A Novel Cache Update Algorithm for Consistency Maintenance over Cluster Based MANET

P.Kuppusamy
Associate Professor / CSE
King College of Technology
Namakkal, Tamilnadu, India
kuppusamyp.phd@gmail.com

Dr.B. Kalaavathi
Professor / CSE
K.S.R Institute for Engineering and Technology
Tiruchengode, Tamilnadu, India
kalabhuvanesh@yahoo.com

Abstract—**Data Caching improves data availability among the mobile nodes (MNs) that are connected together through unreliable wireless links. The MNs are often updating the cached data due to its dynamism. The query access rate of cached data is varied based on neighbor nodes requirements. The caching nodes must ensure the staleness of data with source. This paper proposes the novel Cluster Based Update (CBU) algorithm to avoid the stale data in caching nodes. The network is partitioning into non overlapping clusters and Cluster Head (CH) is selected based on weighting factor. The CH maintains Global Cache Index (GCI) that consist information about its cluster members and neighbor CHs. The CH maintains the query access entry for cached data in its cluster members. The source also sends data update rate to CH. Thus consistency is maintained based on data update and its query access rate through CHs. The simulation results shown that proposed algorithm reduces the latency, overhead than existing approach by increasing nodes and speed respectively.**

Keywords - cache consistency, query access rate, data update rate, cluster, cooperative cache

I. INTRODUCTION

In rapid development of technologies, MNs are interconnected through wireless links, high bandwidth satellites and access points. The mobile ad hoc network (MANET) consist more number of MNs and less number of data sources. The traffic is increased due to all MNs access the data from the single source. The traffic is considerably reduced by increasing the data availability in the network. The data caching increases the possibility of MNs accessing required data from neighbors instead of remote source [8], [17]. The key issue in cache management is the maintenance of data consistency between the caching node and the source.The MANET is used in industrial and business applications such as real-time data update like stock markets, weather report, news updates, location dependent information, electronic commerce etc [1].

Many of these applications are involved in real-time data processing. Hence, caching becomes essential to maintain communication between the caching nodes and the source to offer the desired data to MN. But, MNs can be disconnected voluntarily or involuntarily from the network due to bandwidth, battery power constraints [2]. This leads to loss of critical data when the MN is 'sleeping' or make mobility into another part of the network. To avoid such situations, network is partitioned into clusters and data must be cached locally in MNs to provide desired data to the neighbors [9]. Thus Cooperation of MNs shares the data among them to improve the network performance in terms of increasing packet delivery ratio, reduce the latency, bandwidth, traffic contention and overhead.

II. MATERIALS AND METHODS

Many cache consistency schemes have been illustrated in the literature for MANETS. The consistency schemes are categorized into three schemes: 1). pull or client based, where a caching node requests data updates from the source [1], 2). push or source-based, where the source sends updates to the caching node [1], [6] and 3) cooperative caching, where both the source and the caching node cooperate to keep the data up-to-date [8], [17].

Push-based approaches broadcasts invalidation reports (IRs) periodically to caching nodes by the source [3]. An IR normally transmits the updated data ID and the time to live (TTL). When a neighbor MNs generates a query, the caching node waits for the periodic invalidation and reply the query if it has a valid data. If the requested data is invalid, it usually waits for the periodic IR. Hence, these schemes make longer latency for the

query [3]. Modified Time Stamp (MTS) scheme, it does not wait for the periodic IR [12]. The request is forwarded to the source if it found invalid data. These schemes suffer from longer average latencies due to waiting for the periodic IR and generate large traffic when cache misses take place with high query access rate. Li et al. proposed improved IR approach, MN caches the past N reports with TTL for future purpose that are broadcasted by the source. When a MN is reconnected after a little disconnection, it can locate the missed IRs based on TTL. Thus it pulls missed IRs from the source to validate its cache. When MN misses more than N IRs, it deletes the complete cached data [11]. Sankara Gomathi et al. proposed Modified Energy Efficient Cache Invalidation Algorithm (MEECIA) that especially decrease the uplink bandwidth consumption in the mobile ad-hoc network based on Slow, Fast, Super-fast modes. The mode is selected based on threshold value specified for time and the number of MNs querying the updated data to other node. MN cannot answer the query until it receives the IR without considering the priority request. Hence it produces unnecessary latency due to wait for the IR [13]. Battery energy limitation, deficient bandwidth, data availability and accessibility are major factors in MANETs due to the mobility of the MNs. Cluster based Cooperative caching can handle the data availability issue efficiently [9]. But, this approach maintains the periodical updates without considers a query access rate. Jin et al. presents Greedy Walk-Based Selective Push (GWSP) scheme that attempts to decrease the redundancy in conventional push schemes by pushing data updates only if the caching node is likely to serve queries and also if there is no more data updates ahead of the Time-to-Refresh (TTR) expires [6]. The source maintains cached data id, state, TTR and query access rate. The source selects the caching nodes based on these factors to receive updated cached data. But, this scheme produce high overhead due increasing the size of the network. Dynamic pull-scheme offers that the caching node maintains a TTR for the cached data and pulls data updates when TTR expires. This approach produces larger traffic load and lower query latency at low query access rates, but tends to have similar delays at larger query rates [10], [15].

