Implementation study of Hybrid caching scheme in Wireless P2P networks with Asymmetric Cooperative caching

B.S.V.N.L.GAYATRI ,Dr M.H.M. KRISHNA PRASAD

Dept of Information Technology,
UCEV–JNTUK, Vizianagaram, A.P, India

***Abstract*—As the day to day applications in mobile computing are increasing, different wireless peer-peer networks came into existence. There are several protocols and schemes are implemented to increase the network performance. The most implemented technique used to improve the performance of data access is Caching. Cooperative caching, which allows the sharing and coordination of cached data among multiple nodes, can further explore the potential of the caching techniques. Due to mobility and resource constraints of ad-hoc networks, cooperative caching techniques designed for wired network may not be applicable to ad-hoc networks. Hence, this paper presents a asymmetric cooperative caching scheme to efficiently support data access in ad-hoc networks. A Hybrid approach (Hybrid Cache) which can further improve the performance by taking advantage of Cache Data and Cache Path by avoiding general weaknesses. This proposed scheme increases the system performance in aspect of bandwidth, power utilization and end-end delay.**

Index Terms—Wireless networks, P2P networks, cooperative cache, Hybrid cache

I. INTRODUCTION

Recent explosive growth in computer and wireless communication technologies has led to the development of Mobile Adhoc Networks' (MANETs) that are constructed only from mobile hosts. The interest in developing mobile wireless ad hoc networking solutions has been due to their flexibility, ease of deployment and potential applications such as battlefield, disaster recovery, outdoor assemblies, etc. For example, a class of students may need to interact during a lecture, friends or business associates may run into each other in an airport terminal and wish to share files, or a group of emergency rescue workers may need to be quickly deployed after an earthquake or flood. In such situations, a collection of mobile hosts with wireless network interfaces may form a temporary network without the aid of any established infrastructure or centralized administration. This type of wireless network is known as an *ad hoc network.*

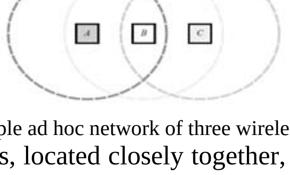


Figure1: A simple ad hoc network of three wireless mobile hosts

If only two hosts, located closely together, are involved in the ad hoc network, no real routing protocol or routing decisions are necessary. In many ad hoc networks, though, two hosts that want to communicate may not be within wireless transmission range of each other, but could communicate if other hosts between them also participating in the ad hoc network are willing to forward packets for them. For example, in the network illustrated in Figure 1, mobile host *C* is not within the range of host *A*'s wireless transmitter (indicated by the circle around *A*) and host *A* is not within the range of host *C*'s wireless transmitter. If *A* and *C* wish to exchange packets, they may in this case enlist the services of host *B* to forward packets for them, since *B* is within the overlap between *A*'s range and *C*'s range. Indeed, the routing problem in a real adhoc network may be more complicated than this example suggests, due to the inherent non uniform propagation characteristics of wireless transmissions and due to the possibility that any or all of the hosts involved may move at any time.

Most of the previous researches [1–5] in MANETs focus on the development of dynamic routing protocols that can improve the connectivity among mobile hosts which are connected to each other by one-hop/multi-hop links. Although routing is an important issue in adhoc networks, other issues such as data access are also very important since the ultimate goal of using such networks is to provide information access to mobile hosts [6]. One of the most attractive techniques that improve data availability is caching. In general caching results in (i) enhanced QoS at the clients – i.e., lower jitter, latency and packet loss, (ii) reduced network bandwidth consumption, and (iii) reduced data server/source workload.

In addition, reduction in bandwidth consumption infers that a properly implemented caching architecture for a MANET environment can potentially improve battery life of mobile clients. Caching has been proved to be an important technique for improving the data retrieval performance in mobile environments [15–18]. With caching, the data access delay is reduced since data access requests can be served from the local cache, thereby obviating the need for data transmission over the scarce wireless links.

