A TIME INDEX BASED APPROACH FOR CACHE SHARING IN MOBILE ADHOC NETWORKS

Lilly Sheeba S[1] and Yogesh P[2]

[1]Department of Computer Science and Engineering, Jerusalem College of Engineering, Anna University,
Chennai, India
slilly_sheeba@yahoo.com
[2]Department of Information Science and Technology, Anna University,
Chennai, India
yogesh@annauniv.edu

ABSTRACT

Initially wireless networks were fully infrastructure based and hence imposed the necessity to install base station. Base station leads to single point of failure and causes scalability problems. With the advent of mobile adhoc networks, these problems are mitigated, by allowing certain mobile nodes to form a dynamic and temporary communication network without any pre-existing infrastructure. Caching is an important technique to enhance the performance in any network. Particularly, in MANETs, it is important to cache frequently accessed data not only to reduce average latency and wireless bandwidth but also to avoid 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 centre alone. In this paper we propose a system , Time Index Based Approach that focuses on providing recent data on demand basis. In this system, the data comes along with a time stamp. In our work we propose three policies namely Item Discovery, Item Admission and Item Replacement, to provide data availability even with limited resources. Data consistency is ensured here since if the mobile client receives the same data item with an updated time, the previous content along with time is replaced to provide only recent data. Data availability is promised by mobile nodes, instead of the data server. We enhance the space availability in a node by deploying automated replacement policy.

KEYWORDS

mobile ad hoc network, caching, cache sharing, cache replacement

1. INTRODUCTION

Explosive growth in mobile and wireless communication has led to the development of Mobile Adhoc 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 domains 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.

D.C. Wyld, et al. (Eds): CCSEA 2011,CS & IT 02, pp. 01–08, 2011.
 DOI : 10.5121/csit.2011.1201

An attractive technique that improves data availability is caching. MANETs permit the nodes to roam freely in all possible directions. However MANETs are constrained by limited energy,

computation power and bandwidth which make cache management a challenge. Caching takes advantage of the multi hop wireless communications that exist among mobile nodes. As there is no infrastructure support mobile nodes co operate with each other to forward data. Moreover, caching provides sharing data among mobile nodes in spite of frequent network partitioning. By caching frequently accessed data, data availability can be enhanced. 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.

Scenario1: In case of a military application one mobile node may be connected to the Internet by a satellite and this serves as a proxy to other mobile terminals. Any accessed information in case of a war can be shared by other mobile terminals via local adhoc communication [3]. Even in this case the accessed or cached information is time sensitive since it relates to the current scenario of a military mission. Hence the cached information may not serve to be useful for long durations because by then the scenario might have changed.

Scenario 2: 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 adhoc 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 informations. If this time related information is present then even if any recent update for the already cached information crosses the node it can possibly update that particular item effectively. 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 is either 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 are administered to enhance the cache performance in a mobile environment. The three policies are Cache Admission policy, Cache Update policy and a Cache Replacement Policy. These three policies effectively respond to time specific information.

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 overhead.

Duane Wessels and Kim Claffy[17], 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 [10] 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 [6], 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.

[19, 20, 21] proposes various cooperative caching schemes in mobile adhoc 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 et al. [1], 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 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 informations 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 informations 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 informations. Another disadvantage cited here is that only some clients retain state information within a zone.

According to Chow et al. [2], [11] 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 et al.[12] 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.

Du et al. in [15] and [16], 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 et al. [3] 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 [13], 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 [4], [14],[18]. 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 item. 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 et al. [5] proposed two protocols IXP and DPIP. In IXP (Index 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 index packets. DPIP (Data Pull/Index 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 item 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. PROPOSED SYSTEM

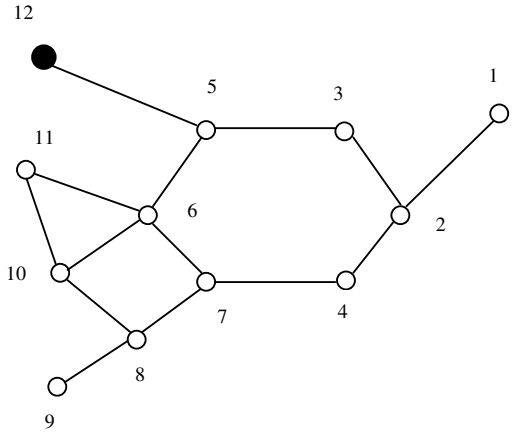

Figure 1. A simple mobile adhoc network

Figure 1. shows part of a mobile adhoc network. Since all the nodes are mobile it leads to a temporary, dynamic topology always. In this topology N_1, N_2, ….., N_{12} mobile nodes. N_{12} is the

data centre having a database of items i1, i2, …., im . It may be a node connected to a wired network.

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

Here each node maintains an Item Table and each item in the table corresponds to three entries. For instance, the item entries for item i1 are as follows. The first entry is *i1.present* which is a boolean value. It is TRUE, if the item is present in the local cache of the node and FALSE if otherwise. The next entry is *i1.neighbour* which indicates the neighbour node that has cached the item i1. The third entry for the item is *i1.time_index* which is a time attribute whose value gives the time period up to which the item can be retained in the cache. This value is determined initially by the data server and is delivered along with the item on request.