Xu et al. offered consistency of location-dependent data scheme to validate and update information that change their values depending on the location of the MN such as traffic reports and parking information [16]. This scheme considers spatial-temporal consistency due to MN mobility. The data can have dissimilar values for different locations. When a MN caching a data moves from one region another region, the value of the data may become stale and hence it needs to be updated. Kahol et al. presented asynchronous stateful scheme for maintaining the cache consistency [7]. In this scheme mobile support station (MSS) broadcasts data which is updated in to the relative cache node. MSS tracks of all caching nodes. Hence, it avoids all the unnecessary IR's. Thus it reduces the overhead. But, Caching node waits for cache invalidation. Hence it generates larger query latency. Song et al proposed Cost -based Cache Invalidation (CCI) to deals with MNs disconnection time and the update frequency on source side [14]. A traditional pull-based strategy accomplishes less query latency at the cost of higher traffic, whereas push-based strategy accomplishes lower traffic at the cost of larger query latency. Cooperative-based strategies provide best performance than previous strategies.

Finally, push-based schemes generates the latency due to IR reports typically contain information that is not needed to a large number of MNs. Thus it wastes MNs processing time and bandwidth. Pull based schemes, caching node sends request to the source when TTL expires. Thus it reduces overhead. But, caching node waits to renew the TTL by the source. Hence it increases the latency. To our knowledge, no existing scheme actually adapts the frequency of data update reports at the source and data query access rate by the MNs in cluster based ad hoc network. This paper addresses latency, overhead and packet delivery ratio based on data update and query access rate in MNs disconnection and mobility.

A. Problem Identification and Proposed Solution

The MNs access the desired data from the remote source in wireless network. Therefore, MNs data access latency is increased. To avoid the latency, MNs caches the data in their local cache. Hence, Caching MNs resolve their neighbor's queries by its cached data. But, consistency must be maintained between the remote source and caching nodes data to avoid invalid data accessibility. The consistency is much affected in MANET due to disconnections and mobility. This issue motivates the researcher to make study to maintain consistency in MANET based on data update to query access rate mechanism.

III. CLUSTER BASED COOPERATIVE CACHING

A. Cooperative Cache Configuration

The network topology is dividing into non overlapping clusters [9]. The cluster member's elects the Cluster Head (CH) based on the energy and connectivity in each cluster. The CH maintains two cache tables that are Local Cache Index (LCI) and the Global Cache Index (GCI). The LCI is the catalogue that consist the information of cached data present in the corresponding cluster members and GCI is the catalogue that contains the information of the cached data in the adjacent clusters. The MNs are coordinating with each other to share the cached data among them. The cluster members periodically send their Received Signal Strength (RSS), energy level to its CH. The mobility and connectivity strength of cluster member is predicted based on the RSS value.

When cluster member makes mobility, the mobility information of MN is updated on both the LCI and GCI. The new CH is elected while the energy level or connectivity of any CH is decreased than it's any of cluster members. Thus the CH is adaptively transformed based on energy and mobility. When MN send query for desired data to its CH, it checks in its LCI. If present, the cached data is send to the respective MN. Otherwise, the CH checks its GCI, to recognize whether the data is in adjacent clusters. If present, the data item is send to the respective MN through its CH and cached in the MN. This information about cached data is updated in CH's LCI catalog to share with neighbors.

B. Prediction of Disconnection and Mobility

The mobility states of MNs can observe through SNR (Signal-to-Noise Ratio) that is computed based on RSS (Received Signal Strength) at regular intervals [9]. These observations are calculated the CHs, MNs locations, and strength of the connectivity. The MN disconnects from the network to save the energy during sleep period. The CH does not receive reply from the MN during that sleep period. But, CH receives the updates from the source and stores in its local cache for a little bit time. The MN sends the cache update invalidation request to its CH when reconnect (awake) to the network. Then CH transmits the updated data. CH deletes the data during MN in sleep mode for long time. Thus CH resolves the disconnection crisis in clusters.