Various routing algorithms have been designed to route messages in ad hoc networks. To reduce the bandwidth consumption and the query delay, the number of hops between the data center and the requester should be as small as possible.

Routing protocols in conventional wired networks generally use either *distance vector* routing algorithms, both of which require periodic routing advertisements to be broadcast by each router. In distance vector routing [1,5], each router broadcasts to each of its neighbour routers its view of the distance to all hosts, and each router computes the shortest path to each host based on the information advertised by each of its neighbours.

This paper describes the design and performance of a routing protocol for adhoc networks that instead uses *dynamic source routing* of packets between hosts that want to communicate. Source routing is a routing technique in which the sender of a packet determines the complete sequence of nodes through which to forward the packet; the sender explicitly lists this route in the packet's header, identifying each forwarding "hop" by the address of the next node to which to transmit the packet on its way to the destination host. Many routing algorithms (such as AODV [3] and DSR [4]) provide the hop count information between the source and destination.

In this paper, we design and evaluate cooperative caching techniques to efficiently support data access in ad hoc networks we propose three schemes: CachePath,CacheData and HybridCache. In CacheData, Intermediate nodes cache the data to serve future requests instead of fetching data from the data center. In CachePath, mobile nodes cache the data path and use it to redirect future requests to the nearby node which has the data instead of the faraway data center. To further improve the performance, we design a hybrid approach (HybridCache), which can further improve the performance by taking advantage of CacheData and CachePath while avoiding their weaknesses.

II. RELATED WORK

In adhoc networks, a data request is forwarded hop-by-hop until it reaches the data center and then the data center sends the requested data back. Various routing algorithms have been designed to route messages in adhoc networks. To reduce the bandwidth consumption and the query delay, the number of hops between the data center and the requester should be as small as possible.

Due to mobility, the node which caches the data may move. The cached data may be replaced due to the cache size limitation. As a result, the node which modified the route should reroute the request to the original data center after it finds out the problem.To deal with this issue, a node N_i caches the data path only when the caching node, say N_j is very close. Intuitively, if the network is relatively stable, the data update rate is low, and its distance to the caching node (denoted as $H(i,j)$) is much lower than its distance to the data center (denoted as $H(i,C)$), the routing node should cache the data path. Note that $H(i,j)$ is a very important factor. If $H(i,j)$ is small, even if the cached path is broken or the data are unavailable at the caching node, the problem can be quickly detected to reduce the overhead. Certainly,$H(i,j)$ should be smaller than $H(i,C)$. The number of hops that a cached path can save is denoted as

$$Hsave = H(i,C) - H(i,j)$$

where H_{save} should be greater than a system tuning threshold,called T_H, when CachePath is used.

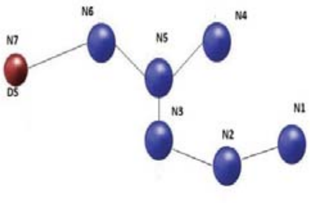


Fig 2: Adhoc network with nodes

III. ASYMMETRIC APPROACH

An asymmetric approach is proposed to reduce the overhead of copying data between the user space and the kernel space, and hence to reduce the data processing delay. The proposed algorithm well considers the caching overhead and adapts the cache node selection strategy to maximize the caching benefit on different MAC layers. Our results show that the asymmetric approach outperforms the symmetric approach in traditional 802.11-based adhoc networks by removing most of the processing overhead.

Due to cache consistency issue in both CacheData and CachePath, We have done some work on maintaining strong cache consistency in single-hop based wireless environment. However, due to bandwidth and power constraints in adhoc networks, it is too expensive to maintain strong cache consistency, and the weak consistency model is more attractive.

A simple weak consistency model can be based on the Time-To-Live (TTL)mechanism, in which a node considers a cached copy up-todate if its TTL has not expired, and removes the map from its routing table (or removes the cached data) if the TTL expires. As a result, future requests for this data will be forwarded to the data center.