3.1. Item Discovery

A data item i1 is requested by a node R.

1) R checks its local cache:

```
If (i1. present = = TRUE)
        Display i1;
Else
        If (i1.neighbor! = NULL)
                Forwards the request to i1.neighbor;
        Else
                Forwards the request to the data centre;
```

2) Intermediate Node receives the request from R:

```
If (i1. Present == TRUE)
        Sends i1 to R;
Else
        If (i1.neighbor! = NULL)
                Forwards the request to i1.neighbour;
        Else
                Forwards the request to the data centre;
```

Node R first checks for the item i1 in its local cache. If *i1.present* is TRUE, then it immediately displays the item. If it is FALSE, then the node checks for the corresponding entry in *i1. neighbor*. If matching entry is found and a neighbour node has the 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 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 towards the data server.

3.1. Item Admission

Node R decides if it can cache the data with respect to space availability.

1)R receives the requested data item i1 with the time index t1 which is represented as <i1,t1>:

```
If (cache space is available)
        If ((i1.present == TRUE) && (t1> i1.time _ index))
        {
                Update i1;
                i1.time _ index = t1;
        }
        Else
        {
                i1.present = TRUE;
                i1.neighbor = NULL;
                i1.time_index = t1;
        }
Else
        For each neighbour
        {
                If (i1.present == TRUE && t1>i1.time_index)
                {
                        Update neighbour.i1;
                        neighbour.i1.time_index = t1;
                        i1.time_index = t1;
                        i1.neighbor = neighbour;
                         i1.present = FALSE;
                }
                Else if (cache space is available)
                {
                        neighbour.i1.present = TRUE;
                        neighbour.i1.neighbor = TRUE;
                        neighbour.i1.time_index = t1;
                        i1.present=FALSE;
                        i1.neighbor = neighbour;
                        i1.time _index = t1;
                }
        }
        Broadcast update packet <R, i1, i1.time_index>;
```

2) Intermediate Node on receiving the Update packet <R1, i1, t1>:

```
If (t1> systime)
{
        i1.time_index = t1;
        i1.present = FALSE;
        i1.neighbour = R;
}
```

Node R that requested the information receives it along with the time index. Initially, it is checked if the received item is a new one or just an update of an existing item by comparing the time index of the received item with the corresponding entry for the item in the item table. It then checks its cache to find any space availability, to accommodate the 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 item is forwarded to the neighbour and corresponding entries are updated. The

update information is then broadcast along with time index to all its neighbours and corresponding entries in the neighbour nodes are updated.

Upon receiving the Item Update packet the nodes compare the received time index value with that of the system time. If the received index indicates a recent item then corresponding changes are made in the item table.

3.3. Item Replacement

```
If (i1.time_index== systime)
{
        Removes the item from Cache;
}
```

In case any item is present in the nodes cache, with its *i1.timeindex* lesser than the current system time then all the entries corresponding to that item will be either marked for deletion or automatically deleted from the cache.

4. CONCLUSIONS

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 an item plus stops from any duplicate entries made for the same item within a zone.

Moreover since time indexes 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.
Additional care is taken to maintain only updated 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 utilisation.

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 is targeted towards developing a suitable data structure that aims at reducing the space required to save a single element in the cache memory. Some cache conscious techniques can be employed to provide for varying cache sizes, which need not be constant always.

REFERENCES

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

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

[3] S. Lim, W. Lee, G. Cao, and C.R. Das, "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, 2006.

[4] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," IEEE Trans. Mobile Computing, vol. 5, no. 1, pp. 77-89, Jan. 2006.

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

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

[7] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," Proc. ACM MobiCom '98, pp. 85-97, Oct. 1998.

[8] A.Silberschatz, P.B.Galvin, and G.Gagne, Operating System Concepts. John Wiley and Sons, 2004.

[9] C.E.Perkins and E.M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," Proc. Second IEEE Workshop on Mobile [1] N. Chand, R.C. Joshi, and M. Misra, "Cooperative Caching in Mobile Ad Hoc Networks Based on Data Utility," Mobile Information System, vol. 3, no. 1, pp. 19-37, 2007.

[10] A. Rousskov and D. Wessels, "Cache Digests," Computer Networks and ISDN Systems, vol. 30, nos. 22-23, pp. 2155-2168, 1998.

[11] C.-Y. Chow, H.V. Leong, and A. Chan, "Cache Signatures for Peer-to-Peer Cooperative Caching in Mobile Environments," Proc. 18th Int'l Conf. Advanced Information Networking and Applications (AINA '04), pp. 96-101, 2004.

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

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

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

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

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

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

[18] L.Yin and G.Cao, "Supporting Cooperative Caching in Ad Hoc Networks", Proc. IEEE INFOCOM '04, pp. 2537-2547, 2004.

[19] T. Hara, "Effective replica allocation in adhoc networks for improving data accessibility", Proc. INFOCOM '01, pp.1568-1576, April 2001

[20] F. Sailhan and V. Issarny, "Cooperative caching in adhoc networks ", Proc. MDM'03, pp.13-28, Jan. 2003

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

[22] G. Zipf, Human Behavior and the Principle of Least Effort. Addison Wesley, 1949.

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