IJCSC International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010, pp. 329-332

Improved Data Cache Scheme Using Distributed Spanning Tree in Mobile Adhoc Network

P. Victer Paul[1], T. Vengattaraman[1], P. Dhavachelvan[1] & R. Baskaran[2]

[1]Department of Computer Science, Pondicherry University, Puducherry, India
[2]Department of Computer Science and Engineering, Anna University, Chennai, India
Email: [1]victerpaul@gmail.com, [1]vengat.mailbox@gmail.com, [1]dhavachelvan@gmail.com, [2]baskaran.ramachandran@gmail.com

ABSTRACT

Mobile ad hoc network (MANET) is an autonomous collection of mobile nodes that communicate over relatively bandwidth constrained wireless links. The frequent mobility of nodes causes topology change and ineffective communication. So, there need an interconnection technique that should guarantee network connectivity, efficient routing and maintain network performance in MANET. In MANET, every node tends to have similar task and common interests, several nodes might need to access the similar data at different times. So, by caching repeatedly accessed data item within MANET we can save external source access for same data which is costlier and insecure. In this paper, a data cache system is proposed which uses Distributed Spanning Tree (DST) as interconnection technique. Analysis from simulation of our proposed work shows that data cache system efficiency can be improved using DST technique.

Keywords: MANET Distributed Spanning Tree, Data Cache, OMNeT++

1. INTRODUCTION

Mobile ad hoc networks (MANETs) are a collection of distributed nodes which communicate using multi-hop wireless links with frequent node mobility. In MANET, nodes work for tasks of similar goal (common interest). So, most of the nodes try to access the same data at different time through their Access Point (AP). The Access Points may be located at the extremities of the MANET, where reaching them could be costly in terms of delay, power consumption, and bandwidth utilization. Additionally, the access point may connect to a costly resource (e.g., a satellite link), or an external network that is susceptible to intrusion. For such reasons and others that concern data availability and response time, MANET applications should check for the existence of the desired data inside the network before attempting to connect to the external data source [1][2]. So, instead of accessing external network for every data request, data can be cached and served on demand of other node. So, efficient Data Cache technique in MANET can enhance the various QoS Factors like data latency, bandwidth, hops required etc.

MANET deploys nodes in a dynamic situation with unlimited abilities to span the wired and wireless Internet access. Thus a mechanism needed to manage the network due to the frequent topology changes to ensure effective communication in the network. Clustering in large-scale MANET as a means of achieving scalability through a hierarchical approach in which every node in the cluster is one hop away from every other node, that is; each cluster is a diameter-1 graph. But static clusters may easily become the traffic bottleneck and single point of failure of the cluster [4]. An important problem in a MANET is finding and maintaining efficient routes since host mobility can cause topology changes [5] and performance of search and retrieval of cached data relies on the efficiency of employed routing strategies [8].

From these perspectives in this paper, an efficient data cache system is proposed which uses DST for interconnection to enhance the efficiency of system in MANET.

2. INTERCONNECTION TECHNIQUE

2.1. DST - Prologue

DST structure is organized into a hierarchy of groups. The nodes are put together in groups and groups are gathered in groups of higher level, recursively. This organization, built on top of routing tables allows the instantaneous creation of spanning trees rooted by any nodes and keeps the load balanced between the nodes [6][7]. So we virtually convert the MANET into DST and each tree should have its root node we call it as Head Node (HN) and others are Leaf Node (LN). Every LN will hold the details of its own HN. Likewise every HN will hold the complete details regarding its LNs and all other HNs in the network. The details stored in HNs and LNs in the DST is used to enhance the efficient routing

with minimum message pass. Thus the DST formation in MANET makes the efficient routing which is the preliminary requirement for effective data cache technique.

2.2. DST Formation in MANET

In MANET, DST can be formulated using five procedures namely *initialize()*, *Probe()*, *Recieve()*, *Reply()* and *Forward()* which are described below.

Procedure *initialize()* which initializes DST by creating Head Node (HN) in MANET and an array on each HN to hold its LN details. This procedure also create Leaf Node (LN) and set 'id' of every HN as their own 'id' and then it calls the procedure *Probe()*.

