Efficient Cache Sharing Protocol for Mobile Nodes

Arathi.R.R.B.Tech.,(M.E)
Dept. of Computer Science & Engineering
Jerusalem Engineering College
Anna University of Technology, Chennai

Lilly Sheeba.S.(Ph.D)
Dept. of Information Technology
Jerusalem Engineering College
Anna University of Technology, Chennai

Yogesh.P. M.E.,Ph.D
Dept. of Information Science & Technology
Anna University, Chennai

ABSTRACT
Mobile Ad hoc Network provide an attractive solution for networking in the situations where network infrastructure or service subscription is not available. Its usage can further be extended by enabling communications with external networks such as Internet or cellular networks through gateways. However, data access applications in MANETs suffer from dynamic network connections and restricted energy supplies. Caching helps MANETs, to reduce average latency and wireless bandwidth, and also in alleviating from the situation of heavy traffic near the data centre. With data being cached by mobile nodes, a request to the data centre can easily be serviced by a nearby mobile node instead of the data center alone. In this paper we have proposed a Time Based Approach which provides access to recent data on demand basis. In this system, the data comes along with a time stamp. This approach provides data availability even with limited resources. We have evaluated the performance of our strategy using simulation and compared with existing non cooperative caching scheme.

Keywords
MANETs, caching, cache sharing, data retrieval, cache replacement.

1. INTRODUCTION
Extensive growth in mobile and wireless communication has led to the development of Mobile Ad hoc networks (MANETs) which is an infrastructure less network. MANETs permit mobile nodes to form a dynamic and temporary communication network without using any pre-existing infrastructure. The flexibility and ease of deployment of MANET found it very useful in many application areas like battlefield, disaster recovery, etc. However in MANETs, major issues like routing, security and data availability remain as open problems for research. Data availability is a vital issue since the ultimate goal of using MANETs is to provide information access to mobile hosts. MANETs can be extended by connecting with some other wired or wireless network like the Internet. An attractive technique that improves data availability is caching. Generally speaking, caching is to copy a portion of the data from the data provider to a smaller and faster storage device known as cache, so that future data accesses can be resolved from the cache with less cost. One thing to point out is that caching is a logical entity instead of a physical entity.
The important goal of ad hoc networks is to provide mobile nodes with easy access to information. However, MANETs are limited by intermittent network connections, restricted power supplies, and limited computing resources. These restrictions raise several new challenges for data access applications with the respects of data availability and access efficiency. In ad hoc networks, due to frequent network partition, data availability is lower than that in traditional wired networks. Cooperative caching provides an attractive solution for this problem. Cooperative caching is a technique that allows the sharing and coordination among the mobile nodes. However, the movement of nodes, limited storage space and frequent disconnections limit the availability. By the caching of frequently accessed data in ad hoc networks we can improve data accessibility, performance and availability. Due to mobility and resource constraints of ad hoc networks, caching techniques designed for wired network may not be applicable to ad hoc networks. In ad hoc network, most of the exchanged data in any application domain whether military or sporting is time specific or time sensitive, after which the data becomes an invalid one. It can be either marked for deletion or deleted from memory after the speculated time.

An example scenario is during International Sporting events like Olympic Games, the demand from users to access the Internet to get related information increases. This accessed information can then be shared with other users of same interest if they are in the vicinity of this ad hoc domain. However the accessed information can be considered valid only for a short period of time, after which the medal tally might have changed. Hence any information that is accessed can be made to be relayed along with time related information.

In general, a good cache management technique for MANETs should address these issues:
A cache discovery algorithm that is efficient to discover requested data items from the neighbour node and decide on caching the data items for future use.
There should be a cache replacement algorithm to replace the cached data items when the caching space is not enough to cache the new ones.
A cache update algorithm to ensure that the cached data items are updated.

In this paper we consider all these issues and proposed a new cache protocol based on time specification. Hence any information that is accessed can be made to be relayed along with time related information. If this information is already present then it checks if the received one is a cached information and if it is latest , makes an update. Moreover since all cached information are time specific, the information can be automatically deleted from the cache after the speculated time interval. This time variant may either be proposed by the data server or by the intermediate node that is responding to the particular information access request.

