An Opportunistic Cross Layer Multi Path Routing Protocol to Improve TCP Performance Over Mobile Ad Hoc Networks

Dr.K.Kathiravan
Professor & *Head, Dept of CSE*
Easwari Engineering College
Chennai, India
kathir.cse@srmeaswari.ac.in

R.Radha
Asst Professor, Dept of IT
Easwari Engineering college
Chennai, India
radha.r@srmeaswari.ac.in

***Abstract*— Multi path routing protocol such as AOMDV establishes multiple link disjoint paths between source and destination to improve the reliability, fault tolerance and thereby TCP performance over multi hop MANETs. But, TCP misinterprets the frequent link failure and wireless channel errors as congestion and slow starts the window size, even during the presence of alternate paths in the routing table. In this paper, we analyze the TCP misbehavior and propose cross layer mechanisms that effectively utilizes AOMDV. Our protocol finds out multiple link disjoint paths using AOMDV but next hop selection is done based on current wireless channel condition to reduce wireless channel errors. During primary route failure, we make the TCP source to select a new window size for alternate path. This opportunistic selection of window size is based on the distance between the origin of alternate path and destination. We study the performance of our mechanisms using NS2 and evaluate them individually against AOMDV in terms of TCP throughput, window size, packet delivery ratio and BER and shows significant improvement.**

Keywords---Cross layer, AOMDV, TCP, MANET.

I. INTRODUCTION

Mobile Ad hoc Network(MANET) is a collection of autonomous, self configured, self organized and highly dynamic wireless mobile devices. This network is helpful in any situation where instantaneous network connectivity is needed such as flood recovery, disaster relief and battle field communication. Much more research work has been carried out on developing MAC, Network and Transport layer protocols on this type of network. Routing protocols are responsible for finding the route and forwarding data between computers. These protocols face many challenges due to the dynamic nature of MANET. Nowadays multi path routing protocols draw more attention than single Path routing due to its advantage of improving throughput, end to end delay and reliability[1].

Multi path routing protocols establish multiple paths between a source destination pair. The established path can be node disjoint or link disjoint. Node disjoint paths do not have any nodes in common except the source and destination hence they do not have any links in common. But, link disjoint paths do not have any links in common. However, they may have common nodes. Among the multiple paths, many protocols[2,3,4,5,6] use one path as primary and remaining as secondary (backup paths). The backup paths are useful when primary path fails. These protocols provide better fault tolerance by performing faster and efficient recovery from route failure. In [3], Mahesh et al. proposed the earliest protocol which finds available link disjoint loop free paths between source and destination. In [2], Jiwon Park et al. finds out the multiple paths based on SINR during route discovery. Xiaoqin Chen et al. discovers the next hop based on channel fading duration in [4]. Tekaya et al. finds the paths based on maximum nodal remaining energy in [5]. In [6], authors find out routes based on the congestion information.

Multiple paths can also provide load balancing by distributing the data packets across multiple disjoint paths[7,8,9]. In [7], Murakami et utilizes load balancing to overcome congestion and frequent link failure. In [8], Lu Xuechao et al. distributes traffic among multiple paths based on load state of the network. Venkatasubramanian et al. performs load balancing and provides QOS support in [9]. But, these protocols suffer from out of order delivery of TCP packets which is misinterpreted as congestion loss by TCP source that leads to reduced throughput [11]. In [10] author specifies that using multiple paths simultaneously may actually degrade the TCP performance comparing alternate path routing due to the out of order delivery of TCP packets and frequent link failure.

In this paper, we propose two different cross layer mechanisms which is based on AOMDV. Mechanism-I finds out multiple link disjoint paths and stores in a routing table, but selects the next hop based on current wireless channel condition. Every node dynamically measures the signal strength of incoming packets and updates SNR . Each node dynamically measures the channel condition to reduce the wireless channel errors on transmission. In Mechanism- II, the intermediate node detects link failure on mobility and checks whether alternate paths are available and it notifies the source about alternate path and freezes its window size. The TCP window size for new path should reflect the congestion status of the new path so, we calculate the window size based on the distance between new path and destination and intimates the source about new window size through TCP acknowledgement.

