International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 2– No.4, May 2012 – www.ijais.org

TCP Congestion Control through Bandwidth Estimation Mechanism in MANET

Ramratan Ahirwal
Computer Science & Engineering
Samrat Ashok Technological Institute
Vidisha (M.P.) 464001 India

Ganesh Lokhande
Computer Science & Engineering
Samrat Ashok Technological Institute
Vidisha (M.P.) 464001 India

Yogendra Kumar Jain
Computer Science & Engineering
Samrat Ashok Technological Institute
Vidisha (M.P.) 464001 India

ABSTRACT
Mobile ad-hoc network is collection of temporary nodes that are capable of forming dynamic temporary network, self organize, and infrastructure less with nodes contains routing capability, improving the performance of Transmission Control Protocol (TCP) associated with the presence of multi-hop MANET is one of the research challenges in wireless mesh networks. Wireless mesh networks have large round trip time variations and these variations are dependent on the number of hops. The end-to-end TCP throughput degrades rapidly with increase in number of hops. This is one of the biggest problems of TCP over wireless mesh networks In this paper we present method for avoid congestion in MANET environment using bandwidth estimation technique .In our approach we use acknowledgement time intervals for bandwidth estimation in TCP flow, in our scheme we monitor the time spacing between received acknowledgements (ACKs) at the sender node and find out the available bandwidth of the connection between sender to destination, if available bandwidth is less then the actual data size so we decrease the data size according to available bandwidth and avoid congestion, and same time other sender node try to communicate any destination through same available path which is used by previously senders so new sender node and all other previously communicated sender node sends data according to available bandwidth of the intermediate nodes so we increase the network performance and provides congestion free communication.

Keywords
Mobile ad hoc network, TCP, Hybrid, Congestion, Congestion control, Reno, NewReno, Vegas.

1. INTRODUCTION
In the network, the wireless communications provide many benefits to organizations and users such as portability and flexibility, increased productivity, and lower installation costs. Wireless local area network (WLAN) devices, for instance, allow users to move their laptops from one place to another place within their offices without the need for wires and without losing network connectivity. Less wiring means greater flexibility, increased efficiency, and reduced wiring costs. Ad hoc networks, for example those enabled by Bluetooth, allow data synchronization with network systems and application sharing between devices. Handheld devices like personal digital assistants (PDA) and cell phones allow remote users to synchronize personal databases and provide access to network services such as wireless e-mail, Web browsing, and Internet access. To moving bulk data quickly over high-speed data network is a requirement for many applications. These applications require high-bandwidth links between network nodes. In 1981, for maintain the stability of Internet all applications should be subjected to congestion control. Introduce by author J.Postel, TCP [2] is well-developed, extensively used and widely available Internet transport protocol. Unfortunately, TCP has many performance problems when operated over multi-hop wireless mesh networks. In 2010, the principal problem of TCP lies in the congestion control mechanism proposed by author Prasanthi. S and Sang-Hwa Chung "An Efficient Algorithm for the Performance of TCP over Multi-hop Wireless Mesh Networks[1] It uses three duplicate acknowledgements and retransmission timeout to detect packet loss and treats every packet loss as a congestion indication. Moreover, the TCP sender drastically reduces its congestion window to Slow Start state due to frequent retransmission timeout, which degrades the throughput of TCP.

Retransmission timeout: Retransmission time out is basically error message, retransmission time out means if our data packet sends to destination but didn't receive all of the data from you in time to make a continuous transmission the important tasks that a TCP connection must perform is to estimate a proper value of the retransmission time out (RTO).

The RTO is calculated based on the following formula gives in equation (1)

$$RTO = SRTT + RTTVAR \quad \text{.....} \quad (1)$$

Where SRTT (Smoothed Round Trip Time) is a low pass filtered version of the round trip sample (RTTm) measured by the TCP sender. RTTVAR (Round Trip Time Variation), instead, represents an estimate of the standard deviation of the round trip sample, and it is calculated by low-pass filtering the quantity | SRTT – RTTm |.

Recently work has focused on the problems associated with TCP performance in the presence of wireless links and ways to improve its performance. The performance degradation in wireless mesh networks throughput as a TCP connection traverses a large number of wireless hops is now a major research issue [3, 4]. Our work focused on the performance of TCP over Multi-hop wireless mesh network where the retransmitted packets are loss by retransmission timeout. We designed TCP scheme called a TCP Updated New Reno in which we check bandwidth of the link and according to bandwidth we send data packet to actual destination.