In this work, we administer three policies to enhance the cache performance in a mobile environment. The three policies are Data Item Admission policy, Data Item Discovery policy and a Data Item Replacement Policy. These three policies effectively respond to time specific information.

The rest of this paper is organized as follows: We review the related work in section 2. Section 3 describes about the system model. In section 4 we describe our proposed caching scheme. The performance of proposed protocol is evaluated in section 5 with a simulation study. Section 6 will conclude the paper.

2. RELATED WORKS

Caching is a key technique for improving data retrieval rate in both wired and wireless networks. The two basic types of cache sharing are push approach and pull approach. In push based cache sharing, a node broadcasts the caching update to all its neighbour nodes, on receiving a new data item. This updated information resides in the neighbouring nodes for future use. Push based scheme improves the data availability at the cost of communication overhead. The disadvantage of the scheme is that an advertisement may become useless if no demand for the cached items occur in the vicinity. One more problem with the push based approach is that the caching information may not be used if the node moves out from the zone or due to cache replacement. These drawbacks are overcome with the pull based scheme. In case of pull based approach, a node broadcasts a request packet to all its neighbours, when it wants to access a new data item. If a neighbour has the requested data item it sends the data back to the requester node. The main disadvantage here is that, if the requested data item is not cached by any node in the neighbourhood then the request originator must wait for the time out interval to expire before it resends the request to the data centre. This leads to access latency. Another drawback here is, if more than one node have cached the requested data item then multiple copies will return to the requester which in turn will result in extra communication.

Duane Wessels and Kim Claffy[4], introduced the standardized and widely used Internet cache protocol(ICP). As a message-based protocol, ICP supports communication between caching proxies using a simple query-response dialog.

Cache Digests [1] are a response to the problems of latency and congestion. Cache Digests support peering between cache servers without a request-response exchange taking place. A summary of the contents of the server (the Digest) is fetched by other servers which peer with it. Using Cache Digests it is possible to determine with a relatively high degree of accuracy whether a given URL is cached by a particular server. This is done by feeding the URL and the HTTP method by which it is being requested into a hash function which returns a list of bits to test against in the Cache Digest.

In [8], Web Proxy Caching is considered as one of the most important technique for reducing web traffic, which accounts for a large percentage of internet traffic today using Zips law which gives the relative probability of a request for popular page i, is 1/i.

[5] proposes various cooperative caching schemes in mobile ad hoc networks, while in the past these schemes were exclusively proposed for wired networks in a highly static environment. The performance of the dynamic environment highly depends on the mobility of the nodes and frequent disconnections of the node from the network.

Chand [9], proposed a Cooperative Cache Management strategy which allows sharing and coordination of cached data among clients to minimize data access latency and to improve information availability. In this paper a utility based cache replacement policy is adopted, to reduce the local cache miss ratio. Here the least recently used data items having the highest probability of replacement. The main disadvantage with this approach is that, not all data can be replaced based on least utility, since information that are accessed in various applications can have varying time specifications. For instance in a shopping mall application the stock related information can be cached and retained within a node for some more time when compared to information that are being cached in a military and emergency application. In case of military and emergency related applications frequently updating the cached data, is highly inevitable because the accessed data must be only recent information. Another disadvantage cited here is that only some clients retain state information within a zone.

According to Chow [2] mobile clients can access data items from the cache of their neighbouring peers by adopting COCA or Cooperative Caching Scheme wherein two types of mobile clients are identified namely Low Activity Mobile Clients in which data items are replicated and High Mobility Mobile Clients that make use of these replicas. This data replication scheme reduces both server workload and access miss ratio. The main disadvantage here is that it does not take into account the cache admission policy to be adopted in case of replicated data. In short, it consumes large amount of the available resources by caching the same data item in different nodes.

Build upon the COCA framework Chow [3] proposed a Group Based COCA scheme (GroCOCA) which defines a tightly coupled group as a set of peers that possess similar movement pattern and exhibit similar data affinity. In GroCOCA a centralized incremental clustering algorithm is used to discover all groups dynamically and the mobile hosts in same group manage their cached data items cooperatively. This scheme reduces access latency and server request ratio effectively.