The rest of the paper is organized as follows, Section II briefly explains the predecessor protocols. Section III describes the TCP misbehavior on AOMDV protocol during route failure and section IV describes the system architecture and presents the mechanisms used for next hop selection and window size

calculation. In Section V, we discuss the results and we conclude and present the future work of the paper in section VI.

II. RELATED WORK

We propose a new cross layer Mechanism- that extends the Ad hoc On-Demand Multi path Distance Vector (AOMDV) routing protocol [3]. In this section, we review the details of these two predecessor protocols that are useful to our discussion in this paper.

A. AODV

Ad hoc On-Demand Distance Vector (AODV) is a reactive single-path, on-demand routing protocol. When a source node A generates a packet for a particular destination node B, it broadcasts a route request (RREQ) packet. The RREQ contains the following fields:

<source IP address, source sequence number, broadcast ID, destination IP address, destination sequence number, hop-count>

Source sequence number is a monotonically increasing factor which determines freshness of a packet. destination sequence number is the last known sequence number for B at A and hop-count is incremented at each intermediate node which processes the RREQ. A RREQ is uniquely identified by the combination of source sequence number and broadcast ID. An intermediate node only processes a RREQ if it has not received a previous copy of it. If an intermediate node has a route to B with destination sequence number at least that in the RREQ, it returns a route reply (RREP) packet, updated with the information that it has. If not, it records the information available in RREQ and forwards the RREQ to its neighbors. Like the RREQ, a RREP is only processed on first copy. The RREP packet contains the following fields:

<source IP address, destination IP address, destination sequence number, hop-count, route expiration time>

The route expiration time is the time after which the route is considered to have expired and a new route discovery process must be undertaken. A sends packets via the first path it hears about. When an active route link breaks, a route error (RERR) packet, with sequence number incremented from the corresponding RREP and hop-count of 1, is sent by the upstream node of the broken link to A . Upon receipt of a RERR, A initiates a new route discovery process if it still has packets to send to B . Nodes also periodically send hello messages to neighboring nodes to maintain knowledge of local connectivity.

B. AOMDV

The key difference of AOMDV over AODV is that it provides multiple paths to B . These paths are loop free and mutually link-disjoint. AOMDV uses the notion of advertised hop-count to maintain multiple paths with the same destination sequence number. In both AODV and AOMDV, receipt of a RREQ initiates a node route table entry in preparation for receipt of a returning RREP.

In AODV, the routing table entry contains only one route to each destination. But in AOMDV, the routing table entry is modified to contain multiple route entries and multiple loop-free paths.

Single next-hop id is replaced by a list of all next-hop nodes and corresponding hop-counts of the available paths to B from that node. To obtain link-disjoint paths in AOMDV, B can reply to multiple copies of a given RREQ, as long as they arrive via different neighbors.

III. TCP MISBEHAVIOR ON ROUTE FAILURE IN AOMDV.

TCP (Transmission Control Protocol) is a connection oriented transport layer protocol that provides reliable in order delivery of data packets to the destination. If we use TCP without any modification on MANET, it results in serious drop of throughput due to the following reasons i) Un predictable nature of wireless medium – High Bit Error Rate ii)unexpected mobility of nodes – Route Failure iii)High contention for wireless medium. Here we analyze the behavior of TCP during route failure on AOMDV protocol.

The route failure may lead to the selection of alternate path, and the data packets from source or intermediate nodes and acknowledgements from destination have to be rerouted using the new alternate path. But before this happens, time out happens in TCP source which assumes the event as congestion and reset its window size to 0 which reduces the throughput unnecessarily. To overcome this issue, the neighbor which detects the route failure should send notification to TCP source to freeze the congestion window until it receives acknowledgement for the new data packet that is rerouted through the alternate path. The old congestion window may not reflect the correct window size for the new path. So, we develop a mathematical model that reflect the new window size based on the distance between origin of alternate path and destination. It is then fed back to the TCP source through the acknowledgement.