The paper is organized as follows: Section 2 Review work, Section 3 Provide Proposed scheme, 4 Simulation And Result, Section 5 about the Demonstration, Section 6 Conclusion and Future Work, Section 7 References.

2. REVIEW WORK

In this section, we focus on the function of TCP congestion control algorithm in which we adjust the rate of transmitted packets with which the protocol sends packets into the network using a congestion control window. A good congestion algorithm can fully utilize the bandwidth while avoiding over-driving the network and thereby creating packet losses. Since the congestion control mechanism has been introduced in TCP by BSD UNIX, several TCP congestion control algorithms are proposed. On a high level, existing TCP congestion algorithms can be divided into three main categories based on the input of the control mechanism - namely Loss-based TCP, Delay-based TCP and Hybrid TCP. In 2004, Loss-based TCP includes the original TCP Reno proposed by author L. Xu, K. Harfoush, and I. Rhee, TCP BIC [6], in 2003, TCP CUBIC, High Speed TCP [7], and Scalable TCP [8], etc. Among these Loss-based TCP variants, TCP Reno and TCP CUBIC are widely deployed as standard TCP algorithms and default TCP of Linux respectively. Using packet loss as the symptom for network congestion, in Loss based TCP decrease the congestion control window when packet losses occur and increases the window otherwise. In 1994, Delay-based TCP including TCP Vegas proposed by author L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: new techniques for congestion detection and avoidance the three key techniques employed by Vegas, and presents the results of a comprehensive experimental performance [9] and in 2006 by the author D. Wei, C. Jin, S. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," [5] uses the queuing delay as the symptom of congestion. The queuing delay is defined as the difference between the RTT and the propagation delay, i.e. time actually required for a packet to be transmitted from the sender to the receiver.

The Delay-based TCPs are more resilient to transient changes of network conditions such as random packet losses and are also suitable for high BDP networks introduce by author S. H. Low, L. L. Peterson, and L. Wang, "Understanding TCP Vegas: a duality model" [10]. The down side of the approach on the other hand is that, because increasing in queuing delay doesn't necessarily immediately lead to packet loss (due to buffers), when Delay-based TCP stations share the same bottleneck with Loss-based TCP stations, between the time when delay starts to increase and packet loss occurs, the window for the Delay-based TCP will decrease while that for the Loss-based TCP will not, leading to bandwidth "starvation" for the Delay-based stations.

TCP Reno uses the TCP's AIMD mechanism of increasing the congestion window W by one segment per round-trip time for each received ACK of send packet and halving the congestion window for each loss event per round-trip time. TCP Reno maintains the congestion window as follows:

Increase:

$$W = W + 1/W \ldots\ldots \ldots (2)$$

Decrease:

$$W = W - 1/2W \ldots\ldots\ldots (3)$$

When the bandwidth of intermediate nodes does not change, TCP Reno periodically repeats the window increase and decrease. TCP Reno's congestion window in terms of packet loss rate (p) is calculated as:

$$W_{reno} = 1.22/p^{0.5} \ldots\ldots (4)$$

In equation (4), TCP Reno places a serious constraint on the congestion window that can be achieved by TCP in realistic environments. For example, In TCP Reno connection with 1500-byte packets and 100ms RTT, to achieve a steady-state throughput of 1Gbps would require an average congestion window of 8300 segments, and an average packet loss rate of $2 \times 10{-}8$. This requirement is unrealistic in current networks. The congestion window requires more than 4000 RTT to recover after a loss event which prevents efficient use of the link bandwidth. TCP requires extremely small packet loss rate to sustain a large window which is not possible in real life networks proposed by author Habibullah Jamal, Kiran Sultan ," Performance Analysis of TCP Congestion Control Algorithms" [12]. TCP Reno perform very well over TCP when packet losses are small but in case of multiple packet losses in one window then Reno doesn't perform very well .