[3] and [6], proposes COOP, a novel cooperative caching scheme for data access applications in MANETs. The objective is to improve data availability and access efficiency by collaborating local resources of mobile devices. COOP addresses two basic problems of cooperative caching: cache resolution and cache management. It finds the requested data efficiently and manages local cache to improve the capacity of cooperated caches. This scheme significantly reduces response delay and improves data availability for data access applications.

An aggregation caching mechanism was proposed by Lim [10] for improving the data accessibility and reducing average access latency. To retrieve data as quickly as possible, the query is issued and broadcasted to all the nodes in the network which in turn send acknowledgements individually to the source of broadcast. The requesting node will then send the request for the data to the node from which it has received the first acknowledgement. This scheme is inefficient in terms of bandwidth usage because of the broadcasts which will more

likely decrease the throughput of the system due to intensive flooding of the request packets.

In [11], Cache discovery problem is given key focus. It proposes a self-resolver paradigm, in which a client user itself queries and measures which node it should access. In addition to the self-resolver cache discovery framework, stability of a multihop route is considered.

Two caching schemes CacheData and CachePath was proposed in [6]. In CacheData scheme the intermediate nodes cache a data item to serve future requests, while forwarding the data to another requester node. With CachePath scheme, the intermediate nodes cache only the information of the path to the request originator and uses this information to redirect future requests to the nearby nodes with cached data. A Hybrid Cache scheme is also proposed here to overcome the bottlenecks of the above schemes. In Hybrid Cache mechanism, when a mobile node forwards a data item, it caches the data or the path based on some criteria like size of the data item and time to live of the data. The main drawback with these schemes is that the cached information in a node cannot he shared if the node does not lie on the forwarding path of a request to the data centre.

Chiu [7] proposed two protocols IXP and DPIP. In IXP (Factor Push) which is a push based scheme, each node shares its cache contents with all the nodes in its zone. A node always makes its cache contents known to all nodes within its zone by broadcasting factor packets. DPIP (Data Pull/Factor Push) is a pull based protocol by exploiting in-zone request broadcasts. The disadvantage with this work is that it is based on Count Vector Cache replacement policy. According to this policy, the data item with highest count or the data which has been accessed and retained in many nodes is the one first marked for replacement. Hence any data item with count vector value equal to zero will never be replaced. The vital issue here is the unnecessary usage of available cache space.

3. SYSTEM MODEL

Let us consider a mobile ad hoc network shown in Fig. 1. This network has no fixed infrastructure and node are free to move anywhere in the network. Since nodes are mobile so the topology is dynamic and temporary. In this topology N1,N2…N12 are mobile nodes. There exists a data center, N1 , that contains the database of n data items x1, x2,..., xn. This data server may be connected to some external wired or wireless network like Internet. When a node requires some data item it sends request to data server. When a node receives a data item it caches the data item locally for future use.

In any mobile node, the resources that might be available might be limited. Because of this constraint only some data items can be accommodated within the mobile cache. The system administers three policies namely Data Item Discovery, Data Item Admission and Data Item Replacement to overcome this limitation. Here since each information is associated with a time factor, the data items are admitted, updated or replaced purely based on this. Moreover if the local cache of a node is full then the data items are diverted towards the neighbours who have enough space.

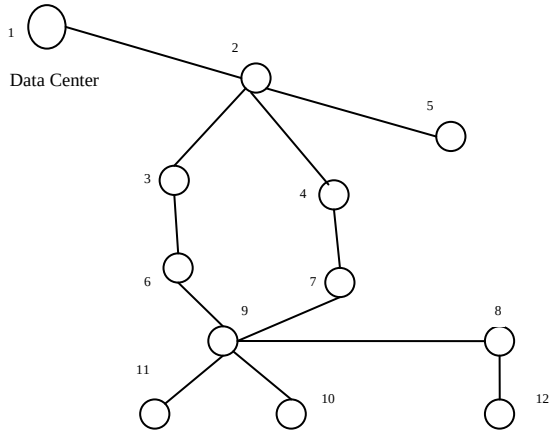

Fig 1: System Model

Here each node maintains a Data Item Table and each data in the table corresponds to three entries. For instance, the data entries for data x1 are as follows. The first entry is x1.present which is a boolean value. It is TRUE, if the data is present in the local cache of the node and FALSE if otherwise. The next entry is x1.neighbour which indicates the neighbour node that has cached the data x1. The third entry for the data is x1.time_factor which is a time attribute whose value gives the time period up to which the data can be retained in the cache. This value is determined initially by the data server and is delivered along with the data on request.