C. Cluster based Update Algorithm

The preceding schemes mostly discuss on how cache consistency should be maintained once the source has updated the data. In such schemes, the data is updated directly by the source without considering status of the data in caching nodes.

In some cases, the source waits for certain period of time ahead of updating the data, as in adaptive Lease protocol [4]. The exploit of update delay can reduces the consistency maintenance cost. The source waits for a little bit predefined delay to update the data in small ad hoc network with stable connectivity as in flexible cache maintenance [5]. However, it does not offer on how long the source needs to wait before updating the data in large size dynamic network. It does not also provide the update delay when MNs makes movement in large ad hoc network. The CBU Algorithm provides better performance in updating the data using CH to its caching nodes in the cluster. In CBU approach, the source monitors data update rate, D_{ur}, and query access rate D_{qr} of MNs for every data D. The caching nodes also maintain these values for each data D that it caches. The caching node computes ratio for every data when it obtains data update from the source and compares it to a threshold value. Data update-to-query access ratio is given in below equation.

$$\frac{D_{ur}}{D_{qr}}$$

The source avoids unnecessary data updates that have less query access rate than update rate. This data update-to-query access ratio value sends to its CH. When the ratio is greater than threshold Γ, the caching node deletes the D from the cache and sends Data Delete (DD) message to its home CH with data update and query access ratio. Hence CH deletes the information associated with D from its GCI. Subsequently, CH sends a Not Update message to the source for the D. In the meantime, MN may receive new data from the source and it attempts to be a caching node for this data. MN sends cache information about this data to its CH. But, CH replies to the MN that it does not cache this data. Hence MN does not cache the data and it avoids unnecessary updates. Thus proposed approach reduces the overhead, energy consumption in the network.

Cluster based Update Algorithm

```
While (data D is cached in caching nodes)
        Source computes data update rate
        Caching nodes computes query access rate
        CH receives cache node id, D id, TTL
        Receives data update rate from source
        Receives query access rate from cache MN
        Compute data update-to-query access ratio
   if(TTL>0) then
        source send data updates to corresponding caching node
        cache node updates it in CH
    if ((Dur / Dqr) ≥ Γ ) then
        cache node sends DD to CH
                CH updates the GCI catalogue and send to source
   else
        Cache node updates the data
    end if
    end if
end while
```

D. Query Access in Cluster

The MN sends a query to its home Cluster Head (CH_i) to get a desired data. The CH_i searches the requested data in its LCI.