IV. A HYBRID CACHING SCHEME (HYBRIDCACHE)

CachePath and CacheData can significantly improve the system performance. We also found that CachePath performs better in some situations such as small cache size or low data update rate, while CacheData performs better in other situations. To further improve the performance, we propose a hybrid scheme *HybridCache* to take advantage of CacheData and CachePath while avoiding their weaknesses. Specifically, when a node forwards a data item, it caches the data or path based on some criteria. These criteria include the data itemsize s_i the TTL time TTL_i, and the H_{save}. For a data item d_i, the following heuristics are used to decide whether to cache data or path:

□ If s_i is small, Cache Data should be adopted because the data item only needs a very small part of the cache, otherwise, cachePath should be adopted to save cache space. The threshold value for data size is denoted as T_s.

□ If TTL_i is small, CachePath is not a good choice because the data item may be invalid soon. Using CachePath may result in chasing the wrong path and end up with re-sending the query to the data center. Thus, CacheData should be used in this situation. If TTL_i is large, CachePath should be adopted. The threshold value for TTL is a system tuning parameter and denoted as T_{TTL}.

□ If H_{save} is large, CachePath is a good choice because it can save a large number of hops; otherwise, CacheData should be adopted to improve the performance if there is enough empty space in the cache. We adopt the threshold value T_H used in CachePath as the threshold value.

V. IMPLEMENTATION OF ROUTING PROTOCOLS AND ALGORITHM

In this paper, Greedy Cache Placement algorithm is effectively implemented. The proposed algorithm well considers the caching overhead and adapts the cache node selection strategy to maximize the caching benefit on different MAC layers. Our results show that the asymmetric approach outperforms the symmetric approach in traditional 802.11- based adhoc networks by removing most of the processing overhead. In mesh networks, the asymmetric approach can significantly reduce the data access delay compared to the symmetric approach due to data pipelines.

There are two routing protocol used:(i)Ad-hoc On-demand Distance Vector (*AODV*) *routing protocol* (ii) Dynamic Source Routing (DSR)

The data server needs to measure the benefit of caching a data item on an intermediate node and use it to decide whether to cache the data. After an intermediate node (Ni) caches a data item, node (Ni) can serve later requests using the cached data, instead of forwarding the requests to the data server, saving the communication overhead between node(Ni) and the data center. However, caching data at node (Ni) increases the delay of returning the data to the current requester, because it adds extra processing delay at Ni, and the data reassembly at node (Ni) may affect possible pipelines.

Adhoc on Demand Distance Vector (AODV) is routing protocol, does not require nodes to maintain routes to destinations that are not actively used. The protocol uses different messages to discover and maintain links: Route Requests(RREQs), RouteReplies (RREPS) and RouteErrors (RERRS). These message types are received via UDP and normal IP header processing applies. AODV does not repair a broken path locally. The main advantage is that routes are established on demand and destination sequence numbers are used to find the latest route to the destination. The connection setup delay is less.

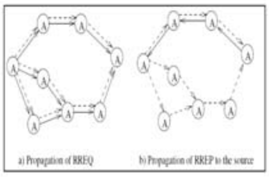


Fig3: AODV route discovery

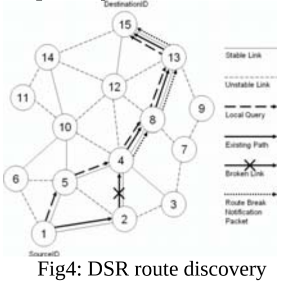


Fig4: DSR route discovery

Dynamic Source Routing (DSR) designed for mobile adhoc networks with up to around two hundred nodes with possibly high mobility rate. The protocol works "On Demand" i.e. without any periodic updates. Packets carry along the complete path they should take this reduces overheads for large routing updates at the networks. The protocol is composed of route discovery and route maintenance. At route discovery, a source requesting to send a packet to a destination broadcasts a RouteRequest(RREQ) packet. Nodes receiving RREQ search in their RouteCache for a route to the destination. When originating or forwarding a packet using a source route, each node transmitting the packet is responsible for confirming that data can flow over the link from that node to the next hop.