The procedure *Probe()*, which is called by every HN creates *probe* message *(pmsg)* and set 'id' field of message as its own id and flood the message to all nodes it is connected. On receiving a message every node executes the procedure *Recieve()*.

Procedure *Reply()* is called by LN to reply to its HN. The LN creates a *reply* message. The 'id' and 'dest' fields of the *reply* message *(rmsg)* is set to be, the 'id' of the LN and the 'id' of the HN respectively. After the *reply* message sent to HN, the LN calls the Procedure *Forward(pmsg)* to flood the *probe* message to all the nodes except the node from where it was received.

During DST formation it should be possible to get any one of the two types of messages, the *probe message* or *reply message*:

If there is a *probe* message, any one of the following would be occurred:

Case-1: The message is received by a HN: It is just discarded.

Case-2a: The message is received by a LN which is not under any HN: LN stores the Head variable as the id which it read from *pmsg*. Then call the procedures R*eply()* and F*orward(pmsg)*.

Case-2b: The message is received by a LN, which is under any HN: It is just discarded.

If there is a *reply* message, any one of the following would be occurred:

Case-1: The message is received by a LN: It just forwards it to the node bearing the id 'id'.

Case-2: If the message is received by a HN: It reads 'dest' from 'rmsg', if 'dest' equals id of current node it shows required HN is reached. It read 'id' from 'rmsg' and add it to its array, otherwise it is forwarded to destination node.

Implementation of the above discussed procedures in nodes of MANET in distributed fashion, make the MANET into a DST structure.

3. DATA CACHE SYSTEM

This section describes the mechanism to achieve Data Cache System in MANET which is explained using four procedures namely *initialize(), Request(), Receive() and doReply()*.

Procedure initialize(): This procedure is starting point of the data cache mechanism and called by every node in the MANET after the completion of DST formulation. This procedure checks whether the node is Head Node (HN) or Leaf Node (LN). If it is HN, it creates a table named HN_Table and creates a table named LN_Table with two columns to store the Data item ID and the value.

Procedure Request(): This procedure creates a Request message, *Reqmsg* in which Sour, Dest, Quer and Headindex fields are set as ID of the requester node, ID of the HN of the requester, Query of the Data item and index of HN visited respectively. *Headindex* is initialized to 0 and *Reqmsg* is sent to HN.

Procedure Doreply(): This procedure is being called by HN on receiving its *Reqmsg* from its LN and creates a DoReply message *doRepmsg* and sends the message to p.vID.

There are four different procedures which are called on receiving those four different messages used in the data cache system.

Procedure Receive(Reqmsg,v): This procedure is called by node v on receiving the message *Reqmsg*. Based on the parameter values of *Reqmsg*, any of following two cases being called.

Case 1: If v is a HN and Dest field of the Reqmsg contains the ID of v which implies that the Reqmsg is sent by some LN to the HN v.

Case 1a: Data item Identifier dID is generated from the content of the query field in Reqmsg and checked in HN_table. If found *DoReply()* procedure is called.

Case 1b: if Headindex field value is greater than number of HNs in the MANET *n(HN)*, which shows that the Data item dID is not yet cached then the Reqmsg if forwarded to External Network. Otherwise Headindex field is incremented by one and forwarded to next unvisited HN.

Case 2: If v is not a HN or field of the Reqmsg does not contain the ID of v then the message is routed to the destination following its routing protocol.

Procedure Receive(): This procedure is called by node v on receiving the message *doRepmsg*.

Case 1: If v is a LN and vID field of the doRepmsg contains the ID of v which implies that the v is a LN which is requested by its HN to send the Data item which

it cached to the requester node whose ID is attached with in message in field LN_ID.

Case 1a: Data item Identifier dID is obtained from field dID of doRepmsg. This dID is checked in LN_table for entry in Did field.

Case 1b: If consistency holds then a reply message, Repmsg is created in which vID, dID, Dval and Sour fields are set with ID of actual requester of Data item, ID of requested Data item, value of the Data item retrieved from value field of LN_Table and ID of the node v.

Case 2: If v is not a LN or field of the doRepmsg does not contain the ID of v then the message is routed to the destination following its routing protocol.