NewReno solve this multiple packet losses problem. New Reno is capable detect multiple packet losses Therefore is much more efficient than Reno in the case of multiple packet losses. MANET suffers from multiple packet loss whenever nodes in the network change their position during communication. When nodes move from their position, they may be unapproachable for a long time. In this case a number of packets meant for those nodes may get lost. TCP-New Reno proposed by author Prasanthi. S and Sang-Hwa Chung "An Efficient Algorithm for the Performance of TCP over Multi-hop Wireless Mesh Networks [1] has advantage of its strategy to detect and handle multiple packet loss thereby avoiding continuous retransmission timeouts. It enters into fast-retransmit phase when it receives multiple duplicate packets .but it does not exit this phase until all the packets which were out standing at the time it entered fast recovery get acknowledged. Thus it overcomes the problem of reducing the congestion window (CWND) multiples times by avoiding retransmission timeout. Fast recovery phase allows for multiple re-transmissions for single timeout. After entering into this phase it checks the maximum segment which is outstanding. If all the segments which were outstanding are acknowledged then it exits the fast recovery phase and sets CWND to ssthresh and continues congestion avoidance. It works in following four stages:

Slow Start (SS): In Slow Start state, in initial the cwnd is initialized to one packet after receiving acknowledgement it cwnd is set one to two. This can be estimated as an exponential growth .When congestion window is greater than ssthresh value then sender moves to Congestion Avoidance state.

Congestion Avoidance*:* In the Congestion avoidance phase, when three duplicate ACKs are received, the sender enters fast-retransmit state.

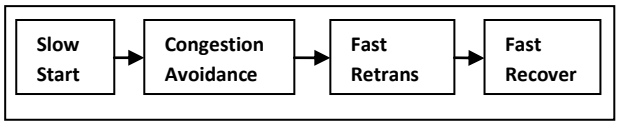

Figure 1 TCP Newreno Congestion Avoidance Steps

Fast Retransmit*:* In this stage the lost packet is retransmitted and ssthresh is set to maximum of FlightSize/2 or 2*SMSS, where SMSS means sender's maximum segment size and FlightSize means amount of data that is unacknowledged at any given time. Fast Recovery: In this stage, CWND is increased by one for every duplicate acknowledgment and a new packet is transmitted, if allowed by CWND. It is assumed that in fast-recovery state, sender resets the retransmission timer upon receiving a partial ACK.

TCP NewReno perform well over multiple packet loss but NewReno suffer from the fact that it's take one RTT to detect each packet loss when acknowledgement for first retransmitted segment is received only then can we deduce which other segment was lost. To solve this problem we use available bandwidth estimation technique in our updated TCP NewReno scheme.

3. PROPOSED SCHEHE

In our proposed scheme we use acknowledgement time intervals to estimation available bandwidth. This available bandwidth estimation helps to avoid congestion in the network. The available bandwidth is the amount of the share of the bandwidth that the flow can occupy without affecting other flows. In case of fully congested situation over intermediate path node, the delay of acknowledgments reception per estimated time period at the sender node can be affected to the fair share of bandwidth. In a fully congested situation, fair share cannot be higher than achieved throughput as it will affect other flows. But for partial congestion or no congestion, the fair share is certainly higher than the achieved throughput. means if we transmit data packet in ideal case (without congestion) means normal data delivery to the destination, but if any intermediate path node use by the any other sender node at the same time on the same route due to increase data packet sending load on intermediate path node because first node already sending data packet on same intermediate path node so first sender data delivery delay will be increases and also if both sender send's data greater than the available bandwidth so grunted that data has been dropped and acknowledgement of sending packet will be delay due to congestion on intermediate path node, to avoid this situation we used bandwidth estimation technique to obtain available bandwidth of intermediate path nodes. If the delay of received acknowledgment of the sending packet at the sender node is increases, then in our approach we calculate delay difference between previous acknowledgment ack_{n-1} and current acknowledgment ack_n send by the destination node, we assume Δt is delay difference of two acknowledgement. So, we can calculate Δt as follows.

$$\Delta t = (ack_n - ack_{n-1}) \text{..... (5)}$$

This delay difference used to estimate available bandwidth of intermediate paths node for set new data sending rate according to the available bandwidth of the intermediate path node to avoid congestion in the network. At the beginning when we create connection and decide sender node and destination node we considered **d** is data packet, t1 is time taken to data packet sending from sender node to destination node through intermediate path node and t2 is time taken to acknowledgement to sender node from destination node through intermediate path node and we assume U_{BDSR} is ideal case basic data sending rate**.** So we can calculate U_{BDSR} in bps from following equation (6).

$$U_{BDSR} = d/(t1 + t2) \text{ ..... (6)}$$

When acknowledgments of sending packets are delayed caused by congestion over heavy loaded route, then in our scheme we estimate available bandwidth from delay difference between two received acknowledgements. We calculate it by subtracting delayed current acknowledgement time from previous acknowledgement time. If increase and decrease difference between these two acknowledgements we required set new data rate to avoid congestion over heavy loaded route in the network. From following equation (7)

$$N_{Threshold} = int\big(\big(U_{BDSR} \times (W/\Delta t)\big)/W\big) \text{..... (7)}$$

Where W is window size.

