data in its own caching space for reuse. This operation needs to evict the least valuable data from the cache based on a replacement algorithm. The data which is to be evicted is stored in idle neighbour node’s storage with this neighbour caching scheme. It requests the data not from far remote source node but from near neighbour which keeps copy of data if node needs data again in the near future. The NC scheme utilizes the available cache space of neighbour to improve the caching performance. However, it lacks of the efficiently cooperative caching protocol among the MHs.

III. PROPOSED GROUP CACHING SCHEME

A. Motivation

Mobile hosts can move arbitrarily anytime and communicate with one another for data transmission in mobile ad hoc networks. Caching performance can be reduces significantly because of the property of dynamic topology. Here caching performance is considered as cache hit ratio and average latency. Cooperative caching can also provide the high accessibility of data objects. Possible reasons are listed as below:

1) The caching nodes may leave the mobile ad hoc network because of the energy of battery is exhausted. We can say cached data of caching nodes can’t give any services to other nodes and will be removed after it reconnects network. On the contrary its content of cache space is empty whenever mobile host join the network. The caching performance will be enhanced if we can utilize its cache space as soon as possible.

2) The cache size of general personal computer is always bigger than cache size of mobile hosts. The cache space in all mobile hosts can’t be utilized effectively if there is no cooperative caching protocol among mobile hosts. Mobile hosts can integrate their available cache space and efficiently its cache data if mobile hosts can know the caching status of their neighbours.

3) On demand routing protocol is preferred to be used for saving energy and bandwidth in mobile ad hoc network. Thus caching node is selected based on routing path between source and destination. Therefore, their cache space become full easily and energy will be consumed faster if multiple routing paths are passed through the same nodes.

We design cooperative caching protocol among mobile hosts in order to deal with above described conditions. Our main goal is to provide a caching and power efficient protocol in mobile ad hoc network. First of all, mobile host and its neighbour form a group.

Each mobile host sends their caching status to its group periodically secondly. Third, mobile host's group members cooperate to perform the tasks of placement and replacement whenever a data object needs to be placed or replaced in mobile host. We assume here that mobile hosts have ability to place data in its group member [13]. The group caching has some of the benefits. (1) It can reduce redundancy of cached data object because of the mobile hosts can check status of caching of other group members whenever receiving a data object. (2) It can store more different data objects than increased data accessibility. (3) Group can store more data objects from destinations than single mobile host because of the group members are cooperative in nature to cache the data objects.

B. Group Caching (GC)

1) *Network Model:* Mobile Ad hoc network is having set of mobile nodes. Mobile nodes are connected wirelessly. There is no fixed infrastructure in the network. Here, we are assuming that network is in connected state. If there is partitioned, each component will be treated as an independent network.

2) *Group Definition:* Each mobile node & it's one hop neighbour s are considered to form a group. One hop neighbour covers in the area of transmission range of a mobile node. Each mobile node is having unique group member ID. We are using mechanism of "Hello" messages to maintain connectivity of group. "Hello" messages are sent periodically from each mobile node as "keep a-live signal" at local level. Based on this scheme mobile node comes to know who are its one hop neighbour. There is advantage of maintaining only one-hop neighbour is reduction of energy consumption in wireless bandwidth network.
Figure 1 shows Architecture of Mobile Ad Hoc network. Figure shows group of node 4. Mobile node 4 and its one-hop neighbours 2,3,5,7,8 form a group. Each mobile node sends their caching status to its group members. In this way when node 4 receives data item, it can check the caching status of each group member and can select the appropriate group member to place the cache data.