VI. The Simulation Model

The simulation is based on *ns-2* [8] with the CMU Wireless extension. In our simulation, both AODV [3] and DSDV [4] were tested as the underlying routing algorithm. Because our schemes do not rely on specific routing protocols, the results from AODV and DSDV are similar. To save space, only the results based on AODV are shown here. The node density is changed by choosing the number of nodes between 50 and 100 in a fixed area. We assume that the wireless bandwidth is 2 Mb/s, and the radio range is 250m.

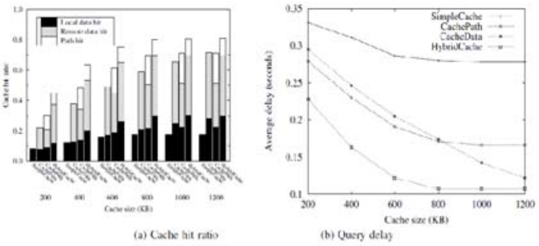
Most system parameters are listed in Table I. The second column lists the default values of these parameters. In the simulation, we may change the parameters to study their impacts. The ranges of the parameters are listed in the third column. For each workload parameter (e.g., the mean TTL time or the mean query generate time), the mean value of the measured data is obtained by collecting a large number of samples such that the confidence interval is reasonably small. In most cases, the 95% confidence interval for the measured data is less than 10% of the sample mean.

TABLE I

SIMULATION PARAMETERS

Parameter	Default value	Range
Database size n	1000 items	
s_{min} (KB)	1	
s_{max} (KB)	10	
Number of nodes	100	50 to 100
v_{max} (m/s)	2	2, 20
Bandwidth (Mb/s)	2	
TTL (secs)	5000	200 to 10000
Pause time (secs)	300	
Client cache size (KB)	800	200 to 1200
Mean query generate time T_{query} (secs)	5	1 to 100
T_H	2	1 to 5
T_s (% of $(s_{min} + s_{max})$)	40	10 to 100
T_{TTL} (secs)	5000	500 to 10000

Because of the high cache hit ratio, the proposed schemes perform much better than SimpleCache (see Fig. 5). Because the hop count of local data hit is 0 and the average hop count of remote data hit is lower than that of path hit, CacheData achieves low query delay. Comparing these three proposed schemes, we can see that HybridCache performs much better than CacheData or CachePath, because HybridCache applies different schemes (CacheData or CachePath) to different data items, taking advantages of both CacheData and CachePath. As the result of the high local data hit ratio, remote data hit ratio and overall cache hit ratio, HybridCache achieves the best performance compared to other schemes.



(a) Cache hit ratio (b) Query delay

Fig5: System performance as a function of the cache size

***Simulation Results*:** The simulation performed on ns-2[7].The implementation of the Cooperative cache layer is ported from the real system implementation, but simplified to fit the simulator. At the transport layer we set the MTU (Maximum Transmission Unit) to be 500 bytes. When the data packet received from the upper layer exceeds the size of MTU, it breaks the packet into fragments and passes the fragments to the routing layer.

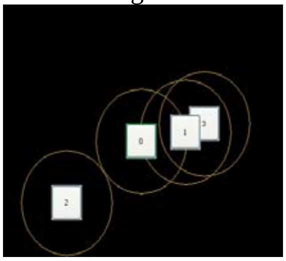


Fig6:wirelessP2Pnetwork with 4 nodes

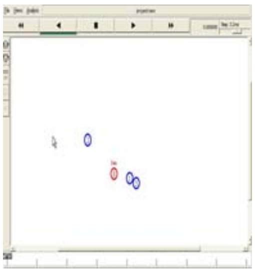
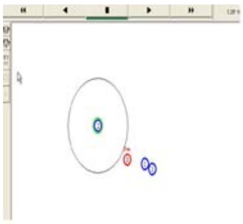


Fig6.1: nodes advertising themselves

Wireless P2P network 4 nodes is shown in fig6.Some screenshots are shown from fig6 to fig6.9.Since the standard ns-2 does not support multichannel, we add the multi-interface and multi-channel functionality based on techniques provided [9].