According to equation (7) the new data sending rate for next packets will be changed according to this equation and if the acknowledgment delay time increase then the $N_{Threshold}$ value will be decreases so the packets will be sent in network according to the available bandwidth hence congestion will not occur in the path and every time this calculation automatically adjust the data sending rate. For example assume initial U_{BDSR} is 500 bps, window size 20, ack_{n-1} = 2sec**.** ack_n = 4sec. So, Δt=2

$N_{Threshold} = \big(\big(500 \times (20/2)\big)/20\big)$;

$N_{Threshold}$ = 250 bps**;**

$N_{Threshold}$ is assign to U_{BDSR} and change value is 250bps.

Here threshold means ideal case data send's by the sender node according to ideal available bandwidth we set cut-off range, but after change available bandwidth because remaining bandwidth used by another sender node so our new threshold (cut-off) set according to above equation. And control the data rate with congestion minimization on the network.

3.1 Proposed Architecture

Here we design basic architecture diagram for both cases NewReno and updated NewReno case, NewReno case if we assume one sender node S2 sends data packet through I1 and I2 nodes to destination D2,. in ideal case S2 to I1 data rate is 50Mb/s and I1 to I2 data rate is 90Mb/s so that time no any congestion on the network because sender S2 sends data in maximum data rate 50Mb/s and I1 and I2 link bandwidth 90Mb/s is greater than the bandwidth S2 to I1. but after some time node S1 and S3 need to send data packet to destination D1 and D3 through I1 and I2 but I1 to I2 link maximum 90Mb/s data rate and here if all three sender S1, S2 and S3 sends simultaneously data without know available bandwidth so multiplex 160Mb data to I1 and that case I1 drop 70Mb data per second because congestion comes on to the intermediate node I1. But next figure 3. Shows updated scheme of NewReno in that case if same situation given in above description, node sends data packets through that intermediate node so sending packets will be delay and received acknowledgments also delay at sender node sends through destination node due to heavy loaded on intermediate path nodes. So in our proposed scheme we calculate delay deference between acknowledgements at the sender node and estimate available bandwidth of intermediate path nodes using delay interval of acknowledgements according to equation (6) and (7) and set new data sending rate to sends next data packets from source to destination through intermediate path

nodes according to available bandwidth of intermediate path nodes and minimize the congestion and packet drop rate of the mobile ad-hoc network

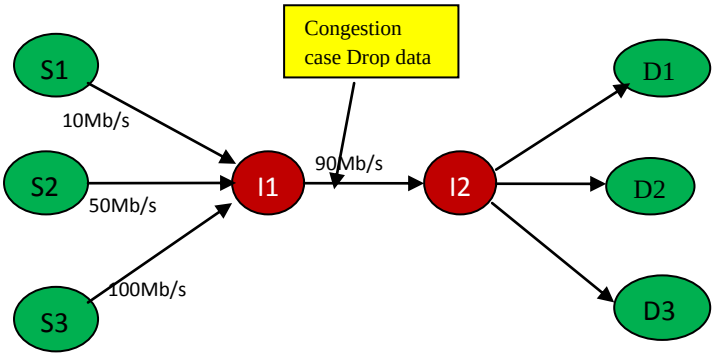

Figure 2- Architecture of TCP/Newreno Scheme

.