We simulate the following network topology of Fig. 1 in ns2 using AOMDV (link disjoint mode) as routing protocol. AOMDV selects 0-1-2-6 as primary path and we run the simulation for 100 seconds. During simulation, node 1 moves towards node 0 at 20 meters per second speed at 60 th second which results in primary route failure. Now the data packets forwarded by node 1 or acknowledgements forwarded by node 2 are discarded after seven MAC layer retransmissions. These seven retransmissions do not occur continuously due to intra flow contentions. After this event node 2 sends RERR message to no node 6 and node 6 finds alternate path and reroutes acknowledgements. Before this happens, TCP timeout occurs for the lost data packets and misinterprets the event as congestion loss and makes TCP to enter slow start state that drops the throughput unnecessarily in the presence of alternate path.

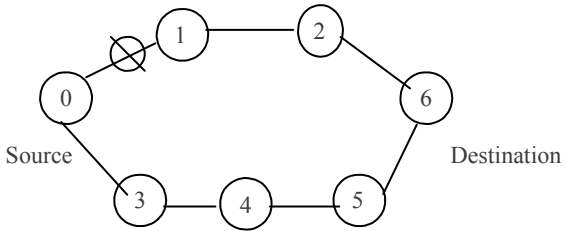

Figure 1.Network Topology illustrating Route Failure

The drop in throughput and congestion window size is plot as a graph in the following Fig 4 and Fig 3. AOMDV finds out multiple link disjoint paths to improve fault tolerance and reliability. Due to TCP misbehavior, It is not been utilized effectively in many cases as above. The above situation is illustrated using Fig 2.