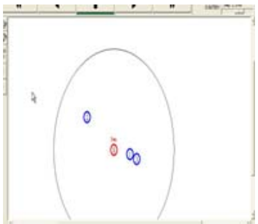
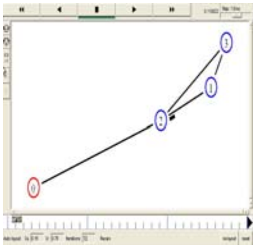


Fig6.2: route discovery process Fig: 6.3 node transmitting the packets

AODV uses a destination sequence number for each route entry. The destination sequence number is created by the destination for any route information it sends to requesting nodes. Using destination sequence numbers ensures loop freedom and allows to know which of several routes is more "fresh ".Given the choice between two routes to a destination a requesting node always selects the one with Greatest sequence number. When a node wants to find a route to another one, it broadcasts a RREQ to all the network till either the destination is reached or another node is found with a "freshenough"route to the destination. Then a RREP is sent back to the source and the discovered route is made available.



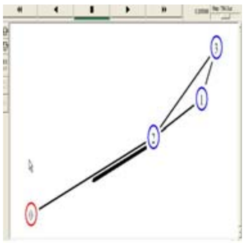
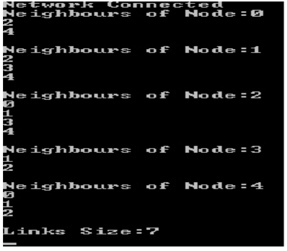
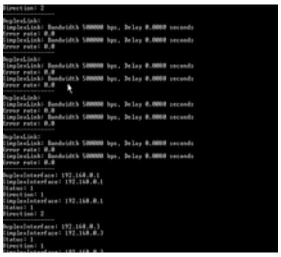


Fig: 6.4 Data passing from node1 to node2 Fig6.5: Data passing from node2 to node 0

When a node detects that a route to a neighbour node is not valid it removes the routing entry and sends a RERR message to neighbours that are active and use the route. This is possible by maintaining active neighbourlists .This procedure is repeated at nodes that receive RERR messages.A source that receives an RERR can reinitiate a RREQ.

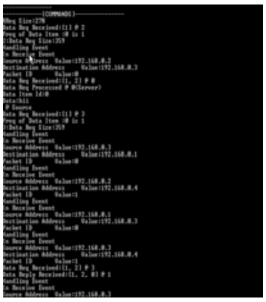


```
Network Connected
Neighbours of Node:0
2
4

Neighbours of Node:1
2
3
4

Neighbours of Node:2
0
1
3
4

Neighbours of Node:3
1
2

Neighbours of Node:4
0
1
2

Links Size:7
_
```

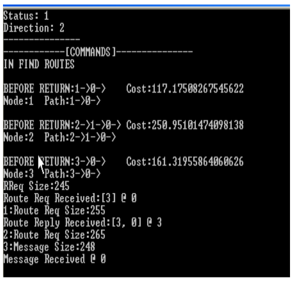


```
Direction: 2
----------------
DuplexLink:
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
----------------
DuplexLink:
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
----------------
DuplexLink:
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
----------------
DuplexLink:
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
SimplexLink: Bandwidth 5000000 bps, Delay 0.0000 seconds
Error rate: 0.0
----------------
DuplexInterface: 192.168.0.1
SimplexInterface: 192.168.0.1
Status: 1
Direction: 1
SimplexInterface: 192.168.0.1
Status: 1
Direction: 2
----------------
DuplexInterface: 192.168.0.3
SimplexInterface: 192.168.0.3
Status: 1
Direction: 1
```