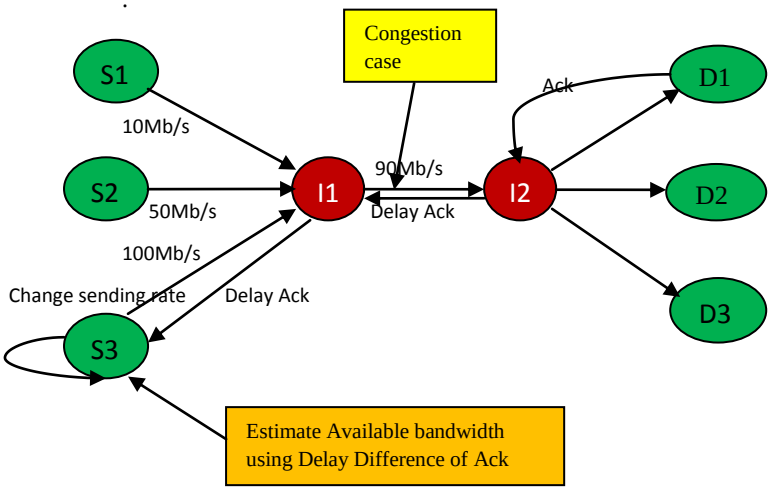

Figure 3- Architecture of TCP/U_NewReno Scheme

3.2 Flow Graph for our Approach

Here we create flow graph 4 according to our algorithm very first we create mobile node and then sender and receiver node, sender node call the routing module and generate route request packet, that time sender broadcast routing packet to its all neighbour if nodes belong in 250 meter range, if we find destination so that create routing table and sends routing acknowledgement to actual sender node, after that sender node sends data packet through newly created routes to destination, After destination node sends acknowledgement to destination, But our module also check bandwidth of each node, if any node available bandwidth less than the data rate of sender node so each intermediate node inbuilt acknowledgement packet with bandwidth information and send to sender node, so that sender node minimize the data rate and efficiently utilize available bandwidth with minimize congestion or avoid congestion on the network. Here above approach we use and enhanced the performance of the network.

Steps of Flow Graph:

(1) Step 1: Use NS-2.31 simulator for simulation

(2) Step 2: Generate mobile node = M

(3) Step 3: Create Senders $S_{1 \ldots\ldots} S_n$ and destination D_1 …… Dn

(4) Step 4: attach routing protocol as AODV to the sender nodes

(5) Step 5: Broadcast routing packet through all radio range nodes.

(6) Step 6: Check radio range and next neighbour if out of range go to step 11

(7) Step 7: if yes accept RREQ packet from neighbour and forward to next neighbour.

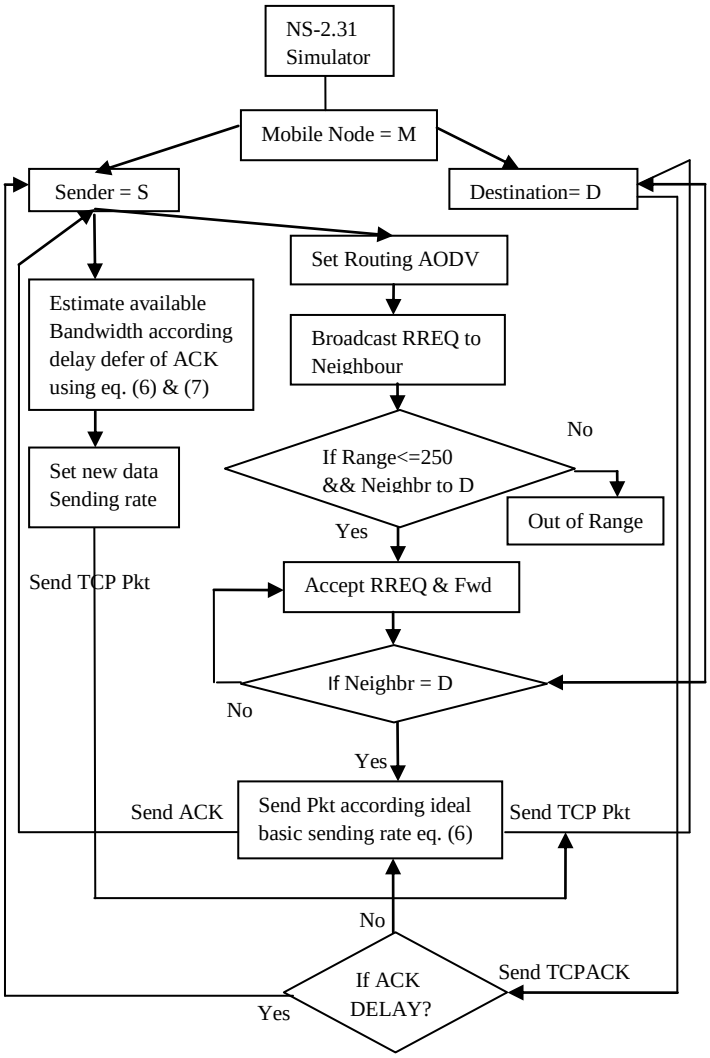

Figure 4 Flow Graph for our Module

(8) Step 8: If Destination Find out create routing table

(9) Step 9: Send positive acknowledgment to source node for data sending

(10) Step 10: according to acknowledgment sender set data rate and send it to destination so congestion not comes on the network.