4. PROPOSED ALGORITHM

The idea of our proposed algorithm is based upon the fact that each node in the network is willing to share its cache contents with its neighbours. When a node updates its local cache it broadcasts these updates to all neighbour node in the zone. Each node in the zone will maintain a Data Item Table.

4.1 Data Item Discovery

When a data item x1 is requested by a node R, first the node will check whether x1.present is TRUE or FALSE to see the data is locally available or not. If it is TRUE, then it immediately displays the data item. If it is FALSE, then the node checks for the corresponding entry in x1.neighbor. If matching entry is found and a neighbour node has the data item, then the request is forwarded to that node. On the other hand, if no matching entries are found, then the request is directed towards the data server itself.

If any matching entry corresponding to the data item is found in any intermediate node on the way to the data server, the node immediately responds to the request instead of forwarding the request. The flowchart for data item discovery is shown in Fig. 2.

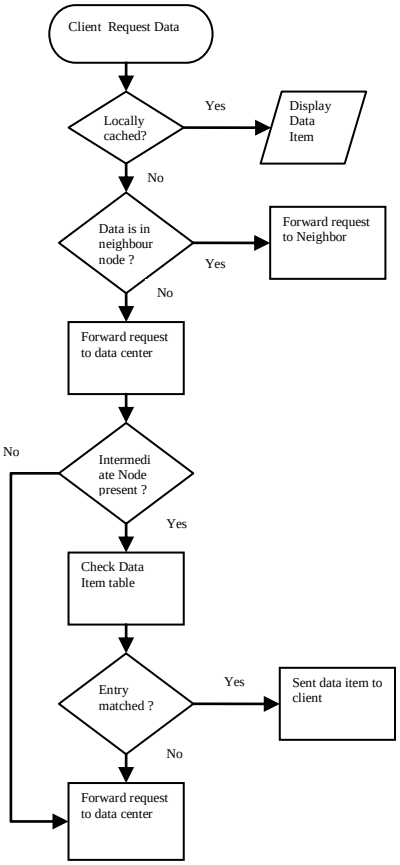

Fig 2: Data Item Discovery

4.2 Data Item Admission

Node R that requested the data item x1 receives it along with the time factor t1 . Initially, it is checked if the received data item is a new one or just an update of an existing data item by comparing the time factor of the received data item with the corresponding entry for the data item in the data item table. It then checks its cache to find any space availability, to accommodate the data item if it is a new one, in its cache. In case of any unavailability it checks for space availability in its neighbouring nodes, and if present, the data item is forwarded to the neighbour and corresponding entries are updated. The update information is then broadcast along with time factor to all its neighbours and corresponding entries in the neighbour nodes are updated. Upon receiving the data item update packet the node compares the received time factor value with that of the system time. If the received factor indicates a recent data then corresponding changes are made in the data item table. The flowchart for data item admission is shown in Fig. 3.