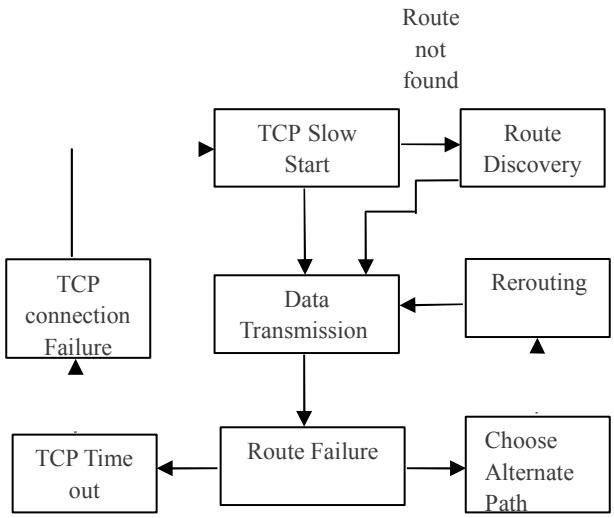

Figure 2.Events During Route Failure

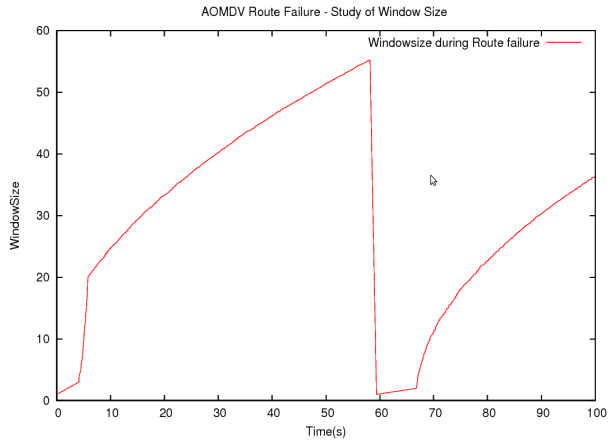

Figure 3.TCP Window Size on Route Failure

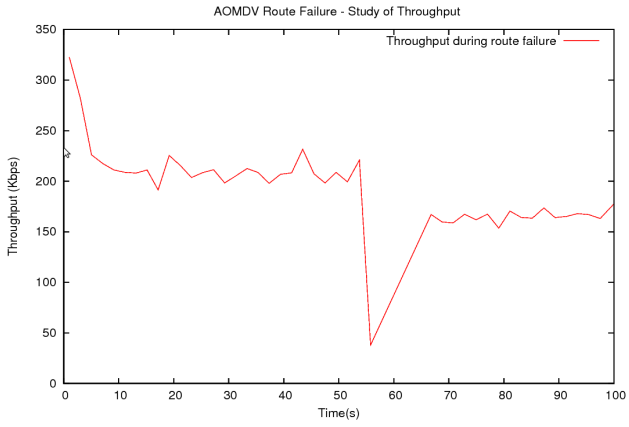

Figure 4.TCP throughput on Route Failure

IV. SYSTEM MODEL.

In this paper, we propose two cross layer mechanisms that improve TCP performance during wireless channel errors and route failure respectively. The overall system model is diagrammatically represented in Fig. 5 We utilize AOMDV for finding possible link disjoint routes between source and destination. But it does not consider the wireless channel condition which results in frequent link failure leading to misinterpretation of route failure and deletion of valid routes in the routing table. We propose a Mechanism- I which selects the next hop and link rate based on the current channel condition. We also find out the TCP window size for alternate path during route failure using Mechanism- II. This opportunistic window size calculation is done by intermediate node which detects the link failure. The new window size is fed back to the source using TCP acknowledgement. The methodologies are explained in detail below.

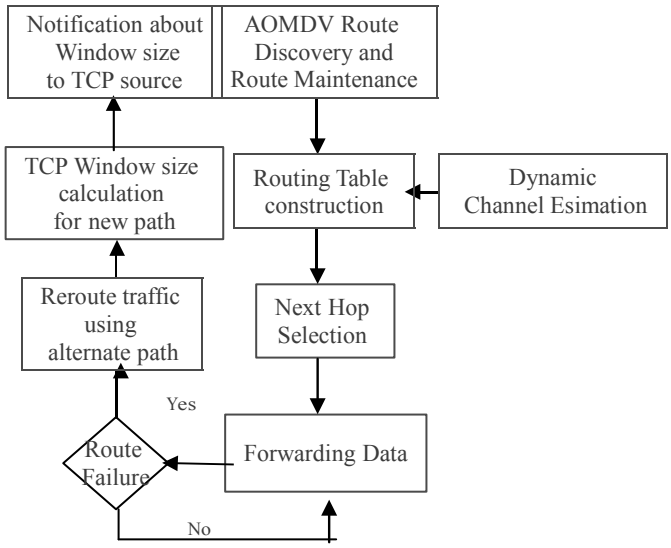

Figure 5.System Model

A. *Methodology I a. Next hop selection*

Next hop selection is done using a greedy strategy i.e. We select the local optimum wireless link for transmission at each stage with the hope of finding global optimum end to end path. To enable next hop selection based on wireless channel condition, every node contains an additional field Bit Error Rate(BER) for every possible next hop in each route list of the routing table. BER is calculated periodically. Every node measures the cumulative Signal to Noise ratio of the wireless link while receiving each HELLO frame and is reset periodically. A timer is used to update the BER value in the routing table. When the timer expires, Bit Error rate can be derived from measured Signal to Noise Ratio[11]. For a given SNR and a given channel code rate, a BER value is obtained using a lookup table and linear interpolation. It also gives the relationship derived theoretically. SNR is calculated as follows.

$$SNR = 10 \log(RxPower)/(Noise + \sum RxPower(i)) \quad (1)$$

Rx_Power is the frame signal strength at the receiver. It is calculated by Two ray ground propagation model.

Rx _ Power(i) is the signal strength of other frames at the node.

Noise_ should be calculated from the receiver sensitivity of the data rate used by the frame.

i ranges from 1 to n frames

Bit Error rate is chosen as the primary metric and hop count as the secondary metric. Before forwarding the data, the node selects the next hop whose current BER is less. If two routes have same BER then the next hop with shortest hop count is chosen.