3) *Cooperative Caching Tables:* Each mobile node contains two tables (a) Node-table (Self-table), (b) Group Table. (a) **Node table** contains fields: Cached data id, Cached data item, Data source id, Time stamp, Counter Time. **(i)Cached data id** indicates id of data cached by node itself. **(ii)Cached data item** indicates data itself.**(iii)Data Source Id** indicates id of Source node from which data is belong to.**(iv) Time Stamp** indicates time when it is used by application of node at last.**(v) Counter Time** indicates time when data is being updated at last. Node table is updated whenever placement is performed in mobile node. (b) **Group Table** contains fields: Cached data id, Data source id, Group member id, Time stamp, Counter Time. Mobile node updates the group table whenever it receives notification of caching status from the group member. From group table mobile node knows which data objects cached in which group member. Therefor whenever request is received in mobile node, it can search node table and group table to find record of the request for data object exists.

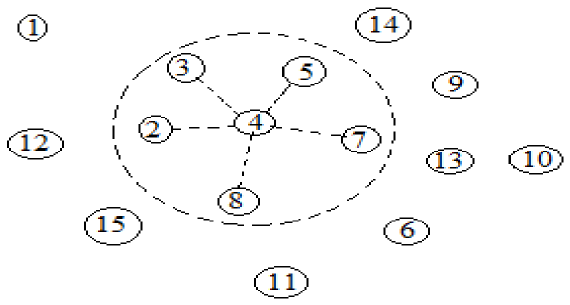
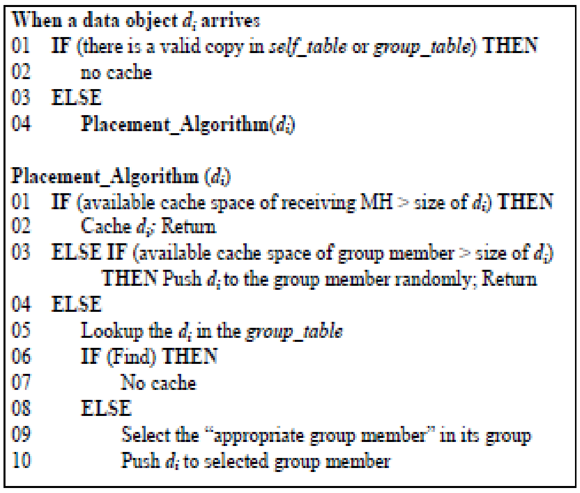


Fig. 1 Group Model for Proposed Architecture

4) *Caching Control Message:* Caching control message is used to exchange the caching status in a group periodically. Caching control message contains fields like: Group member id, Cached data id, Data source

id, Time Stamp, Counter Time, Remaining available cache space. The caching control message is periodically sent by mobile nodes. Mobile node updates its group table whenever it receives caching control message. In this way to perform cache placement and replacement, each mobile node may maintain localized caching status of one-hop neighbours.



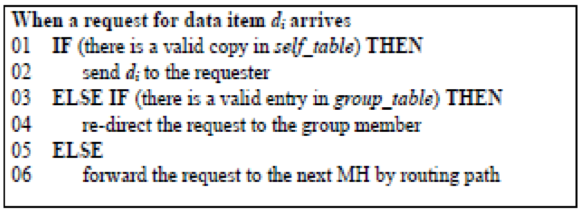
```
When a data object $d_i$ arrives
IF (there is a valid copy in *self_table* or *group_table*) THEN
    no cache
ELSE
    Placement_Algorithm($d_i$)

Placement_Algorithm ($d_i$)
IF (available cache space of receiving MH > size of $d_i$) THEN
    Cache $d_i$; Return
ELSE IF (available cache space of group member > size of $d_i$)
       THEN Push $d_i$ to the group member randomly; Return
ELSE
    Lookup the $d_i$ in the *group_table*
    IF (Find) THEN
        No cache
    ELSE
        Select the "appropriate group member" in its group
        Push $d_i$ to selected group member
```

Fig. 2 Process of Receiving a data items

5) *Placement and Replacement Policy:* This section present scenario how and where to place data object in a member of group When mobile node receives data object from destination. Mobile host knows remaining available cache space of other mobile node in a group and ids and time stamps of their cached data objects. Whenever mobile node receives a data object, it caches the data object if it is having enough cache space otherwise receiving mobile node checks for available cache spaces of its member of a group. The receiving mobile node puts the data object to group member randomly if available cache space of any group member is sufficient to store the data objects.