(11) Step 11: If received acknowledgments delayed at sender node we estimate available bandwidth of intermediate path nodes using delay interval of acknowledgements and set new data sending rate to avoid congestion in network.

(12) Step 12: Out of range

(13) Step 13: Simulation End

3.3 Algorithm for Efficient Way TCP Communication

Step1: Create mobile node = M;

Step2: Set routing protocol = AODV; // for Routing Protocol

Step3: Set sender = S; // S € M

Step4: Set Destination = D; // D € M

Step5: Initialize radio range = 250m;

Step6: Set MAC = 802.11; // for Media access Control

Step7: Sender B_RREQ to next-> neighbour;

While (next->neighbour <= 250 || next->neighbour! = Destination)

{

Accept route requested packet;

Forward B_RREQ to next neighbour;

If (next->neighbour == Destination)

{Create route table;

Check more than one route table;

Send (R_ACK to sender node through more than one path)

}

Else { Check route request limit time;

If (RRLT>= Route search time)

{Exit 0 ;}

}

}

// after connection establish between sender to destination we apply TCP/Newreno for reliable and congestion control

Step8: Attach application data = FTP; //File transfer Protocol

Step9: Set Agent = TCP/Newreno; //Newreno send Data according to available bandwidth

Step10: Check sender routing table ();

If (r_table ==true)

{

Send (TCP pkt to given route at basic sending rate)

Destination receives (TCP pkt)

//Destination sends TCP ACK to sender node

Goto Sender (Rx-TCP-ACK);

}

Else {

Destination unreachable;

Routing table not exit;

}

Step11: Sender (Rx-TCP-ACK)

If (TCP-Ack is delay)

{

Estimate available bandwidth

Send TCP data through Minimize new TCP Sending-Rate;

}

Else

{

Send data through basic sending- Rate;

}

End Simulation;

4. SIMULATION AND RESULT

We have simulate our work using the simulation NS-2.31 network simulator and we have considered different parameter to analyze the result .these parameter given in following table 1. In normal case we have use no of nodes 10 , the simulation time is 100 second, packet size 512 bytes and we have simulate to the Updated TCP New Reno technique on these parameter and find out the result show in the form of graphs and tables of the congestion window , throughput, ,packet drop rate , packet sending rate in MANET .

Table 1.Simulation parameter

Sr. No	Simulation Environment	Parameter
1	Simulator Used	NS-2.31
2	Number of Nodes	10
3	Dimension of simulated area	100m×100m
4	Routing Protocol	AODV
5	Simulation Time	100 sec.
6	Transport Layer	TCP , UDP
7	Congestion Window Type	NewReno,U_NewReno
8	Traffic Type	CBR (3pkt/s)
9	Packet Size	512 bytes
10	Number Of traffic connections	530
11	Node Movement at Maximum Speed (m/s)	random
12	Transmission Range	250m

In our scheme we have simulate the result using available bandwidth estimation technique we have obtained the result of different parameters that gives improve result than previous New Reno scheme. These parameters show result in the form of graphs and tables.

4.1 TCP Congestion Window Analysis

In our simulation we create two TCP connections with New Reno and NewReno update scheme and analyze the comparative result between them. That result shows x-coordinate as simulation time in seconds and y-coordinate represents congestion window, according to result output NewReno technique gives constant congestion window from (0-to- 72) second and after 72 second window size only 10 that result at the time of TCP connection one in figure 5 ,Here we also compare that result through updated NewReno technique , this time our result improve in different- different simulation time and fall because we apply bandwidth estimation mechanism in NewReno technique so each TCP sender node change self congestion window according to intermediate node bandwidth but average result of our scheme is improved and we minimize the congestion of the network and increase the packet reception ratio.

