A Key Based Cache Replacement Policy for Cooperative Caching in Mobile Ad hoc Networks

Preetha Theresa Joy
Dept. of Computer Science
Cochin University of Science and Technology
Kochi, Kerala India.

K. Poulose Jacob
Dept. of Computer Science
Cochin University of Science and Technology
Kochi, Kerala, India.

***Abstract*—Cooperative caching in mobile ad hoc networks aims at improving the efficiency of information access by reducing access latency and bandwidth usage. Cache replacement policy plays a vital role in improving the performance of a cache in a mobile node since it has limited memory. In this paper we propose a new key based cache replacement policy called E-LRU for cooperative caching in ad hoc networks. The proposed scheme for replacement considers the time interval between the recent references, size and consistency as key factors for replacement. Simulation study shows that the proposed replacement policy can significantly improve the cache performance in terms of cache hit ratio and query delay.**

Keywords—Data Caching; Cooperative Cache; Ad hoc Networks; Cache Replacement.

I. INTRODUCTION

The advancement in the new wireless technology has lead to the wide popularity of Mobile Ad hoc Networks (MANETS). The primary attraction of a wireless ad hoc network is the fact that these networks can be formed spontaneously without any fixed infrastructure. This makes the ad hoc networks ideal for applications where initial connection setup is not possible or unreliable like military and disastrous areas. It has also wide range of commercial applications like personal area networks, sensor networks, emergency networks and vehicular communication. In these types of networks, devices generally have limited energy reserves and processing capabilities. Bandwidth is also a scarce resource, limited by the nature of the wireless medium. In a data-management point of view, these restrictions introduce several issues that need to be addressed. Data transfers must be reduced and mechanisms must be deployed in order to confront the frequent disconnections and low bandwidth constraints. Therefore it is a challenging task to present the data efficiently by reducing the delay or waiting time to the end user.

Data caching is widely used in various domains to improve data access efficiency, by reducing the waiting time or latency experienced by the end users. In wireless mobile network, holding frequently accessed data items in a mobile node's local storage can reduce network traffic, response time and server load. Caching in ad hoc networks is effective because a few resources are requested often by many users, or repeatedly by a specific user, which is known as the locality of reference. To have the full benefits of caching, the neighbor nodes can cooperate and serve each other's misses, thus further reducing the wireless traffic. This process is called cooperative caching. Since the mobile nodes can make use of the data stored in another node's cache the effective cache size is increased. However, since the mobile nodes have limited memory, the cache area defined for storage is also limited. Whenever the cache memory is full, we have to find an efficient method to replace some data from the cache to make room for the newly arrived data. The cache replacement strategy decides which data item has to be removed to place the new data items. Cache replacement algorithm plays a central role in response time reduction by selecting suitable subset of data for caching. Numerous cache replacement algorithms were proposed in web caching but only a few were proposed for ad hoc networks.

In this paper we present a coordinated cache replacement policy, E-LRU (Extended LRU) which evicts data based on size, time interval between recent accesses and consistency. Almost all of the replacement algorithms proposed for cooperative caching are function based policies which involves the calculation of a value function based on different parameters , which uses complex data structures that makes the implementation difficult. Least Recently Used (LRU) on the other hand is the simplest replacement policy suggested from early days, for data caching. The disadvantage of LRU is, it considers only too little information for replacement. The replacement policy we propose is an extension to the LRU policy, aims at replacing the data objects based on the time difference between the recent references. The novelty of our approach lies in the key based replacement where we first consider size of the data item, then takes in to account the time interval between the recent references and the pages with longest reference interval time is dropped first. The advantage of our scheme is that the pages with shortest access time interval, which have more probability of reference in the future, will be kept in the cache. Simulations shows that our replacement policy out performs LRU in cache hit ratio and average query delay.

978-1-4673-4529-3/12/$31.00 ©2012 IEEE

The remainder of the paper is structured as follows. Section II reviews the related works in cache replacement, Section III details the system outline and assumptions, Section IV presents the cache admission and replacement policies used, Section V explains the simulation scenario, Section VI includes the analysis of results and Section VII concludes the paper.