B. Methodlogy II. Window Size Calculation

Congestion window in TCP imposes an acceptable data rate for a particular connection based on congestion information that is derived from time out events as well as from duplicate acknowledgements. The route failure may lead to the selection of alternate path, and the data packets from source or intermediate nodes and acknowledgements from destination have to be rerouted using the new alternate path. We may lose the relationship between congestion window size and tolerable data rate for the new path. To fulfill the requirement we calculate the new window size using a simple method. Our method finds out whether the origin of the chosen alternate path is closer to the destination or far. The reason for taking this distance as a factor and its effect is discussed with example for 3 different scenarios.

The intermediate node which has alternate path is the one which is going to advertise the congestion window size factor through TCP acknowledgements. So the link layer of the intermediate node must be TCP aware. The intermediate node find out the number of hops traveled using IP header TTL field and it takes the hop count from its routing table. The detailed pseudo code for the window size calculation is shown in Algorithm 1.

1) Scenario I

In this scenario of Fig 6. Let us assume that the primary path is 0-1-3-5-6-8. During transmission, we assume that node 6 moves away which leads to the selection of alternate path at origin node 5. Here the distance from origin node of alternate path (5) to destination (8) is 2 which is less comparing the no of hops forwarded(3). So, we propose to take the new congestion window as ¾ th of the old congestion window. In this case, The path 0-1-3-5 (which has 3 hops) is also used for new path. The old congestion window could have taken the congestion that happened in 5-7-8 also because node 5 is common for both old and new paths. In this case, we conclude that the new congestion window can be approximately equal to old congestion window.

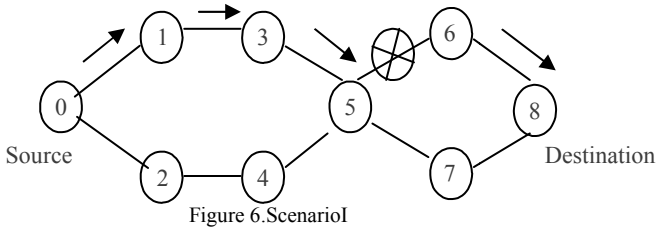

Figure 6.ScenarioI

2) Scenario II

In this scenario of Fig 7 the primary path is assumed to be 0-1-3-5-6-8. and we find that 3-5 fails during transmission. Node 3 will send Route Error message to node 0 which has to reroute the traffic using 2. When the origin of alternate path is closer to source or source itself, we prefer to have the new window size as 25% of old window size. Here, the alternate path may be entirely or partly different from old path. But we choose to have the new window as 25% because the source node and destination node is common for both the paths and the congestion caused in the first two hops will reflect in each others window size.

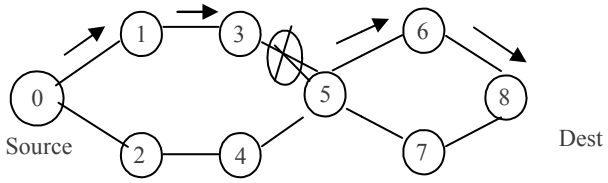

Figure 7.Scenario II

3) Scenario III

In this example the primary path is 10-0-2-9-4-5-7. The link between 9 and 4 fails due mobility of node 4 during transmission. Now, the origin of alternate path is 9 and its distance to destination and from source is same (distance 3)So, we prefer to select the new window size as 50% of the old window size. The network configuration is given in Fig.8 . When the distance between source, alternate path and distance between destination , alternate path is same we can slightly increase the window size because there are

