ISSN:0975-9646

Improving the Efficiency of Cache Updating Process in Wireless Mobile Networks

Jenilet J[*],.E.Nagarajan [**]

[*]*P.G Student, Sathyabama University, Chennai -600 119.*
[**]*Asst.Poffessor, Sathyabama University, Chennai-600 119.*

Abstract- To reduce network delay and bandwidth gain in the MANETs environments using cache updating process. In cache updating, to update the cache nodes using pull-based algorithm. Whereas, the pull-based algorithm, the cache ask to the data source for updating the data item into the cache. In this algorithm implements prefetching and piggybacks the data item from the data source. In this paper we proposed Decentralized Distributed Cache Updating Process, where the Query directory is fixed based on the storage capacity. If the capacity is low related to previous fixed QD, we can assign new query directory for the group of mobile nodes for updates. This was implemented using Ns2 and compared to push-based and pull-based schemes for improving the performance.

Index Items-Cache update, MANET, Prefetching, piggyback.

1 INTRODUCTION

In MANETs, the limited mobile devices are used, and the nearest mobile are grouped to form a cluster form. In this group contains different forms of mobile devices which performs several operations. Here data caching is important in this mobile environments. Some of nodes in groups act as Query directory, which stored the directory of each cache nodes in the networks. And the cache nodes, cache the queries from the mobile nodes and process its and send responses to the mobile nodes. Each cache nodes data items are similar to the data source, where it provides strong consistency between data source and cache nodes.

In[4] explains the cache consistency scheme, to maintain cache consistency of data items in a cache that is time-to-live, cache invalidation protocols, and client polling. Time-to-live are efficient to set up the value for each data item. Therefore, it is necessary to set the TTL field to a relatively short interval and reload the object frequently to avoid returning stale data. cache invalidation protocol that describes in[1], it is the process of deleting invalidate date in cache. A cache node changes a variable and then invalidates the cached values of that variable.

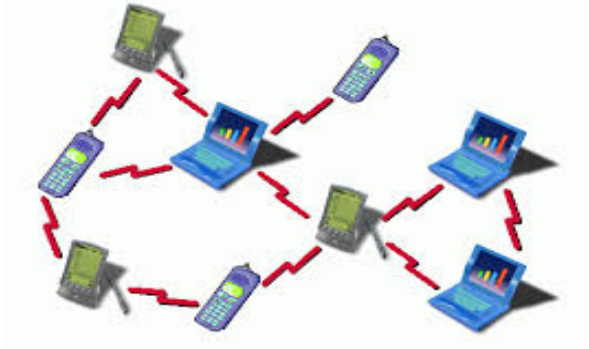

Fig.1 mobile nodes

Fig.1 shows the mobile nodes are connected to interact with each other to exchange the query to the nearest mobile nodes. Client polling is the client node receives the query from the server it process the query when the client is active.

2 RELATED WORKS

In MANETs environments, major work has been done in cache consistency scheme that are related to implements push, pull and hybrid algorithms. In push based server only update the cache nodes and then the pull based cache update the data items and the hybrid both the operation has been done.

Pull based scheme discussed into two categories: prefetching and TTL.

2.1 Prefetching

Prefetching [3] it's like a client polling, cache validation is starts on client schedule 4].It provide to achieve a strong consistency. Prefetching is based on the item request rate. And it reduce the waits state because the each and every time cannot visit the data source and increase the availability. Here it provide minimal usage of data source. In the prefetching scheme is to reduce the query delay and query latency.

2.2 TTL

Many TTL approaches [5] are introduced for MANETs, Here we used fixed TTL values are assigned for cached data items. And several mechanisms are introduce for assigning TTL values. First mechanism is adapted by the last update time. This approach is used in dynamic environments. Second approach is the value is calculated by the difference between the query time and last update time. This approach does not give the absolute solution. Finally the TTL value is computed by TCP orientation.

Assigning the TTL value by two ways, one is to fixed TTL[5] that mean the constant TTL value is assign for a set of item in the cache. Another is adaptive TTL that provide higher consistency along with lower traffic. TTL is a data which stores in cache will be expired if it has not used within threshold time since last update. It provides simplicity, good performance and flexibility to assign TTL values.

Algorithm

Function cache update
  If RN send DRP msg to DQ
    QD checks if the d_i is presented
      If d_i is presented
      The DRP msg forwarded to CN
      else

forwarded to nearest QD
CN received the request and check d_i presented or not
Suppose the d_i is presented send DERP msg to QD
Otherwise
Fetch the d_i from server
Returns the DERP response to RN
End

3 System Methodologies

3.1 Existing system Design

In the previous paper introduce server based scheme [6]. It is the process of updating the cache data item by the data source. Here the request gets from the requestor directly to the data source, the data source does not maintain the information about the client nodes. So it forward the request to the all the cache nodes and the nodes update the similar data to the data source. So the data source has overhead processing and network delay will occur.