II. RELATED WORKS

Caching is a very popular technique to enhance the performance of both wired and wireless networks. It is well studied for wired web applications. Due to the space limitation, the mobile nodes can store only a subset of the frequently accessed data. A good replacement mechanism is needed to distinguish between the items to be kept in cache and that is to be removed when the cache is full. The important factors that can influence the replacement process are access probability, recency of request for a data item, number of requests to a data item, size, cost of fetching data from server, modification time, expiration time, distance etc. Most of the replacement decision proposals are based on the above factors.

Yin and Cao [4] proposed a generalized cache replacement policy for mobile environment. The value function they proposed can be used for different performance metrics and they considered minimum query delay and minimum download traffic as the target. The value function was based on parameters like probability of reference, cost of fetching data item, cost of validation, probability of invalidating cached data item and cost of getting updated data item to the cache. Based on these parameters the algorithm replaces a data item with min Value (i) / S_i, where S_i is the size of the data item. Here a strong consistency model is assumed. Xu and Lee [5] proposed a gain based replacement policy SAIU, for on demand broadcasts. The gain function for each data item is calculated as gain (i) = L_i. A_i / S_i. U_i where L_i is data retrieval delay, A_i is the access rate, S_i is the size of the data item and U_i is the update frequency.

Another algorithm proposed by Zeitunlian and Haraty [6] uses a least unified value cache replacement for SACCS, scalable asynchronous cache constituency scheme. Here the replacement is based on the reference information of the object, fetch cost and size. They considered the complete reference history for finding the probability of reference in the future. The book keeping involved in this method is too high. Chem. and Xiao [7] presented a cache replacement policy called on bound selection which uses data access and update information for replacement decision. The above mentioned schemes use a value function to replace the data. Since the relative importance of these parameters can vary from one type of request to another, some policies are needed to adjust the weights dynamically to achieve the best performance. The above mentioned replacement policies are based on infrastructure based wireless networks. However, unlike the infrastructure based wireless communications, the deployment of these new algorithms in ad hoc networks faces several important restrictions due to the mobility of nodes. These strategies did not assume cooperative caching and multi hop communication in ad hoc networks.

The cooperative caching techniques mentioned in [8], [9] uses Least Recently Used (LRU) as the replacement policy. N. Chand [10] suggested a Least Utility Valued with Migration replacement strategy for cooperative caching .The utility value is calculated based on popularity, distance, coherency and size. Here the replaced items are stored in the neighboring nodes based on space availability. But due to the mobility of nodes in ad hoc network the topology may change frequently and the distance parameter may become obsolete. Edward Chan and Li [11] considered the impact of energy on cache replacement policy. They considered the energy cost for each data access. For this, they considered the energy for in zone communication, energy for sending the object, energy for receiving and energy cost for forwarding the object. Based on this they proposed a dynamic ECORP DP and ECOPR _greedy algorithms to replace data. However, in order to implement these schemes, one needs to maintain complicated data structures. Compared to the wide range of replacement policies suggested for infrastructure based networks, few are proposed for cooperative caching.

III. SYSTEM OUTLINE AND ASSUMPTIONS

The system environment is assumed to be a mobile ad hoc network which includes a collection of nodes n_i (i=1,2,……N) that communicate with each other in a coordinated manner and $V_i \subseteq$ {n1,n2,…….nN} such that Vi is the set of neighboring nodes for a node 'n'. Data is originated in the server which is assumed to be in a fixed position. Mobile hosts are identified as <Hostid, Name> for 1<=i<=N, where N is the density of the network which is decided by the user. Each data item is uniquely identified by means of data item id D_i for1<=i<= n where 'n' is the size of the data base. Each node n_i maintains a local cache of data items which it frequently accesses. Whenever a node receives a request for the data item, local cache is checked. If the item is found, the request is satisfied from the local cache, otherwise the request is forwarded to the neighboring nodes. The nodes within a range 'r' are identified as neighboring nodes. The process of accessing data is shown in Fig. 1. Cache consistency is maintained through a weak consistency model using the parameter TTL (Time to Live).When the TTL time expires, the cached data item is not valid and we have to fetch the updated data from the sever. Cache admission control decides whether the incoming data should be cached or not. Here we assume that the data taken from neighboring nodes are not cached. Thus we can reduce redundancy and can avail more cache space.