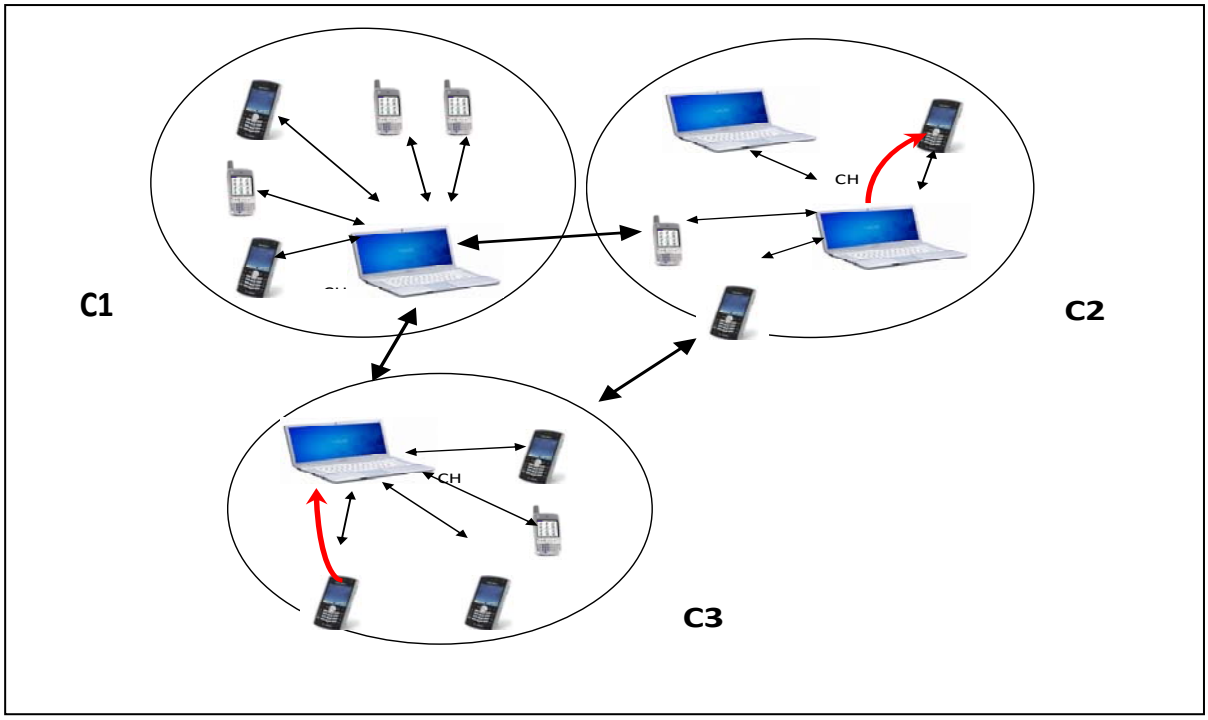

Figure 1: Cluster Based Data Transmission over MANET

The CH_i replies to MN while data is in its home cluster caching nodes. Otherwise CH_i checks GCI that contains adjacent cluster head's (CH_j) information. The CH_j replies to CH_i about Caching node that contains the requested data. Subsequently, MN receives the data from CH_j's caching node and intimate to its CH_i about the data. This accessing MN becomes a caching node for this data in its cluster to serve the neighbors. The clocks of the MNs including CHs, source and cluster members have synchronized in the MANET. The Figure.1 shows that MN in C_3 send query to its home Cluster Head (CH_3). But, CH_3 does not have the desired data and it searches in GC that found data in C_2. The CH_3 send query to neighbor CH_2, sends the information of MN containing data as a REP to the CH_3. Then CH_3 forwards the REP to MN. Once MN receives data, it caches the data item in its local cache and CH_3 updates the LCI with MN.

IV. SIMULATION

The Network Simulator Version-2 (NS2) is used to simulate the proposed scheme with channel capacity of MNs is set as 2 Mbps [18]. The simulation uses the Distributed Coordination Function (DCF) of IEEE 802.11 for wireless LANs as the MAC layer protocol. It has the functionality to notify the network layer about link breakage. In our simulation, MNs move in a 1000×1000 m region for 100 sec simulation time with network size is 60 nodes. Assume each node moves randomly with the same average speed 5 m/ sec. All MNs have the same transmission range of 250 m. The simulated traffic is Constant Bit Rate (CBR).

A. Performance metrics

In this paper, proposed schme CBU is compared with Flexible Cache Consistency (FCPP) Maintenace [5]. We evaluate the performance according to the following metrics.

Average query latency: The average latency involved in submitting a query from the client and getting the reply.

Control overhead: The control overhead is defined as the total number of control packets normalized by the total number of received data packets.

TABLE I. SIMULATION SETTINGS AND PARAMETERS

Parameters	Values
No. of nodes	10,20,30,40,50 and 60
Area size	1000×1000
Radio range	250 m
Simulation time	100 sec
Cache size	1000 KB
Mobility model	Random way point
Speed	2, 5, 7, 10, 12, 15 m/sec
Database size	1000 items
Data size	500 bytes
Mean query generating interval	5sec

V. RESULTS AND DISCUSSIONS

A. *Based on nodes*

The simulation scenario is configured by changing the number of MNs as 10,20,30,40,50, and 60. The mobility speed of MN is 5 m/sec.The addition of MNs in the network, increses the caching nodes. Due to increase of caching nodes in clusters, the query latency and overhead are increased slightly, and overall packet delivery ratio is decreased.