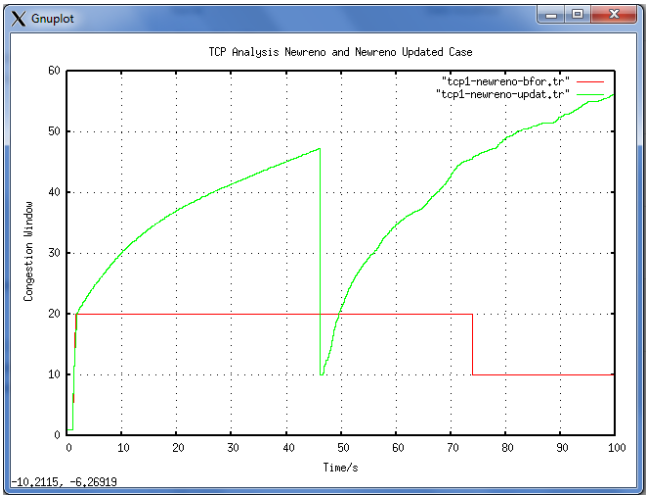

Figure 5. TCP Conection -1 Analysis

Here we represent TCP connection analysis in table 2 and table conclude that updated Newreno gives better result as comapre to newreno mechanism.

Table 2. TCP Conection -1 Analysis

TCP Connection-1 Analysis		
Time	**NewReno**	**Update_NewReno**
1.02147	1	10
5.00548	20	26
10.0015	20	30
25.0463	20	40
40.0392	20	45
75.0095	10	47
99.8787	10	56

In graph 7 we analysis second TCP connection on the bases of above parameter and find out our updated Newreno approch give result according to intemediate node bandwidth.

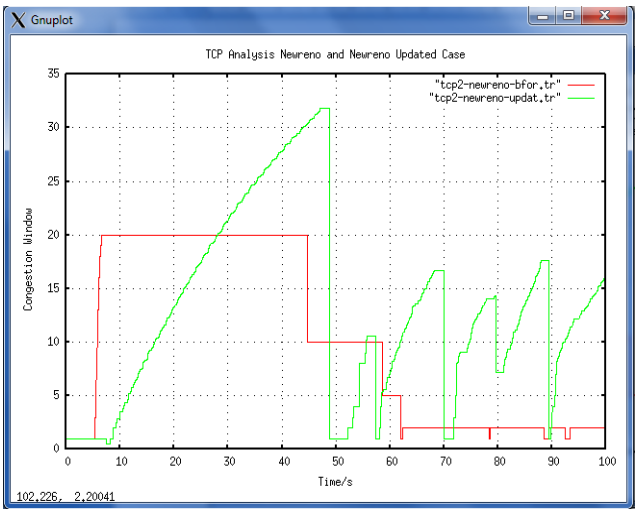

Figure 6. TCP Conection -2 Analysis

Here we represent TCP connection analysis in table 3 and table conclude that updated Newreno gives better result as comapre to newreno mechanism.

Table 3. TCP Conection -2 Analysis

TCP Connection-2 Analysis		
Time	**NewReno**	**Update_Newreno**
1.02147	1	1
5.00548	1	1
10.0015	4	20
25.0463	20	20
40.0392	20	28
75.0095	3	14
99.8787	3	16

4.2 Throughput Analysis

This metric represents the total number of bits forwarded to higher layers per second. It is measured in bps. It can also be defined as the total amount of data a receiver actually receives from sender divided by the time taken by the receiver to obtain the last packet. In our simulation we use TCP/New Reno and TCP/U_New Reno mechanism and find out our updated New Reno approach gives better result on the bases of throughput that concludes if we use bandwidth estimation and according to bandwidth data sending technique apply so our result is very good.

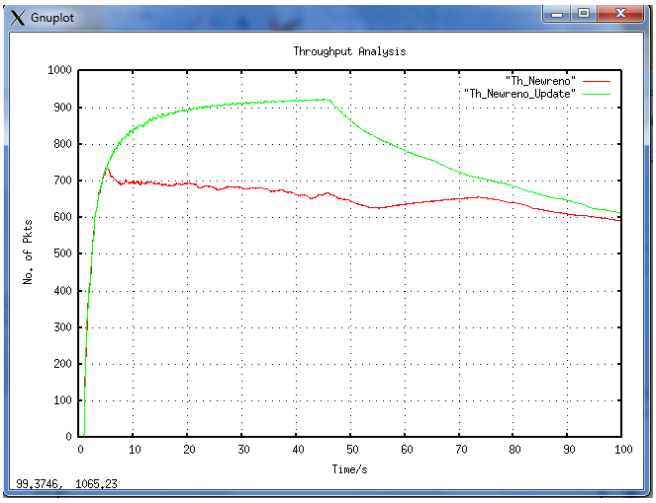

Figure 7. Througput Analysis

In table 4 we show the throughput analysis in discrete event of time and we get updated TCP New Reno mechanism gives better throughput at each point as compare to previous TCP NewReno mechanism.