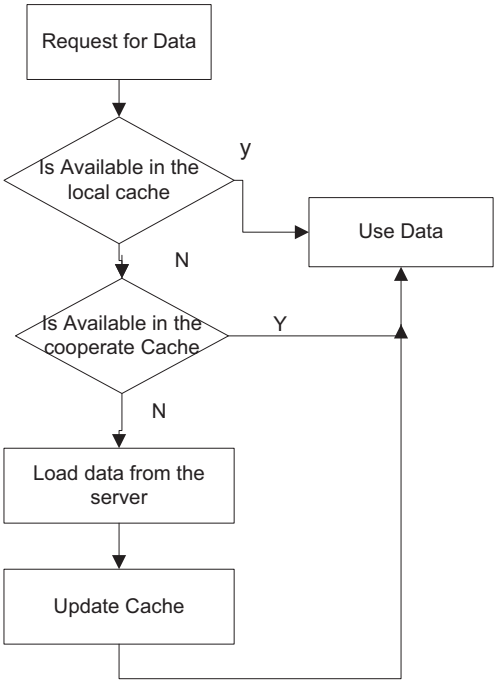

Fig.1 Data Retrieval Process

IV. CACHE ADMISSION AND REPLACEMENT

In cooperative caching nodes share the cache contents of neighboring nodes to utilize the full advantage of caching. The available cache replacement mechanisms for ad hoc network can be categorized in to coordinated and uncoordinated depending on how replacement decision is made [13]. In uncoordinated scheme the replacement decision is made by individual nodes. Coordinated cache replacement considers the information present in neighboring nodes for replacement.

In order to improve the content diversity in the cooperative cache, our scheme does not cache any data coming from the neighboring nodes. This increases the availability of information for the user, as more data items are cached and also avoids additional request to the server. Previous studies [15] have shown that the requests for smaller objects are more compared to bigger objects, so the probability of achieving high hit rate is increased if we store more number of small objects. In our proposed model we set a threshold value for the size of the data item admitted to cache. The threshold value is set as 50% of the total cache size. Any object bigger than this threshold is'nt brought in to cache.

A. Replacement Algorithm

Due to limited sized cache, nodes cannot store all the content they need, but are forced to choose some items to keep and someone to discard every time when newly retrieved data fill up their memory. In cooperative caching if data replacement decision is made by individual node by considering the local cache, the performance is degraded because the data may be present in the neighboring node. If the cache is full, appropriate data from the cache is evicted to make room for the incoming one. The algorithm illustrated here considers recency and the number of references for a particular data item and gives more significance to data items that are referenced more than once. When we have data items referenced only once, it is given more priority for replacement. At this time LRU is used for replacement. If an item is referenced more than once the inter arrival between the most recent two references is considered for eviction.

Let t_c be the current reference time and t_r be the previous reference time then T_{int}, the inter arrival time is given by (1)

$$T_{int} = t_c - t_r \quad (1)$$

If $t_c - t_r = 0$, an item whose last reference time is smaller is replaced .

If Tint > 0, then the replacement decision is made on the value of K (i) which is given as (2)

$$K(i) = \text{Max} \quad \sum_{i=1}^{n} tc - t_r \quad (2)$$

The data items with maximum inter arrival time is considered for replacement. In both cases if more than one data item have the same value, the TTL parameter is taken and the one with lower TTL value is removed as the data with lower TTL will be outdated soon.