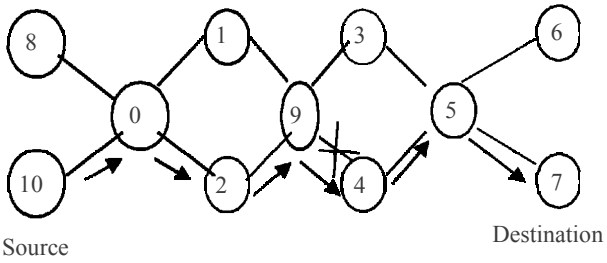

Figure 8.scenario III

The algorithm for window size calculation is given in Algorithm 1. The algorithm is executed by the intermediate node which detects primary route failure and has to choose the alternate path. The primary route failure leads to TCP timeout at source in many cases as explained in the given scenarios. When the alternate path is available in the routing table, this time out may unnecessarily reduce the overall throughput. To overcome this, we choose the TCP window size based on the distance betwwen the origin of alternate path and destination to reflect the congestion status of the new path.

We do our window size calculation in three different ways.For the TCP connection whose hop count is less than or equal to 2 or when the route failure is closer to the destination , the old window size reflects the total congestion of the network. In this case, the old congestion window can be approximately taken as the the new window size.

Algorithm 1 Window Size Calculation

DistanceAP_Dest = Distance between Origin of alternate path and Destination
NoOfHops = No Of hops travelled from source to Origin of alternate path
*SlowStartThreshold**=2
If(*NoOfHops*<*DistanceAP_Dest* || *NoOfHops*<=2)
WindowSize=.75
return *window size* feedback to TCP source
End If
If (*NoOfHops*>*DistanceAP_Dest*)
WindowSize=.5
return *window size* feedback to TCP source
else
window size=24
return *window size* feedback to TCP source
End IF

window size factor 0.75 is stored in TCP acknowledgement and In second classification, where the route failure is near source, Window size factor 0.25 is stored in acknowledgement. Otherwise, 0.5 is taken as factor when the alternate path starts exectly at the middle of the old path.

This factor is multiplied by the old congestion window size preserved during the recent timeout by the TCP source. The new window size is calculated only when the time out occurred due to Route failure which was notified by intermediate node by Route Failure Notification. This prevents TCP source from unnecessarily increasing the Congestion Window Size. Slow start threshold which was wrongly reduced by half during route change is again multiplied by two.

V. RESULTS AND DISCUSSION

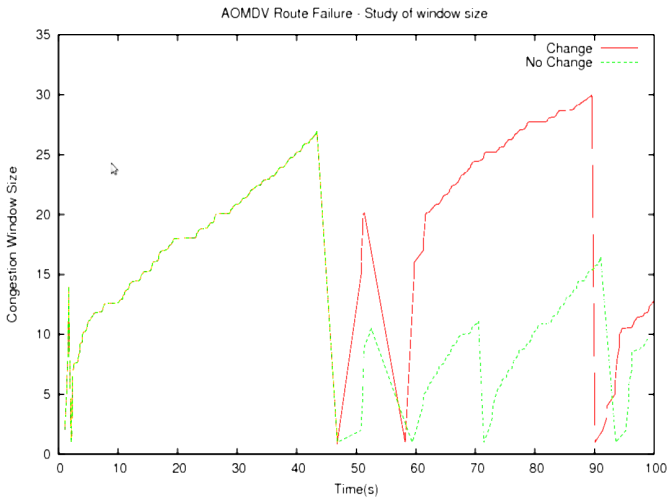

Figure 9.TCP throughput Vs Time

We evaluate the performance of our cross layer mechanism in this section. We simulate the scenario shown in Fig 6 using ns2.35 simulator. We created a TCP connection between node 0 and node 6. We made the MAC layer of intermediate node calculated the multiplication factor for new window size and sent it through the TCP acknowledgement. So our Mac layer must be TCP aware. We purposely made node 6 move towards node 5 at time 40sec. The route failure makes the node 5 to choose alternate path, but before node 5 understands route failure TCP source times out and slow starts and our mechanism finds out suitable window size which improved the window size and thereby throughput based on the proposed algorithm and analyzed the TCP throughput and TCP window size. Our mechanism performs better than AOMDV which is shown in Fig 9 and Fig 10. The protocol with the name change is showing our mechanism and no change showing AOMDV.