Fig6.6& Fig6.7: connection of network , Bandwidth&Time delay shown in trace file



```
----------------
-------------[COMMANDS]---------------
RReq Size:270
Data Req Received:[1] @ 2
Freq of Data Item :0 is 1
2:Data Req Size:359
Handling Event
In Receive Event
Source Address  Value:192.168.0.2
Destination Address     Value:192.168.0.3
Packet ID       Value:0
Data Req Received:[1, 2] @ 0
Data Req Processed @ 0(Server)
Data Item Id:0
Data:hii
 @ Source
Data Req Received:[1] @ 3
Freq of Data Item :0 is 1
3:Data Req Size:359
Handling Event
In Receive Event
Source Address  Value:192.168.0.3
Destination Address     Value:192.168.0.1
Packet ID       Value:0
Handling Event
In Receive Event
Source Address  Value:192.168.0.2
Destination Address     Value:192.168.0.4
Packet ID       Value:1
Handling Event
In Receive Event
Source Address  Value:192.168.0.1
Destination Address     Value:192.168.0.3
Packet ID       Value:0
Handling Event
In Receive Event
Source Address  Value:192.168.0.3
Destination Address     Value:192.168.0.4
Packet ID       Value:1
Data Req Received:[1, 2] @ 3
Data Reply Received:[1, 2, 0] @ 1
Handling Event
In Receive Event
Source Address  Value:192.168.0.3
```



```
Status: 1
Direction: 2
----------------
-------------[COMMANDS]---------------
IN FIND ROUTES

BEFORE RETURN:1->0->    Cost:117.17508267545622
Node:1  Path:1->0->

BEFORE RETURN:2->1->0-> Cost:250.95101474098138
Node:2  Path:2->1->0->

BEFORE RETURN:3->0->    Cost:161.31955864060626
Node:3  Path:3->0->
RReq Size:245
Route Req Received:[3] @ 0
1:Route Req Size:255
Route Reply Received:[3, 0] @ 3
2:Route Req Size:265
3:Message Size:248
Message Received @ 0
```

Fig6.8: Source and Destination address Fig6.9 DistanceVectorRouting

VII. PERFORMANCE EVALUATION

The goal is to evaluate the performance of a MANET with the proposed caching scheme.

As performance metrics, we adopted the following measurements:

□ Traffic – The amount of traffic that each mobile node in the network has to process because the node generates the packets or because the packets have to be forwarded.

□ Hops – It includes the request from the requester to the node that serves the document and back again to the requester node.

□ Delay - Defined as the time elapsed between a document request and the reception of the corresponding reply.

□ Percentage of timeouts - Defined as the proportion of requests that must be retransmitted again because the reply does not reach the destination before the timeout is reached.

□ Local hit ratio – It is the ratio between the number of documents served by the local cache and the total number of documents requested by each node.

□ Remote hit ratio – It is the ratio between the number of documents served by a node that is not a server and the total number of documents requested by each node.

(A)Effect of Network Load:

Figure7 represents the mean traffic processed by the nodes. Figure 7.a shows that the traffic generated in the scenario using RWP is greater than that using MG. This is caused by the AODV broadcast messages employed to create the routes between the mobile nodes. In Figure 7.b we can observe that as the periodicity of document requests increases, the delay is also augmented. As the time between requests increases, the number of documents expired in the nodes' local caches is also increased and the documents in the local caches are less updated. This can be observed in Figure 7.e where the cache hits decreases as the network load decreases.The route TTL configured in AODV is ten seconds and hence the network with a mean time between requests less or equal to this time will take advantage of the already created routes while greater time between requests will have to create the routes again. However, Figure 7.c shows that under RWP nodes need less hops to obtain the documents than under MG although the difference declines as the number of blocks increases. Finally, Figure 7.d shows that the number of timeoutsis diminished as the network traffic decreases (the mean time between requests increases) until 25 seconds between requests but for 50 seconds between requests the number of timeouts is increased.