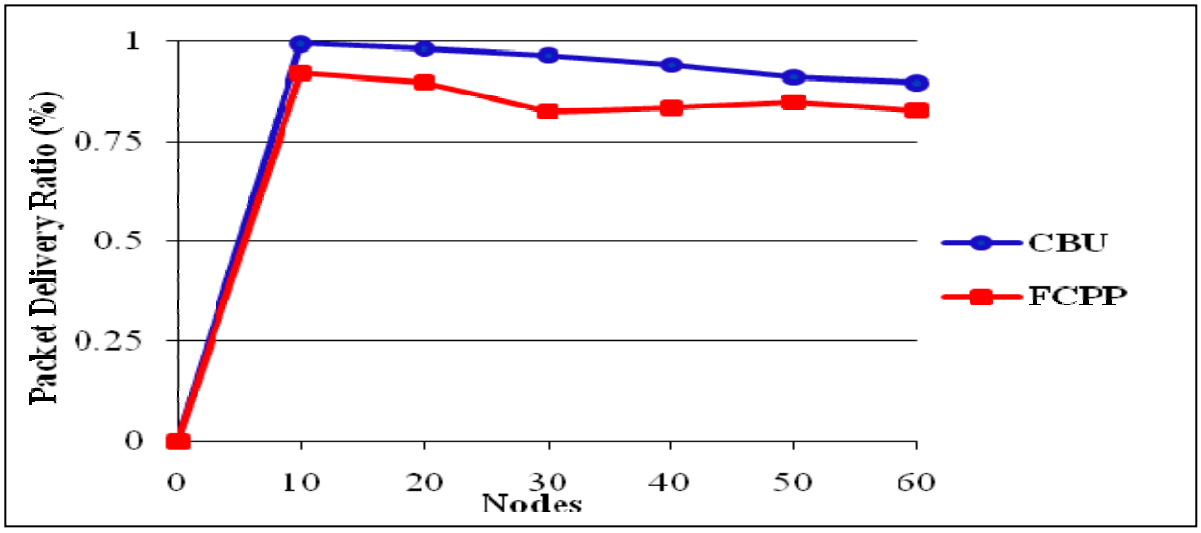

Figure. 2: Nodes Vs Packet Success Ratio

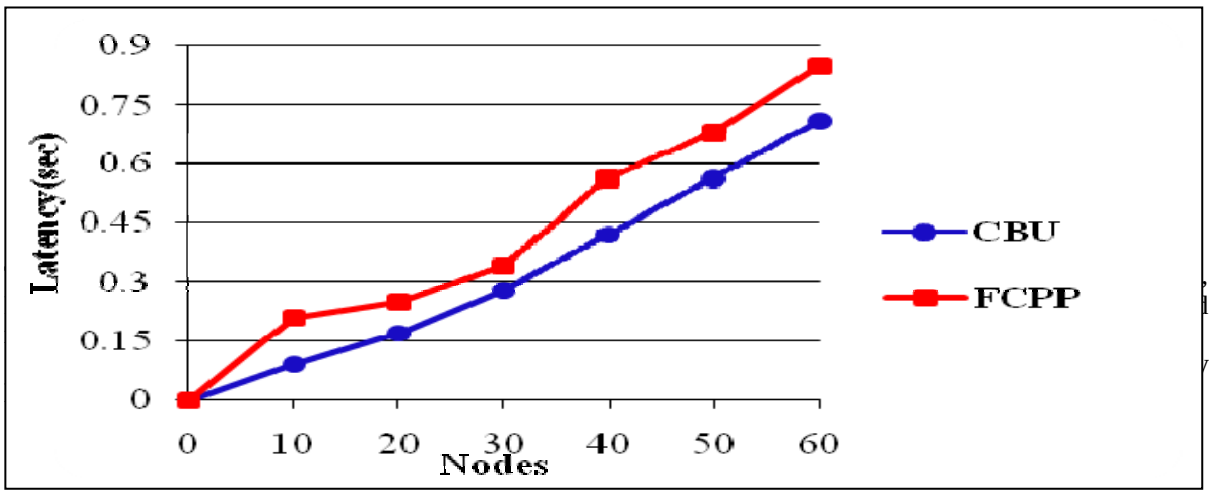

Figure. 3: Nodes Vs Latency

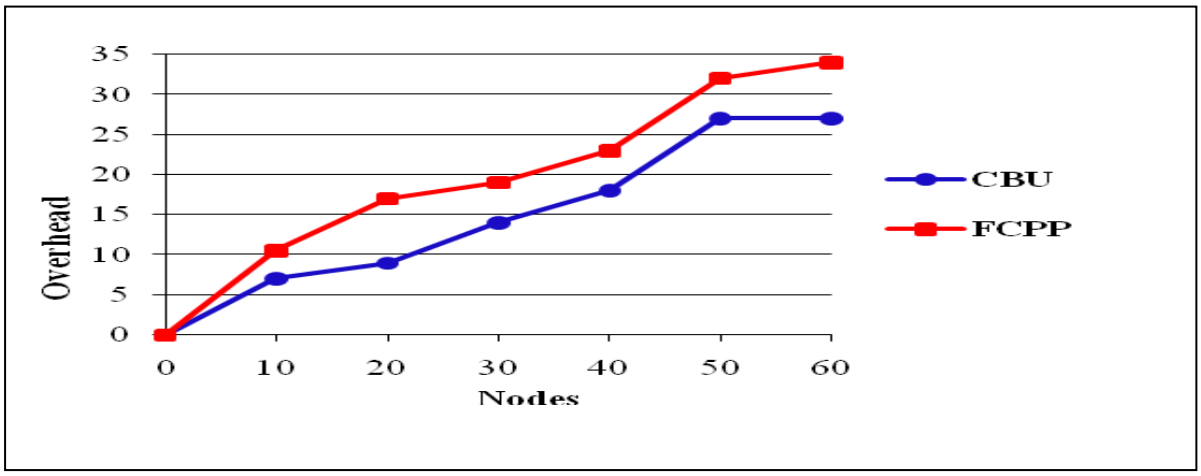

Figure. 4: Nodes Vs Overhead

B. Based on speed:

This second simulation scenario is configured by varying the mobility speed of the MNs as 1, 3, 5, 7, 9, and 12 m / sec with 60 nodes. The Fig. 5 shows that the proposed CBU scheme offers more packet delivery ratio than the existing FCPP schemes.