In Fig. 11 We created many TCP connections and made the nodes move around randomly for the above topology. We observed the TCP throughput for 5, 6, 7, 8, 9 and 10 connections and we have plotted the throughput against Number of connections. The throughput decreases as the number of connections increase. But our scheme performs better than AOMDV. There is around 25% increase in throughput.

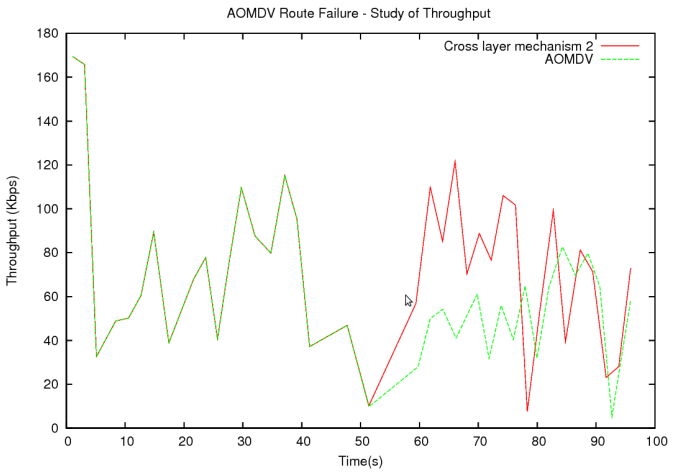

Figure 10.TCP Window Size Vs Time

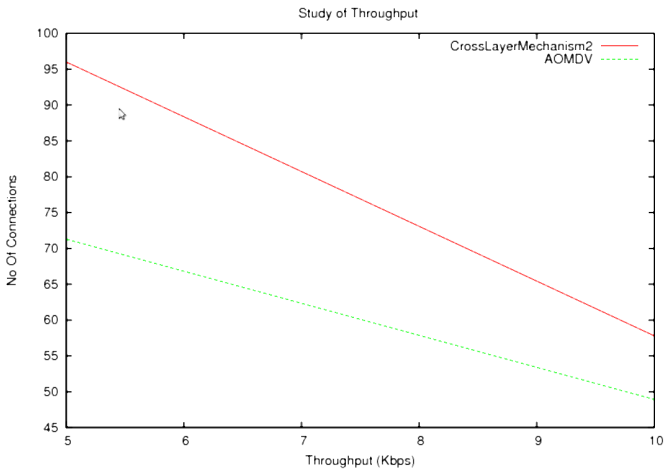

Figure 11.TCP throughput Vs Nof Of Connections with random mobility

In Fig.12 We have shown the performance of AOMDV against various Byte Error Rate. The throughput decreases seriously when the Byte Error Rate increases. The reason for this drop is that it introduces many wrong route failures which leads to unnecessary route discoveries. We have simulated the

topology of Fig 1. We introduced Uniform Error Model intentionally for the primary path 0-1-2-6 and our mechanism selected the primary path as 0-3-4-5-6 and even though the hop count is larger, our mechanism works better because it makes the next hop selection based on the current Byte Error Rate.

The range for the Byte Error Rate in the Uniform Error Model from .000001 to 0.001 observed the TCP performance as below.