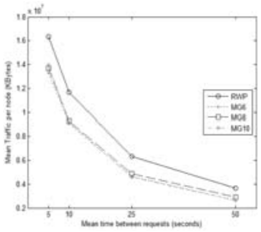


Fig7 (a) mean traffic

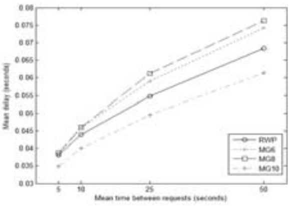


Fig7 (b) mean delay

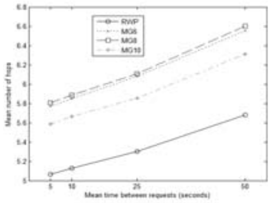


Fig7(c) mean hops

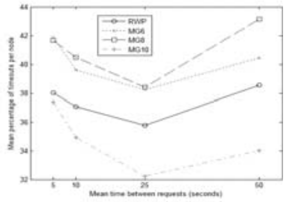


Fig7 (d) percentage of timeouts

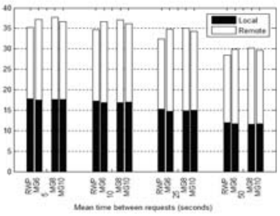


Fig7 (e) cache hits as a function of the mean time between requests

(B)Comparisons between AODV and DSR Routing Protocols

The major difference between AODV and DSR stems out from the fact that DSR uses source routing in which a data packet carries the complete path to be traversed. However, in AODV the source node and the intermediate nodes store the next hop information corresponding to each flow for data packet transmission. AODV uses a destination sequence number (DestSeqNum) to determine an up-to-date path to the destination. A node updates its path information only if the DestSeqNum of the current packet received is greater than the last DestSeqNum stored at the node. AODV does not allow to handle unidirectional links. Unidirectional links can be handled in DSR.

VIII. CONCLUSIONS

This paper presents a caching scheme for Mobile AdHoc Networks that implements a local cache in each mobile node of the network. We can conclude that the mobility model used to evaluate a caching scheme clearly influences the obtained performance results of the network. We have compared proposed caching schemes using routing protocols. Hybrid Cache takes advantage of Cache Data and Cache Path while avoiding their weaknesses. Simulation results showed that the proposed schemes can significantly reduce the query delay when compared to Simple Cache and significantly reduce the message complexity.

As a future research, one can evaluate the proposed caching scheme on more mobility models, and the impact of mobility, cache size, catch maintenance, attacks, type of communication to be enhanced.

IX. REFERENCES

[1] S. Das, C. Perkins, and E. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," *IEEE INFOCOM*, pp. 3–12, 2000.

[2] J. Broch, D. Maltz, D. Johnson Y. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *ACM MobiCom*, pp. 85–97, October 1998.

[3] C. Perkins, E. Belding-Royer, and I. Chakeres, "Ad Hoc On Demand Distance Vector (AODV) Routing," *IETF Internet draft, draft-perkinsmanet aodvbis-00.txt*, Oct. 2003.

[4] S. Das, C. Perkins and E. Royer, Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks, *IEEE INFOCOM* (2000), 3–12.

[5] L. Yin and G. Cao, Supporting Cooperative Caching in Ad Hoc Networks, *IEEE INFOCOM* (March 2004), 2537–2547

[6] C. Perkins and P. Bhagwat, Highly Dynamic Destination Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, *ACM SIGCOMM* (1994), 234–244.

[7] ns Notes and Documentation, http://www.isi.edu/nsnam/ns/, 2002.

[8] R. Agu¨ ero and J.P. Campo, "Adding Multiple Interface Support in NS-2," Jan. 2007.

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

[10 A. Raniwala and T. Chiueh, "Architecture and Algorithms for an IEEE 802.11-Based Multi-Channel Wireless Mesh Network," Proc. IEEE INFOCOM, 2005.

[11] B. Tang, H. Gupta, and S. Das, "Benefit-Based Data Caching in Ad Hoc Networks," IEEEE Trans. Mobile Computing, vol. 7, no. 3, pp. 289-304, Mar. 2008.

[12] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Network," Mobile Computing, pp. 153-181, Kluwer Academic Publishers, 1996.