The receiving mobile node lookups the group table to see if there exists a group member that already caches data object if available cache space of every group member is not sufficient to cache received object. If yes data object is not cached. If it's no, receiving mobile node selects "appropriate group member" which is having oldest timestamp of cached data object in group table and sends data object to that group member to do placement. When group member receiving a data object from receiving mobile node, it repeatedly performs LRU replacement operations to increase available cache space until received data object can be cached.

6) *Data Discovery Process:* Process of data discovery is a process of searching in caching of nodes for requested data object. In a group caching scheme, when any requester node wants to retrieve a data object from data source, first of all it checks its node table to see if data exists locally. If yes, it returns data object to the application. It means cache hit. If no it looks its group table for the data object, if yes, requester redirects data request to that group member, and wait for replied data object. It means global group cache hit.



```
When a request for data item $d_i$ arrives
IF (there is a valid copy in *self_table*) THEN
    send $d_i$ to the requester
ELSE IF (there is a valid entry in *group_table*) THEN
    re-direct the request to the group member
ELSE
    forward the request to the next MH by routing path
```

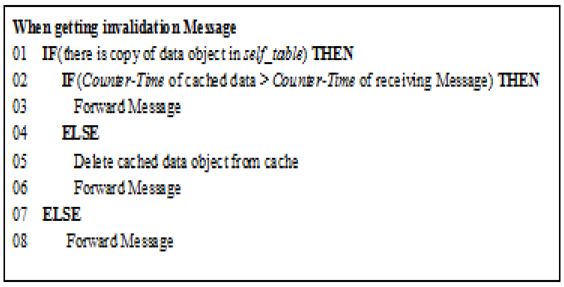
Fig. 3 Process of Receiving Data Request

Requester starts to execute data discovery process if it can't find cached record for desired data object in node table and group table. First of all requester construct routing path to destination and send data request to next mobile node in order to reach the data source which is also called as destination. Here, when intermediate nodes receive data request in routing path, they look up their node table and group table for the data request. Here, process of look up first node table and then group table respectively. Receiving mobile node forwards request to the next mobile node in the routing path if it can't find record for the request in its node table and group table.

If destination receives data request, it replies data object via routing path. Intermediate mobile node performs cache placement and replacement describe in placement and replacement section according to their node table and group table whenever it receives pass-by data object.

7) *Data Consistency:* Here cache data object is associated with an attribute Counter time. If mobile node updates data, keep copy of that data in its cache by updating Counter Time, by changing data source id to itself, by setting time stamp value to zero. After that it will pass invalidation message to every node in its network. If any node in network is having same cached data item it will compare value of counter time with time of its cached data item. If counter time value of its cached data item is old compare to receiving message value the node will delete that particular cached data.

8) *Mobility of Node:* If mobile node does not receive "Hello" message during pre-defined number of "Hello" cycles from its neighbour, it means that neighbour is leaving or shutdown. When mobile node leaves, group table needs to be updated and remove related records of leaving neighbour.



```
When getting invalidation Message
IF(there is copy of data object in self_table) THEN
    IF(Counter-Time of cached data > Counter-Time of receiving Message) THEN
        Forward Message
    ELSE
        Delete cached data object from cache
        Forward Message
ELSE
    Forward Message
```

Fig. 3 Process of Receiving Invalidation Message

IV. PERFORMANCE EVALUATION

In this section performance evaluation is shown. Section *A* gives simulation model. Section *B* verifies results of Simple Caching (SC), Cache Data [1], [12] and Zone Cooperative [11] and they are compared with proposed Group Caching. Only requester caches the replied data object for itself in simple caching. All the schemes use LRU as cache replacement policy. Section *C* introduces performance metrics. Results are shown by Section *D* in terms of performance evaluation.