3.2 Proposed system Design

In this paper we proposes Decentralizes distributed cache updating process, its totally a client based scheme[1]. Here the Query directory collects all the cache node information along with address in it's the group, and the cache nodes caches the query from the request nodes. Ca In this paper exposes the algorithm called pull-based.

In this algorithm implements prefetcing, piggybacks and TTL values [5][7]. Each data items in the cache nodes has the TTL values which the value same to the data source then the data item is unexpired. Suppose the data item is not similar to data source then it treated as expired. So the cache ask to the data source for updating and prefetch the desired data item from the data source.che nodes contains the already asked frequents data.

4 Architecture And Operations

This section implements interaction between different components and syatem model.

4.1 system model

The system model consists of number of mobile nodes connected to the MANETs and the each nodes are interact with others using wireless connection using WiFi. It is the access point to interact with them. The mobile nodes are used to communicate with data source with use of multi-hop communication.

The system architecture consists of Request node(RN), Cache node(CN),and Query directoty((DQ). The RN gives the request to the nearest QD's. If this QD finds the query in its cache, it forwards the request to the CN caching the item, which, in turn, sends the item to the requesting node (RN). Otherwise, it forwards it to its nearest QD[1], which has not received the request yet. If the request traverses all QD's without being found, a miss occurs and it gets forwarded to the server which sends the data item to the RN.

The server autonomously sends data updates to the CNs, meaning that it has to keep track of which CNs cache which data items. This can be done using a simple table in which an entry consists of the id of a data item (or query) and the address of the CN that caches the data. If any request to the CN caching the item, which, in turn, sends the item to the requesting node (RN).

DCIM suspends server updates when it deems that they are unnecessary. The mechanism requires the cache node to monitor the rate of local updates, and the rate of RN requests, for each data item. Each CN also monitors these values for each data item that it caches. Whenever a CN receives an update from the server.

This section describes the operation of cache updating, but first we listed the messages in the DCIM table this is implemented in COACS[1]. The RN sent the request to the Query Directory it checks where the data items presented in the CN.

4.2 Server Operation

When the server receives the CURP message from the CN. The server checks all the item are compared to the last update time. If the item have not changed then it is treated as a valid and sent SVRP is send to the CN. Suppose the data item have not similar to the server then it called as invalidation. So server send SUPR message to the CN[6]. The server inform about this through the SVRP message. In this approach is client based, the processing at the server is minimal.

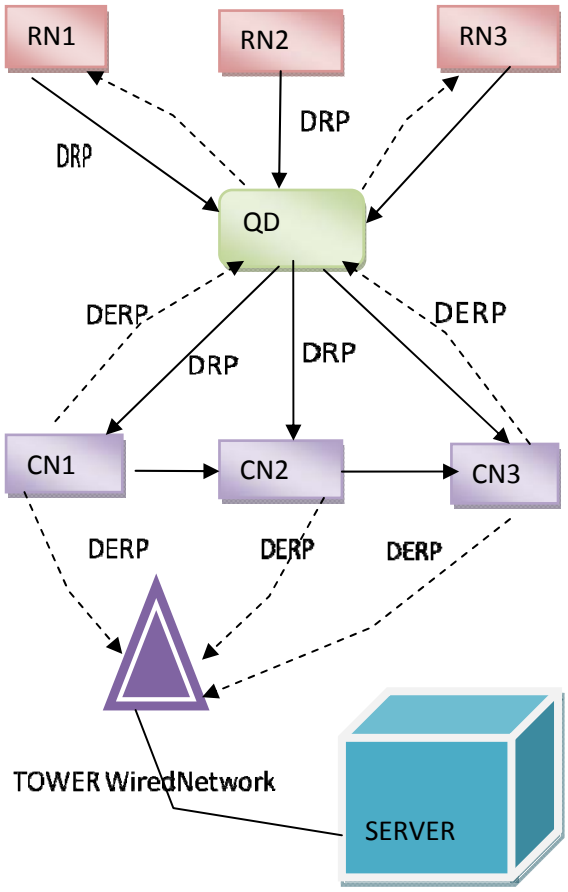

Fig 2 System model

4.3 QD operation

This section describes how the QD is assigned in this system depending on the storage capacity. Here the number of QDs are bounded in this based on the limits such as lower and upper limits, where the QD elected[1] it will not yield reduction in average QD load. The upper limits, corresponds to a delay threshold.