Procedure Receive(): This procedure is called by the internet (server in Internet) on receiving the Request message, *Reqmsg*.

Procedure Receive(): This procedure is called by mobile node v on receiving the Inform message, *imsg* from the outside network has to update the entry in HN_Table. Received *imsg* is checked for its intended destination from Dest field of imsg. HN_Table of node v is updated with the Data item ID and the corresponding node which holds the Data item which are retrieved from imsg. Message *imsg* is routed to the corrected destination if v is not the intended node.

These procedures are executed in distributed fashion among the nodes of MANET which involve in data cache scheme. Thus each node performs the efficient data cache technique using DST as interconnection technique.

4. SIMULATION AND ANALYSIS

This section illustrates the simulation results obtained during the analysis and we used OMNeT++, which is an object-oriented modular discrete event network simulator. We simulated a mobile ad-hoc network by placing 30 nodes randomly within a region as shown in Fig.1 and propagation delay is set as 100ms.

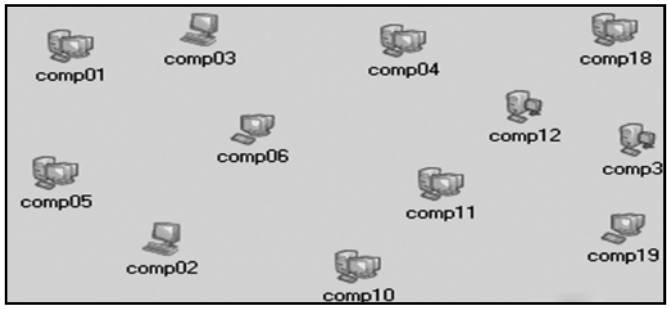

Fig.1: Simulated MANET in OMNeT++ Simulator

In our simulation, Number of messages created to form DST in MANET is about 173 and the time taken is 2.68 seconds with propagation delay of 100 ms. Fig.2 shows Number of messages required by each mobile node in MANET to form DDST.