A. The Simulation model

The simulation is performed on NS2 [9] with the help of CMU wireless extension. In this simulation, the AODV routing protocol [14] was tested as the underlying ad hoc routing algorithm. The simulation time is set to 6000 seconds. The number of mobile hosts is set to 100 in a fixed area. Here, we assume that the wireless bandwidth is 2MB/s and the radio range is 100m. There are totally 1000 data items distributed uniformly among all MHs. The number of hot data objects is set to 200 and all hot data objects are distributed uniformly among all MHs. The probability of queries for the hot data is set to 80%. The query rate of MHs is set to 0.2/second. In order to simulate the node join and leave operations, we set a join/leave rate. If the value of join/leave rate is 20, there will be ten MHs randomly joining and leaving the network every 20 seconds. If an MH joins or leaves the network, its content of cache will be cleared.

B. The Node Movement Model

We model the movement of nodes in a 1500m x 500m rectangle area. The moving pattern follows the random way point mobility model [15]. Nodes are placed randomly in the area initially. Here, each node selects a random destination and moves toward the destination with a speed selected randomly from (0 m/s, to 10 m/s). After the node reaches its destination, it pauses for a random period of time and repeats this movement pattern. The detail of other simulation parameters is shown in Table 1.

C. Performance Metrics

The evaluated performance metrics are average hop count, cache hit ratio (includes remote cache hit ratio in remote caching nodes), and average latency of data objects.
Average hop count: The number of hop counts between the source and the destination or caching nodes.
Cache hit ratio: The combined cache hit ratio in the requester and its group members.

Average latency: The time interval between the time of generating a query in the requester and the time of receiving requested data object from the data source.

TABLE I The Simulated Parameters

Simulator	Network Simulator (NS2) [9]
Simulation Time	6000 seconds
Network Size	1500m x 500 m
Mobile Host	100 nodes
Transmission range of MH	100m
Mobility Model	Random way point
Speed of Mobile host	1 ~ 10 m/s randomly
Total No. of data item set	1000 data item
Average query rate	0.2 / second
Hot data	20% of total data item set
Probability of query in hot data	80%
Data size	10 KB
Cache Size	200KB, 400KB, 600KB, 800KB, 1000KB, 1200KB, 1400KB
Compared Schemes	Cache Data [1], Zone Cooperative [2], Proposed Group Caching
Replacement Policy	LRU

D. Simulation Results

1) *Average Hop Count:* We first measure the hop counts in all schemes. Simulation is run under different cache sizes and different join/leave rates. In all schemes, when the cache size is large, the average hop count is reduced. In Group caching the average hop count is the lowest because the group caching improves the cache hit ratio and then reduce the average hop count.

2) *Cache Hit Ratio:* Figure 4 shows cache hit ratios of mobile hosts under different cache sizes and join/leave rates. The measured cache hit ratio includes cache hit that is local cache hit in the requester and cache hit in the other mobile hosts that is remote cache hit except the data source. The cache size is set to 200KB, 400KB, 600KB, 800KB, 1000KB, 1200KB and 1400KB. The data set size is set to 10KB. The pair of source and the destination nodes is randomly selected in the simulation. Generally, the cache hit ratio increase while the cache size increases. Figure 4 (a) shows Group cache has higher cache hit ratio compared to others because both mobile host and its group members can store data objects. These cache data items improves cache hit ratio. Figure 4 (b) shows the experimental results under dynamic topology. In every 20, 40, 60, 80, 100 and 120 seconds, ten mobile hosts are selected randomly for joining and leaving network. Mobile host removes all cached data objects when it leaves network. The content of mobile host’s cache is set to empty when it joins the network.
Group Caching shows the highest cache hit ratio because it utilizes all the available cache space of neighbours (group members). When an MH joins the network, its available cache space can be utilized by other MHs. Therefore, In Group Caching, the cache hit ratio is higher than other schemes. In ZoneCooperative and CacheData schemes, there is no cooperative caching protocol among MHs. So the MH can’t efficiently integrate their neighbour’s cache space.