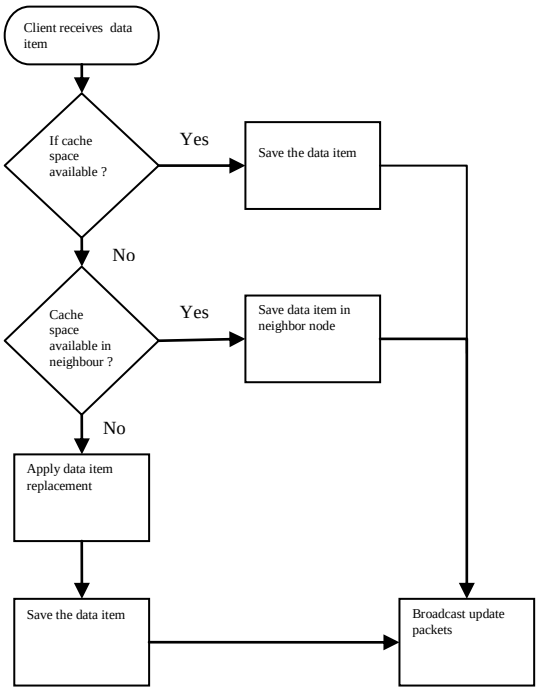

Fig 3: Data Item Admission

4.3 Data Item Replacement

When a node is powered on, it checks through the data items in its cache table. In case any data item is present in the nodes cache, with its x1.timefactor lesser than the current system time then all the entries corresponding to that data item will be either marked for deletion or automatically deleted from the cache. This is an automatic replacement policy. The flowchart for automatic data item replacement is shown in Fig .4.

If cache space is not available, a forced data item replacement is done. Here sorting is applied to the time factors of the data items in cache table .Then the data item with least time factor value is removed. The flowchart for forced data item replacement is shown in Fig. 5.

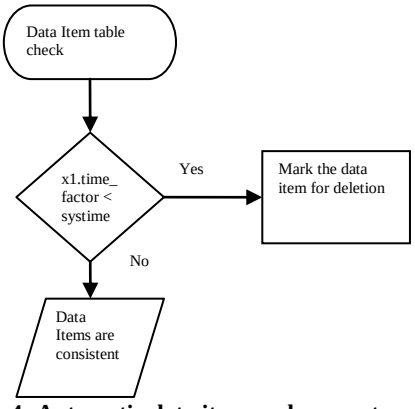

Fig 4: Automatic data item replacement

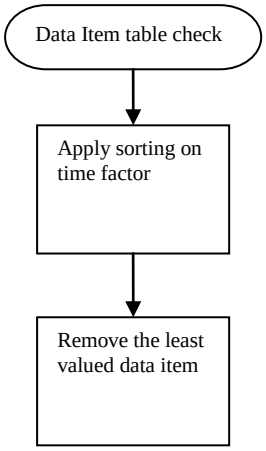

Fig 5: Forced data item replacement

5. PERFORMANCE EVALUATION

We evaluate the performance of proposed caching scheme through simulation experiments.

5.1 Simulation Model

The simulation model is constructed on the basis of the ns2 simulator . In our simulation setting, a group of nodes spread randomly in an area of 1500 m x1500 m. The number of mobile nodes varies from 50 to 100 with the default number of nodes being 70. One node is designated as the data center, and it is located at the upper left-hand corner of the area throughout the simulation. A node moves according to the random waypoint model. After the client reaches its destination it pauses for a period and repeats this movement pattern. There are N data items at the data server. Data requests are served on an FCFS basis at each node. Unless otherwise specified, the size of each data item is 1,000 bytes; other packets, such as request packets and update packets, are assumed to be 20 bytes long. The default data access pattern is uniformly distributed. When the server sends a data item to a client, it sends a time specification value along with the data.

5.2 Simulation Results

For performance comparison with proposed scheme (PS), one other scheme non-cooperative (NC) caching is also implemented. In NC, locally missed data items are fetched from the origin server. This strategy is taken to be baseline case against which the proposed caching schemes are compared. Performance metric: access latency is evaluated.

Fig. 6. shows the performance evaluation graph. The system workload becomes higher with increasing number of nodes in the system, so that access latency increases in NC caching scheme. On the other hand, performance of proposed scheme (PS) improves slightly as the number of nodes increases because there is a higher chance for the nodes to obtain the required data items from their neighbouring nodes. The access latency is defined as a sum of the transmission time and the time spent on waiting for a required communication channel, if it is busy.

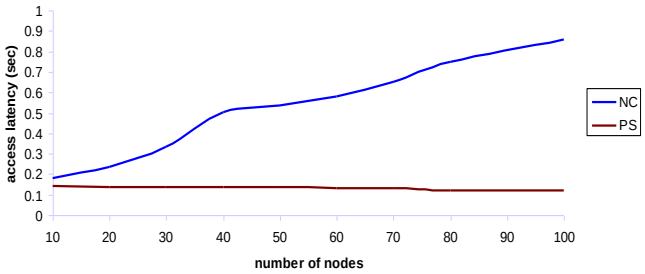

Fig 6: Access latency versus number of nodes

6. CONCLUSION

In this work, a highly reactive Cache Sharing Algorithm is proposed, to effectively and efficiently utilize the space available in the node and all its neighbours. The node and its neighbours cooperate, in caching a data item plus stops from any duplicate entries made for the same data item within a zone. Moreover since time factors are involved, even if a node moves out due to network partition, the corresponding entries will be deleted from its cache at the speculated time, thereby providing for space availability and eliminating the case of looping problem.