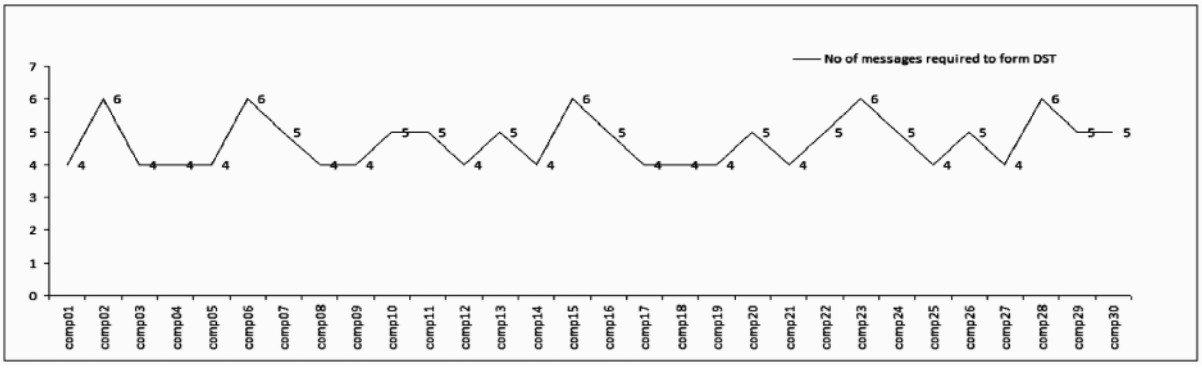

Fig.2: Number of Messages Required by Each Mobile Node to form DST

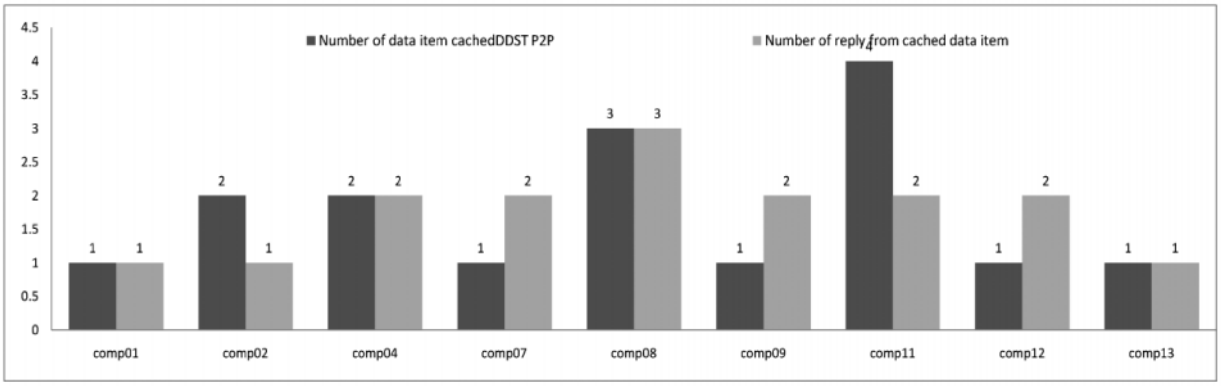

Fig.3: Number of Data Items Cached and Replied by Each Mobile Node in DST MANET

Fig.3 shows the number of data items cached by LNs and which are used to serve other LNs in MANET during the simulation. Total number of systems involved to cache the 10 data items is 14 and they cached item between the ranges of 1 to 4. These nodes accessed the data items from the external source and cached them in their buffer. Any node in MANET request for the cached data item is being served by nodes which cached the requested data item. In simulation, nearly 16 LNs are served using the cached data item. This evident that overall efficiency of the network is improved by nearly 30% - 40% in DST MANET than that of ordinary MANET.

From Fig. 3, we can clearly observe that the data cache scheme which is proposed using DST as the interconnection technique improves the efficiency of search application in MANET by reasonable rate of 30 – 40%. Some of the design issues such as data item consistency, technique to find next unvisited HN, Mobility and Hand-off of the nodes in MANET, enhanced routing technique and routing optimization technique should be taken care to improve the efficiency of the proposed data cache scheme.

5. CONCLUSION

The work presented in this paper has described an effective data cache scheme since several nodes might need to access the similar data at different times in MANET. The proposed data cache scheme uses DST as the interconnection to avoid bottle neck and balance the load among the nodes in the MANET. From the simulation analysis, it is evident that efficiency of search application can be improved by using data cache scheme with DST interconnection structure. As part of future work, we plan to apply enhanced technique using Ant Colony Optimization (ACO) to optimize the DST formulated in MANET for better performance.

REFERENCES

[1] Guohong Cao, Liangzhong Yin, Chita R. Das, "Cooperative Cache- Based Data Access in Ad Hoc Networks", Pennsylvania State University, *IEEE Computer Society*, February 2004.

[2] Hassan Artail and Khaleel Mershad,"MDPF: Minimum Distance Packet Forwarding for Search Applications in Mobile Ad hoc Networks.

[3] P. Krishna, M. Chatterjee, N. Vaidya and D. Pradhan, "A Cluster-based Approach for Routing in Ad-hoc Networks", *ACM SIGCOMM Computer Communication Review*, **27(2)**, 1997, 49–64.

[4] Azzedine Boukerche, "Algorithms and Protocols for Wireless and Mobile Ad Hoc Networks", *Wiley Series on Parallel and Distributed Computing*, Copyright © 2009 by John Wiley & Sons, Inc.

[5] Mobile Ad Hoc Networks (MANETs), http:// w3.antd.nist.gov/wahn_mahn.shtml, Web site owner: The National Institute of Standards and Technology.

[6] Sylvain Dahan, Jean-Marc Nicod and Laurent Philippe, "The Distributed Spanning Tree: A Scalable Interconnection Topology for Efficient and Equitable Traversal", *International Symposium on Cluster Computing and the Grid, 2005 IEEE*.

[7] Sylvain Dahan, "Distributed Spanning Tree Algorithms for Large Scale Traversals", *11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, 2005 IEEE.

[8] J. Broch, D. Maltz, D. Johnson, Y. Hu, J. Jetcheva, "A Performance Comparison of Multi-hop Wireless Ad hoc Network Routing Protocols Source," *Proc. Fourth Annual ACM/IEEE Int'l Conf. on Mobile Computing and Networking*, pp. 85-97, 1998.