3) *Average Latency:* Figure 5 shows the average latency under different cache sizes and different join/leave rates. We know that ZoneCooperative scheme has no cooperative protocol among the MHs. Therefore, when an MH receives a data request, it needs to send a request to its zone and waits for the response. As a result, it leads to the long latency if there is no cache record in a zone along the routing path. In Group Caching, the MH and its one-hop neighbours form a group. If a data request is received, the MH can check its self_table and group_table immediately. No communication with its neighbours is needed to know the caching status in other group members. The average latency is reduced as a result. Also due to the placement and replacement algorithms executed in the group, all the MHs in a group member can cache more data objects and then reduce the redundancy of the cached data. Therefore, the average latency is reduced compared with other schemes.

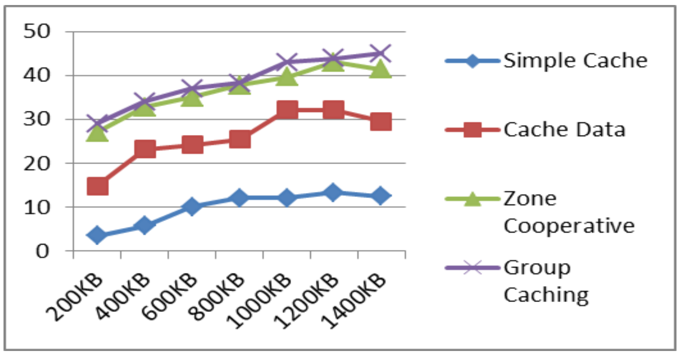


Fig. 4(a) Cache Hit ratio Vs Cache Size

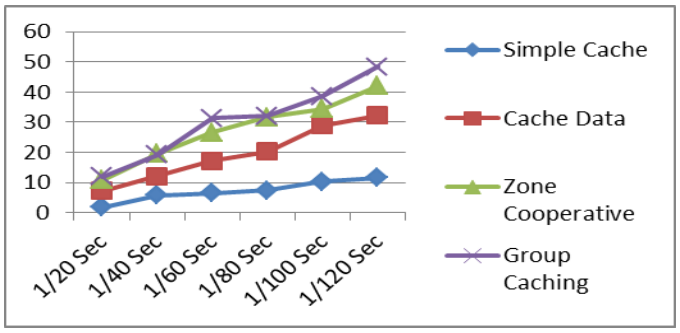


Fig. 4(b) Cache Hit ratio Vs Leave/Join Rate

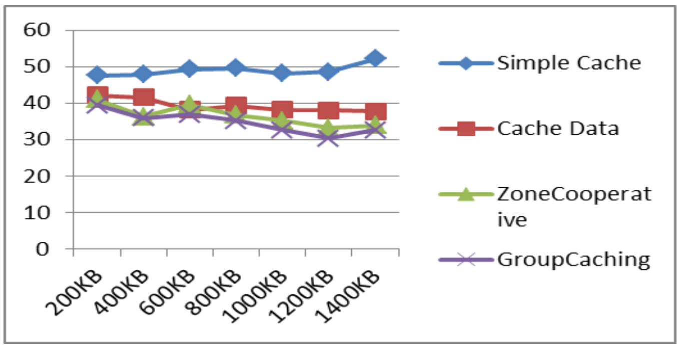


Fig. 5(a) Latency Vs Cache Size

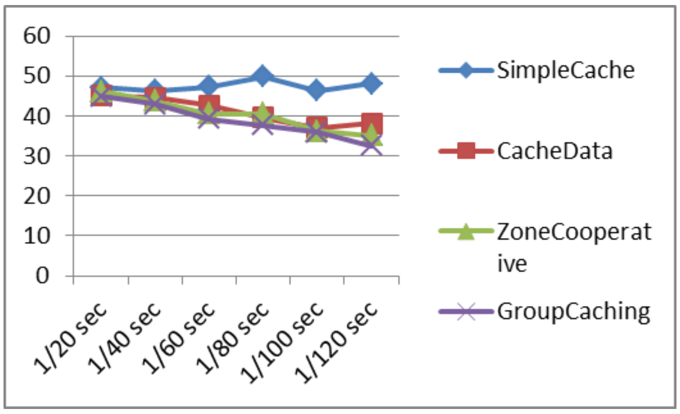


Fig. 5(b) Latency Vs Leave/Join Rate

V. CONCLUSION

In this paper, we propose a cooperative caching scheme (Group Caching) for mobile ad hoc networks. MHs maintain the localized caching status among the group members. Therefore, the MHs can cooperative to store different data objects. Furthermore, if a mobile host has available cache space, it can be utilized by its neighbours as soon as it joints the group. It improves the cache hit ratio and reduces the average latency compared with existing schemes.

REFERENCES

[1]. L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks", IEEE INFOCOM, pp. 2537-2547, March 2004.

[2]. C. Aggarwal, J. Wolf, and P. Yu, "Caching on the World Wide Web," IEEE Trans. Knowledge and Data Eng., vol. 11, no. 1, Jan./Feb. 1999.