V. SIMULATION SCENARIOS AND METRICES

We have developed a simulation model in JAVA. In our simulation, nodes are randomly placed in an area of 800X800 m^2. Each node is identified by a node id and a host name. The data server is implemented as a fixed node in the simulation area. The data server contains all the data items requested by the mobile nodes. The size of each data item is uniformly distributed between s_{min} and s_{max} .The nodes in the network move randomly based on a random path. The nodes within a transmission range of 100m are taken as the neighboring nodes. The nodes that generate data request are selected randomly and uniformly. The time interval between two consecutive queries generated from each node/client follows an exponential distribution with mean of 10sec. Each mobile node generates a single stream of read only queries. After a query is sent out, the client does not generate new query until the pending query is served. The data access pattern follows a Zipf distribution [17] with a skewness parameter θ as 0.8.

TABLE I. SIMULATION PARAMETERS

Parameter	Default Value	Range
Simulation Area	800X800 m^2	-
Database size	1000 items	-
Cache size(% of total database)	50	20-70
Nodes Density	30	20-100
Size of the data item(KB)	-	1-10

A. *Performance Metrices*

Our performance metrics include cache hit ratio and average query delay. The evaluation of these parameters are done by varying the number of cache locations with respect to number of nodes and the behavior of cache hit ratio for different cache sizes. The hit ratio is defined as the percentage of requests that can be served from previously cached data. Since the replacement algorithm decides whether to cache the data or not, it affects the cache hits of future requests. The query delay is the time interval between the query sent and the data transmitted back to the requester. Average query delay is the query delay averaged over all queries.

VI. EXPERIMENTS AND RESULTS

We compared the performance of our method with LRU for different cache sizes. Different cache sizes were used, ranging from 20% to 80% of the total size of the database. Fig 2 and Fig 3 shows the result with a set of 6 cache sizes which are 20%,30%,40%,50%,60% and70% of the total size of the database, respectively. Fig 2 depicts the comparison of cache hit ratio for different cache sizes. This figure shows that cache hit ratio of E-LRU is more for all cache sizes than LRU. At small cache sizes E-LRU shows significant improvement in cache hit ratio compared to LRU. For large cache sizes the difference in hit ratio of the two policies became less significant. Fig 3 is the comparison of average query delay for the same and it shows that average query delay is always lower for E-LRU than by LRU. This is due to the fact that E-LRU uses the cache space more effectively and the number of data requests send to the server is reduced.

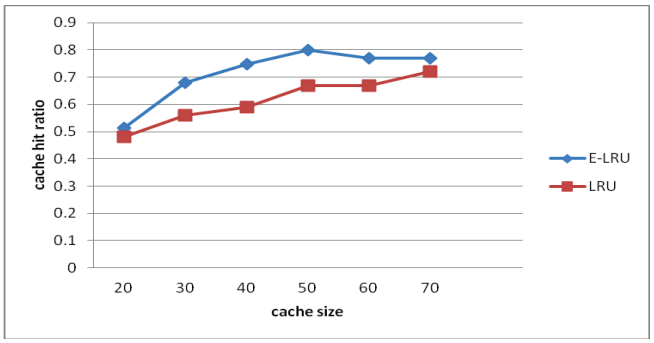

Fig.2. Cache Hit Ratio

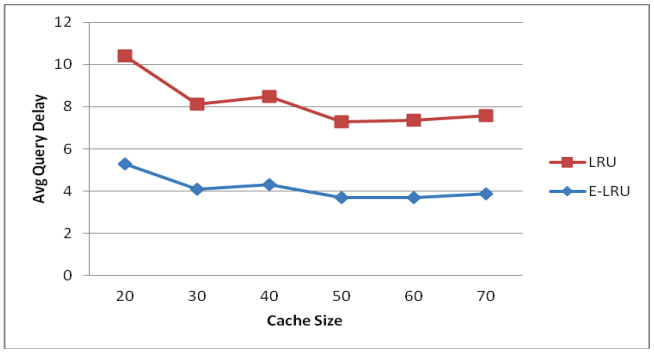

Fig.3.Average Query Delay

VII. CONCLUSION

In this paper, we have explored cache replacement issues for ad hoc networks and presented a new cache replacement policy for cooperative caching. The algorithm takes in to account the inter arrival time of recent references, size and consistency for page replacement. In LRU only the last time of reference is taken and the numbers of references are not considered. Since the inter arrival time of the recent reference is taken more preference is given to objects that have been accessed more than once. Hence we are able to distinguish between data that are frequently referenced with that of occasionally referenced. As we are not caching data with bigger size cache space can be saved. Since the algorithm is based on a key based approach it is simple to implement. Experimental results show that the proposed replacement algorithm can significantly improve the cache hit ratio and lower the data access delay when compared to LRU.