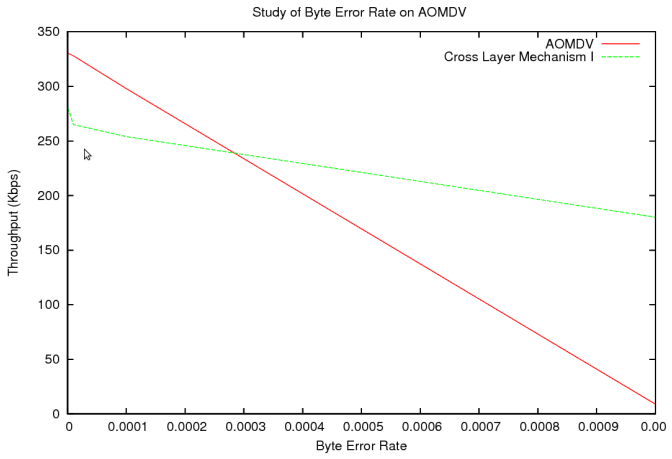

Figure 12.TCP throughput Vs BER

VI. CONCLUSION

In this paper, we proposed two different cross layer mechanisms which is based on AOMDV. Mechanism I finds out multiple link disjoint paths and stores in a routing table, but selects the next hop based on current wireless channel condition from physical layer. Every node dynamically measures the signal strength of incoming packets and updates SNR . Each node dynamically changes the link rate based on channel condition to reduce the wireless channel errors on transmission. In mechanism II, the intermediate node detects link failure on mobility and checks whether alternate paths are available and it notifies the source about alternate path and freezes its window size. The TCP window size for new path should reflect its congestion status. so, we calculate the window size based on the distance between new path and destination and intimates the source about new window size through TCP acknowledgement. We analyzed the performance individually against AOMDV and showed significant improvement.

REFERENCES

[1] Christos Tachtatzis , David Harle, "Performance Evaluation of Multi-path and Single-path Routing Protocols for Mobile Ad-Hoc Networks" International Symposium on Performance Evaluation of Computer and Telecommunication Systems, 2008. (SPECTS 2008) pp 173-180

[2] Jiwon Park, Sangman Moh, and Ilyong Chung, "A Multipath AODV Routing Protocol in Mobile Ad Hoc Networks with SINR-Based Route Selection" IEEE International Symposium on Wireless Communication Systems 2008 (ISWCS'08) pp 682-686

[3] Mahesh K. Marina and Samir R. Das, "Ad hoc on-demand multipath distance vector routing" Wireless Communications & Mobile Computing - Wireless Ad Hoc Networks: Technologies and Challenges, Volume 6 Issue 7, November 2006 ; pp 969–988

[4] Xiaoqin Chen, Haley M. Jones, and Dhammika Jayalath, "Channel-Aware Routing in MANETs with Route Handoff" IEEE transactions on mobile computing vol. 10,no. 1,January 2011 pp 108-121

[5] Tekaya, M, Tabbane, N., Tabbane, S, "DRE-AOMDV:Delay Remaining Energy for AOMDV Protocol " International Conference on Wireless Communications, Networking and Mobile Computing(WiCOM),2011 pp 23-25

[6] Wannawilai,P,Sathitwiriyawong,C."AOMDV with Sufficient Bandwidth Aware " International Conference on Computer and Information Technology(CIT),2010 June29-2010-July-1-2010 pp 305 - 312

[7] Murakami.T,Sasase,Bandai.M."SMR-LB:Split multi-path routing protocol with load balancing policy to improve TCP performance in mobileadhocnetworks"International Symposium on Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005. on 11-14 Sept.2005 pp 1424 - 1428 Vol. 3

[8] Lu Xuechao;Xu Chunxiu;Wu Muqing;Zhen Yan;Wu Dapeng; "Design and Realization of a Novel Multi-path Load-Balancing Routing Protocol in Ad Hoc Network"International Conference on Information Engineering, 2009. ICIE '09. WASE on 10-11 July 2009 pp 247 - 250

[9] Venkatasubramanian, S. Gopalan, N.P "A QoS-based robust multipath routing protocol for mobile ad hoc networks" International Conference on Internet, 2009. AH-ICI 2009. on 3-5 Nov. 2009 pp 1 – 7

[10] Haejung Lim, Kaixin Xu, Gerla, "TCP Performance over Multipath Routing in Mobile Ad Hoc Networks " IEEE International Conference on Communications, 2003. ICC '03. on11-15 May 2003 pp1064 - 1068 vol.2

[11] Gavin Holland, Nitin Vaidya "Analysis of TCP Performance over Mobile Ad Hoc Networks " Kluwer Academic Publishers Wireless Networks 8, pp 275–288, 2002