First we elected the maximum storage capacity node as QD and its store all the address of the cache node information. After it stores more CN information so the capacity is reduced[2]. So we Introduced a concept as DDCUM that is next compare to all the devices and then assign which one has the maximum capacity that node assign as a QD. It provides efficient way to maintain the system performance.

4.4 CN operation

The CN store the cached queries along with their responses plus their IDs. A CN maintain all the information in two tables that are cache information table which stores responses of the queries are locally cached. And Query information table that stores query specific data. The CN checks for expired data items, validation request, and request updates.

CN collects the frequently asked data and the received the request get from the QD. CN assign TTL values[5] for each data items and the value with the query if it is matched it as VALID response to the RN. Suppose the data does not matches then it as INVALID. Then the CN piggyback to server and collect the valid data item. The CN prefetch[3] the item from the server.

RESULTS

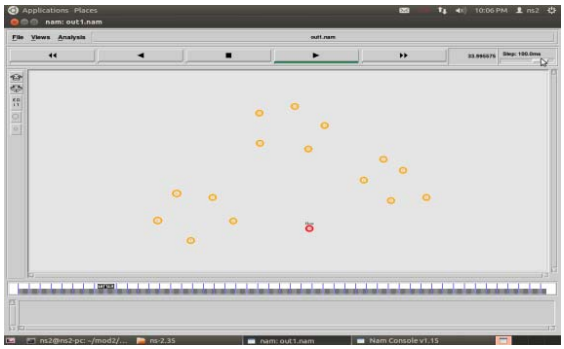

Fig 3 group formation

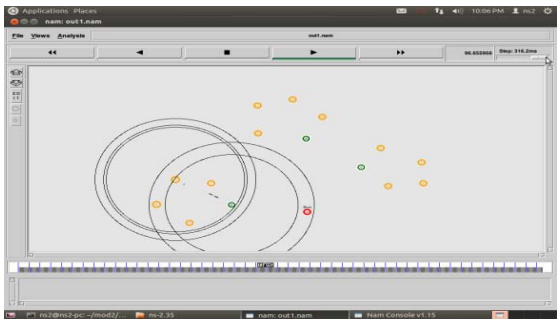

Fig 4 fixed QD

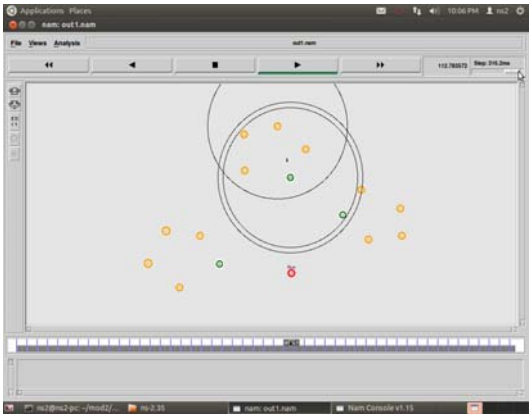

Fig 4 forward query to nearest QD

CONCLUSION

We introduced DDCUP is a client based scheme on estimating the update intervals of data item to set their expiry time. It use of piggybacking and prefetching to increase the accuracy and reduce the both traffic and query delays. DDCUP increase overall system performance.

In future plan, to replace TTL algorithm into running average formula and to develop a complete replica allocation.

REFERENCES

[1] H. Artail, H. Safa, K. Mershad, Z. Abou-Atme, N. Sulieman, “COACS: A Cooperative and adaptive caching system for MANETS”, *IEEE TMC*, v.7, n.8, pp. 961-977, 2008.

[2]G. Cao, “On Improving the Performance of Cache Invalidation in Mobile Environments,” ACM/Kluwer Mobile Network and Applications, vol. 7, no. 4, pp. 291-303, 2002.

[3] M. Denko, J. Tian, "Cooperative Caching with Adaptive Prefetching in Mobile Ad Hoc Networks," IEEE WiMob'2006, pp.38-44, June 2006.

[4] J. Cao; Y. Zhang, G. Cao, X. Li, "Data Consistency for Cooperative Caching in Mobile Environments," Computer , v.40, n.4, pp.60-66, 2007

[5] J. Jung, A.W. Berger, H. Balakrishnan, “Modeling TTL-based internet caches,” *IEEE INFOCOM 2003*, San Francisco, CA, March 2003.

[6] K. Mershad and H. Artail, “SSUM: Smart Server Update Mechanism for Maintaining Cache Consistency in Mobile Environments,” IEEE Trans. Mobile Computing, vol. 9, no. 6, pp. 778-795, June 2010.

[7] B. Krishnamurthy, C. Wills, “Study of piggyback cache validation for proxy caches in the World Wide Web,” *USENIX*, Monterey, CA, December 1997.