REFERENCES

[1] Dan Hirsch, Sanjay Madria A Resource- Efficient Adaptive Caching Scheme for Mobile Ad-Hoc Networks.29th IEEE International Symposium on Reliable Distributed Systems, (2010).

[2] Lekshmi M. Chidambaram, Sanjay K. Madria, Mark Linderman and Takahiro Hara, MELOC: Memory and Location Optimized Caching Model for Small Mobile Ad hoc Networks , 13th International Conference on Mobile Data Management. July 23-26, 2012, Bengaluru, India.

[3] T.Nguyen ,Thuy Dong Thi Bich, An efficient model for cooperative caching in mobile information systems, Int. Conf on Advanced Information Networking and applications,90-95,2011.

[4] Yin, L., Cao, G., and Cai, Y. A Generalized Target-Driven Cache Replacement Policy for Mobile Environments. In Proceedings of SAINT. 2003, 14-21.

[5] J.Xu,Q.L.Hu, L. Lee and W-C Lee, SAIU:" An efficient cache replacement policy for wireless on demand broadcasts, Proceedings of the 9th ACM International Conference on Information and Knowledge management (CKIM 200) pp 46-53,McLean, VA,USA, Nov 2000.

[6] Aline Zeitunlian, Ramzi A. Haraty, "An Efficient Cache Replacement Strategy for the Hybrid Cache Consistency Approach", World Academy of Science, Engineering and Technology 63, 2010.

[7] H. Chen and Y. Xiao, On-bound selection cache replacement policy for wireless data access. *IEEE Transactions on Computers*, 56 12 (2007), pp. 1597–1611.

[8] 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, 525 – 527(2003).

[9] Yi-Wei Ting,Yeim-Kuan Chang. *A novel cooperative caching scheme for wireless ad hoc networks: Groupcaching*. In NAS '07: Proceedings of the International Conference on Networking, Architecture, and Storage,(2007).

[10] Narottam Chand, R.C. Joshi and Manoj Misra, "Cooperative Caching Strategy in Mobile Ad Hoc Networks Based on Clusters," International Journal of Wireless Personal Communications special issue onCooperation in Wireless Networks, Vol. 43, Issue 1, pp. 41-63, Oct 2007

[11] Li, W., Chan, E., & Chen, D. (2007)." Energy-efficient cache replacement policies for cooperative caching in mobile ad hoc network.', In Proceedings of the IEEE WCNC (pp 3349–3354).

[12] O'Neil and P. O'Neil, "The LRU-k page replacement algorithm for database disk buffering",Proceedings of the ACM SIGMOD 1993, pp: 296-306.

[13] Preetha Theresa Joy and K. Polouse Jacob," Cache Replacement Strategies for Mobile Data Caching ",International Journal of Ad hoc, Sensor & Ubiquitous Computing, August 2012, Volume 3, Number 4.

[14] G. Cao, L. Yin, C. R. Das. "Cooperative Cache-Based Data Access in Ad Hoc Networks," Computer, vol. 37, Issue 2, Feb 2004, pp. 32-39.

[15] S. Jin, A. Bestavros, GreedyDual Web caching algorithm: exploiting the two sources of temporal locality in web request streams. Computer Communications, vol. 24, no. 2, Feb. 2001, pp. 174-83.

[16] L. Fan, P. Cao, J. Almeida, A.Z. Broder," Summary cache: a scalable wide-area Web cache sharing protocol, IEEE/ACM Transactions on Networking" , vol. 8, no. 3, June 2000, pp. 281-93

[17] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Sheker, "Web Caching and Zipf-Like Distributions: Evidence and Implications", Proceedings of IEEE INFOCOM, pp.126-134,March 1999

.