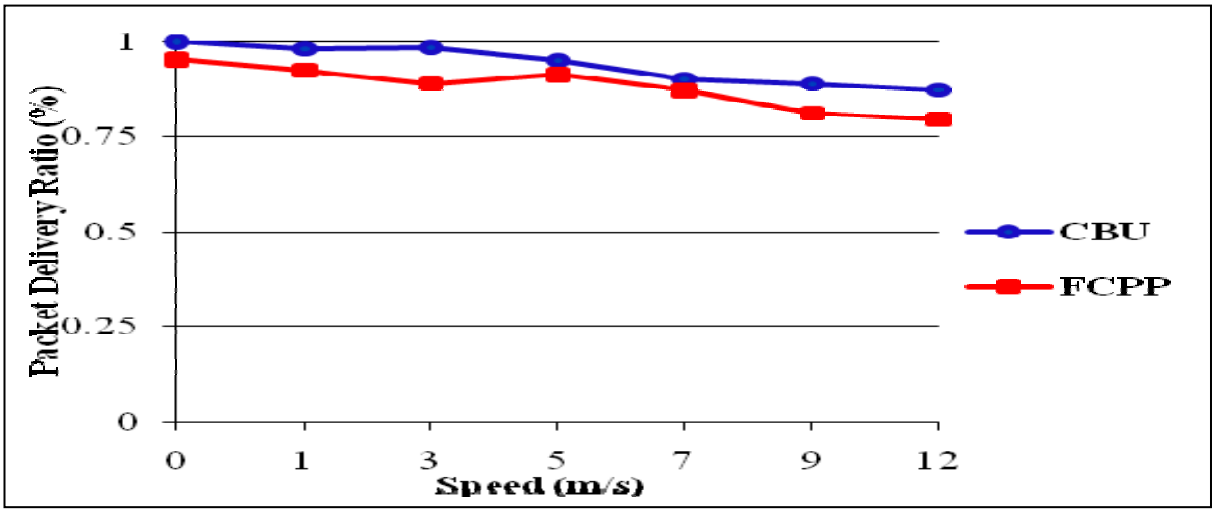

Figure. 5: Speed Vs Packet Delivery Ratio

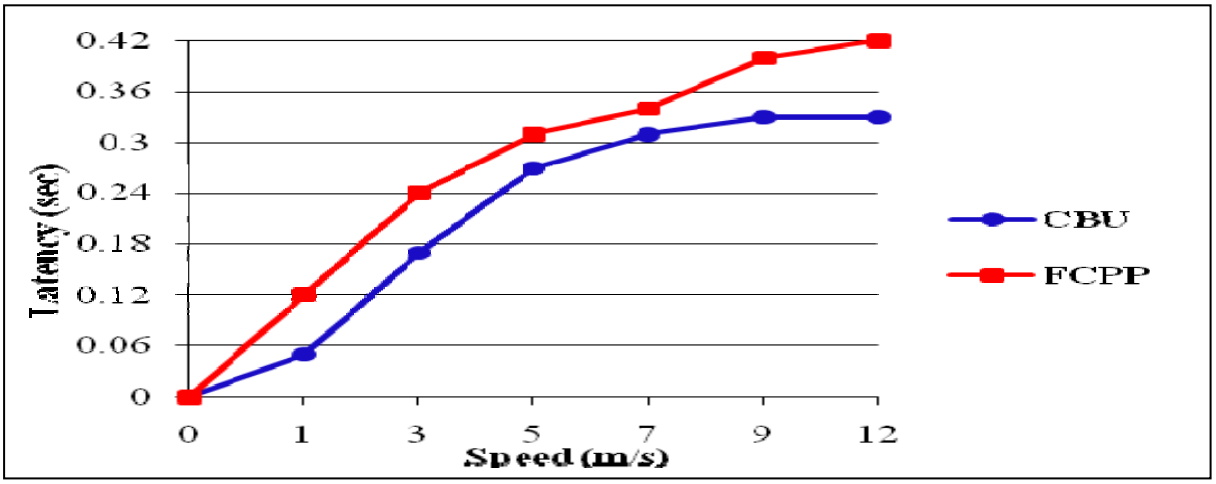

Figure. 6: Speed Vs Latency

Fig.6 and 7 shown that proposed CBU achieves achieves less query latency and overhead. Since MNs always communicate with any one of CH. Hence CH resolves the neighbors queries. The simulation shown that the proposed Cluster Based Update algorithm reduces latency 6.6%, and overhead 3.5%, and increases packet success ratio 5.4% than FCPP respectively with respect to increasing number of nodes. Also CBU reduces latency 3.7% and overhead 9%, and increases packet delivery ratio 4.2% than FCPP respectively with respect to mobility speed. The results proved that the proposed algorithm CBU provides better performance than the existing approaches FCPP.

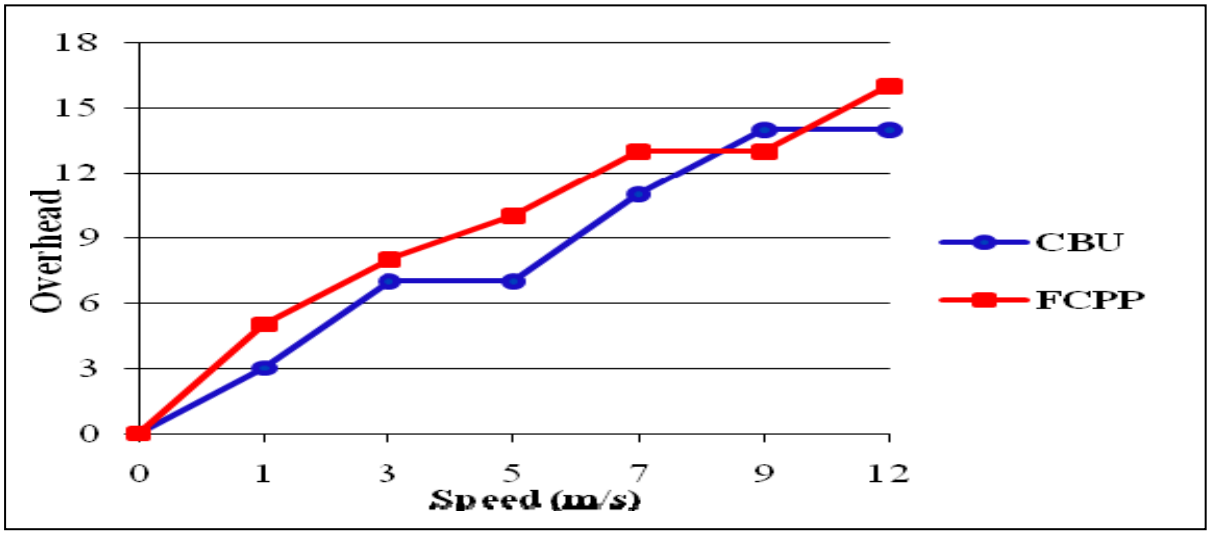

Figure. 7: Speed Vs Overhead

VI. CONCLUSION

This paper presents a Cluster Based Update(CBU) algorithm for cooperative caching in MANET. CBU improves the data consistency among the remote source and caching nodes. Because, Data updates based on update -to- query ratio. Hence, MNs acees the current data from caching nodes as it is in the source. Thus, CBU decrease the query latency, overhead. Also it decreases the energy consumption of MNs based on partition the network into clusters. The CHs shared their catalogue information with adjacent CHs to improve the performance. Thus the cooperative caching improves the data availability in MANET. The simulation result shown that the proposed CBU algorithm out perfoms the existing FCPP algorithm.

REFERENCES

[1] D. Barbara,and T. Imielinski, “Sleepers and Workaholics: Caching Strategies in Mobile Environments”, MOBIDATA: An Intractive journal of mobile computing. Vol. 1 No.1. 1994.

[2] J. Cai and K. Tan, “Energy-Efficient Selective Cache Invalidation”, Wireless Networks Journal, vol. 5, no. 6, pp. 489-502, Dec. 1999.