[3]. W. Lau, M. Kumar, and S. Venkatesh, "A Cooperative Cache Architecture in Supporting Caching Multimedia Objects in MANETs", Proc. Fifth Int'l Workshop Wireless Mobile Multimedia, 2002.

[4]. J. Shim, P. Scheuermann, and R. Vingralek, "Proxy Cache Algorithms: Design, Implementation, and Performance", IEEE Trans. Knowledge and Data Eng., Vol. 11, no.4, July/Aug. 1999.

[5]. D. Wessels and K. Claffy, "ICP and the Squid Web Cache," IEEE J. Selected Areas in Comm., pp. 345-357, 1998.

[6]. G. Cao, L. Yin and C. Das, "Cooperative Cache Based Data Access Framework for Ad Hoc Networks," IEEE Computer, pp. 32-39, February 2004.

[7]. F. Sailhan and V. Issarny, "Cooperative Caching in Ad Hoc Networks", International Conference on Mobile Data Management (MDM), pp. 13-28, 2003.

[8]. C.-Y. Chow, H.V. Leong and A. Chan, "Peer-to-Peer Cooperative Caching in Moible Environments," Proceedings of the 24th Intl. Conf. on Distributed Computing Systems Workshop (ICDCSW), 2004.

[9]. K. Fall and K. Varadhan, "The NS2 manual", the VINT Project, http://www.isi.edu/nsnam/ns/. Apr. 2002.

[10]. J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," Proc. of ACM/IEEE Mobicom'98, Dallas, TX, 85-97.

[11]. Chand, N. Joshi, R. C., and Misra, M., "Efficient Cooperative Caching in Ad Hoc Networks Communication System Software and Middleware," 2006. Comsware 2006. First International Conference on 08-12 Jan. 2006 Page(s): 1-8.

[12]. LiangZhong Yin; Guohong Cao, "Supporting cooperative caching in ad hoc networks," IEEE Transactions on Mobile Computing, Volume 5, Issue 1, Jan. 2006 Page(s):77-89.

[13]. Joonho Cho, Seungtaek Oh, Jaemyoung Kim, Hyeong Ho Lee, Joonwon Lee, "Neighbor caching in multi-hop wireless ad hoc networks," IEEE Communications Letters, Volume 7, Issue 11, Nov. 2003 Page(s):525 – 527.

[14]. C. Perkins, E. Belding-Royer, and I. Chakeres, "Ad Hoc On Demand Distance Vector (AODV) Routing," IETF Internet draft, draft-perkins-manet-aodvbis-00.txt, Oct. 2003.

[15]. T. Camp, J. Boleng, and V. Davies, 2002. "A Survey of Mobility Models for Ad Hoc Network Research". Wireless Comm. & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, 2 (5): 483-502.