Additional care is taken to maintain only updated data items, taking data consistency as a vital factor. Since timing parameters are taken into consideration, if the cached data is not recent and is an outdated one, then an automatic replacement mechanism is encountered to prevent unnecessary space utilization.

All these, make this algorithm a unique one, in enhancing the performance of the cache system in a MANET environment, where node mobility and limited resources are the key issues, by providing for enhanced data availability features even with constrained resources.

Our future work includes more extensive performance evaluation. Some cache conscious techniques can be employed to provide for varying cache sizes and node density, which need not be constant always.

7. ACKNOWLEDGEMENTS

I must thank, first and foremost my internal guide Mrs.S.Lilly Sheeba, Senior Lecturer, Department of Information Technology, and project coordinator Dr.C.R.Rene Robin, Head of the department, Department of Computer Science and Engineering, without whose guidance and patience, this dissertation would not be possible. And engineering, project panel members, Professors of the Department of Computer Science and Engineering for their consistent encouragement and ideas.

8. REFERENCES

[1] A. Rousskov and D. Wessels (1998), “Cache Digests,” Computer Networks and ISDN Systems, vol. 30, nos 22-23, pp. 2155-2168.

[2] C.Y.Chow, H.V. Leong, and A. Chan (2004), “Peer-to-Peer Cooperative Caching in Mobile Environments,” Proc. 24th Int’l Conf. Distributed Computing Systems Workshops (ICDCSW ’04), pp. 528-533.

[3] C.Y.Chow, H.V. Leong, and A.T.S. Chan (2004), "Group-Based Cooperative Cache Management for Mobile Clients in Mobile Environments," Proc. 33rd Int'l Conf. Parallel Processing (ICPP '04), pp. 83-90.

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

[5] F. Sailhan and V. Issarny (2003), "Cooperative caching in ad hoc networks ", Proc. MDM'03, pp.13-28.

[6] G. Cao, L. Yin, and C.R. Das (2004), "Cooperative Cache-Based Data Access in Ad Hoc Networks," Computer,vol. 37, no. 2, pp. 32-39.

[7] Ge-Ming Chiu and Cheng-Ru Young (2009), "Exploiting In-Zone Broadcasts for Cache Sharing in Mobile Ad Hoc Networks IEEE Trans. Mobile Computing, vol. 8, no.3.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker (1999), "Web Caching and Zipf-Like Distribution: Evidence and Implication,"Proc. IEEE INFOCOM '99, pp. 126-134.

[9] N. Chand, R.C. Joshi, and M. Misra (2007), "Cooperative Caching in Mobile Ad Hoc Networks Based on Data Utility," Mobile Information System, vol. 3, no. 1, pp. 19-37.

[10] S. Lim, W. Lee, G. Cao, and C.R. Das (2006), "A Novel Caching Scheme for Improving Internet-Based Mobile Ad Hoc Networks Performance," Elsevier J. Ad Hoc Networks, vol. 4, no. 2, pp. 225-239.

[11] T. Moriya and H. Aida (2003), "Cache Data Access System in Ad Hoc Networks," Proc. Vehicular Technology Conf. (VTC '03), vol. 2, pp. 1228-1232.

[12] T.Hara (2002), "Cooperative caching by mobile clients in push based information systems", Proc. CIKM'02, pp.186-193.

[13] NS Notes and Documentation, http://www.isi.edu/nsnam/ns/, 2008.

[14] Y. Du and S. Gupta (2005),"COOP – A Cooperative Caching Service in MANETs", Proceedings of the IEEE ICAS/ICNS , pp.58–63

[15] Yu Du, Sandeep K.S. Gupta and Georgios Varsamopoulos (2009), "Improving on-demand data access efficiency in MANETs with cooperative caching, AdHoc Networks", 7 (3), pp.579-598.