[3] G. Cao, “On Improving the Performance of Cache Invalidation in Mobile Environments”, ACM/Baltzer Mobile Networks and Application, 2002.

[4] V. Duvvuri, P. Shenoy and R. Tewari, “Adaptive leases: A strong consistency mechanism for the world wide web”, IEEE Transaction of Knowledge Data Engineering, pp: 1266-1276, 2003.

[5] Y. Huang, J. Cao, B. Jin, X. Tao and J. Lu, “Flexible Cache Consistency Maintenance over wireless adhoc network”, IEEE Transaction on Parallel and Distributed Systems, Vol 21, No.8. 1150-1161, 2010.

[6] H. Jin, J. Cao, and S. Feng, “A Selective Push Algorithm for Cooperative Cache Consistency Maintenance over MANETs,” Proc. Third IFIP Int’l Conf. Embedded and Ubiquitous Computing, Dec. 2007.

[7] A. Kahol , “A Strategy to Manage Cache Consistency in a Distributed Mobile Wireless Environment,” IEEE Trans.Parallel and Distributed Systems, vol. 12, no. 7, 2, pp. 686-700, 2001.

[8] P. Kuppusamy, K. Thirunavukkarasu, B. Kalaavathi, “Review of cooperative caching strategies in mobile ad hoc networks”. International Journal of Computer Applications, 2011.

[9] P. Kuppusamy, K. Thirunavukkarasu, B. Kalaavathi, “Cluster based cooperative caching technique in mobile ad hoc networks”. European Journal of Scientific Research, 69: pp: 337-349, Jan. 2012.

[10] J. Lan, X. Liu, P. Shenoy and K. Ramamritham, Consistency Maintenance in Peer-to-Peer File Sharing Networks, the 3rd IEEE Workshop on Internet Applications, 2003. (TTR)

[11] W. Li, E. Chan, Y. Wang, and D. Chen, “Cache Invalidation Strategies for Mobile Ad Hoc Networks,” Proc. Int’l Conf. Parallel Processing, Sept. 2007.

[12] S. Lim, W.-C. Lee, G. Cao, and C.R. Das, “Performance Comparison of Cache Invalidation Strategies for Internet-Based Mobile-Ad Hoc Networks,” Proc. IEEE Int’l Conf. Mobile Ad-Hoc and Sensor Systems, pp. 104-113, Oct. 2004.

[13] S. Sankara Gomathi and S. Krishnamurthi, "Modified Energy Efficient Cache Invalidation Algorithm in Mobile Environment", Proceedings of the World Congress on Engineering July 2007 Vol II ISBN:978-988-98671-2-6.

[14] Y. Song-Yi, S.Wonmin, J.Sungwon and P.Sooyong, "A Cost Effective Cache Consistency Method for Mobile Clients in Wireless Environment", Database System for Advanced Applications. LNCS, vol. 2973, pp. 908-915, 2004.

[15] B. Urgaonkar, A. Ninan, M. Raunak, P. Shenoy and K. Ramamritham, "Maintaining Mutual Consistency for Cached Web Objects", In Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS), Phoenix, AZ, April 2001. (TTR)

[16] J. Xu, X. Tang, and D. Lee, "Performance Analysis of Location- Dependent Cache Invalidation Schemes for Mobile Environments", IEEE Transaction on Knowledge and Data Engineering, vol. 15, no. 2, pp. 474-488, Feb. 2003.

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

[18] Network Simulator, http://www.isi.edu/nsnam/ns.

P.Kuppusamy received his Bachelor's degree in Computer Science and Engineering from Madras University, India, in 2002 and Master's degree in Computer Science and Engineering from Anna University, Chennai, India, in 2007. Currently, he is working an Associate Professor at the Department of Computer Science and Engineering, King College of Technology, Namakkal. He is carrying out PhD research work in association with Anna University of Technology, Coimbatore. His specializations include networks, Cryptograhy and distributed computing. His current research interests are ad hoc networks and mobile computing.

Dr.B.Kalaavathi received the B.E. (Computer Science and Engineering) degree from Bharathiyar University in 1993, M.Tech from Pondicherry University in 2000 and Ph.D from Periyar University, India, in 2010. She is currently working as a Professor and Head in Department of Computer Science and Engineering, K.S.R Institute for Engineering and Technology, Tiruchengode. She is a member of CSI & ISTE INDIA. Her current areas of interest include Mobile Computing, Data Structures and Algorithms Analysis.