Table 4. Througput Analysis Table

Throughput Analysis Table		
Time	**NewReno**	**Update_NewReno**
1.02147	0.97	0.98
5.00548	725.4	732.507
10.0015	696.563	835.75
25.0463	679.835	902.681
50.0392	645.22	864.286
75.0095	653.162	703.745
99.8787	591.144	612.295

4.3 TCP Sender And Reciver Packet Analysis

That table 5 shows comparison between NewReno and Updated NewReno, here we create two TCP connections node four and six as a sender and two and nine as receivers at NewReno case total 3109 packets sends but the update New Reno case 3613 packets sends it means 16% packets sending increases and same receiving percentage increases.

Table 5 TCP Packets Send Vs Receives Analysis

TCP Packet Analysis		
Sender Node	TCP Packet Sends NewReno	TCP Packet Sends Updated NewReno
4	2035	2534
6	1074	1079
Total Packet Sends	3109	3613
Receiver Node	TCP Packet Receive NewReno	TCP Packet Receive Updated NewReno
2	1048	1041
9	2023	2515
Total Packet Receive	3071	3556

5. ABOUT THE DEMONSTRATION

In this demo, we analyze TCP congestion control technique using updated New Reno and New Reno technique and we get the result our approach gives best result as compare to previous approach. We also use two TCP connection and parameter as through put.

6. CONCLUSION AND FUTURE WORK

In this paper, we have discussed and analyzed issue related to the TCP congestion control over MANET environment through extensive simulation using ns-2. updated New Reno technique adopts available bandwidth estimation algorithm at the sender side to optimize the packet drop rate and congestion window size and to improve the TCP throughput , packet sending rate when network congestion is detected .The simulation results have confirmed that updated TCP New Reno technique has a significant improvement over TCP New Reno . We conclude that Updated New Reno technique give best performance of TCP congestion control over MANET with bandwidth information provide to the sender node so that sends data according to available bandwidth. Our approach case throughput is very good as compare to previous New Reno technique and also decrease packet drop rate. In This paper we simulate with two TCP connections case in future we also analyze denser network case. here we only use bandwidth estimation mechanism with the acknowledgement base but acknowledge comes only TCP time if we apply UDP connection so our module does not work because UDP can't gives any acknowledgement so in future we also work UDP case congestion control scheme and analyze the best possible result. In future we control congestion through routing overhead minimization so we apply various location aware routing protocol like LAR (location aware routing) Dream routing etc. and minimize the congestion.

7. ACKNOWLEDGMENTS

The authors are thankful to Samrat Ashok Technological Institute, Vidisha (M.P.) India for extending their supports in this article.

8. REFERENCES

[1] Prasanthi. S and Sang-Hwa Chung "An Efficient Algorithm for the Performance of TCP over Multi-hop Wireless Mesh Networks" 609-735, IEEE 2010.

[2] J.Postel. Transmission Control Protocol. Network Working Group, Request for Comments: 793, Sep.1981.

[3] R.Draves, J.Pandhye and B.Zill, "Routing in Multi-Radio Multi-hop Wireless Mesh Networks", MOBICOM 2004.

[4] V.Kawadia and P.R.kumar,"Experimental Investigations into TCP Performance over Wireless Multihop Networks", Proc. Of ACM SIGCOMM,2005

[5] D. Wei, C. Jin, S. Low, and S. Hegde, "FAST TCP: motivation, architecture, algorithms, performance," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1246–1259, 2006.

[6] L. Xu, K. Harfoush, and I. Rhee, "Binary Increase Congestion Control (BIC) for fast, long distance networks," in *Proceedings of IEEE INFOCOM'04*, March 2004, pp. 2514–2524.

[7] S. Floyd, "High Speed TCP for large congestion windows," *RFC 3649*, December 2003.

[8] T. Kelly, "Scalable TCP: improving performance in high speed wide area networks," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, pp. 83–91, April 2003.

[9] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: new techniques for congestion detection and avoidance," in *Proceedings of ACM SIGCOMM'02*. ACM, August 31-September 02 1994, pp. 24–35.

[10] S. H. Low, L. L. Peterson, and L. Wang, "Understanding TCP Vegas: a duality model," *Journal of the ACM*, vol. 49, no. 2, pp. 207–235, 2002.

[11] The Network Simulator ns-2 http://www.isi.edu/nsnam/ns/

[12] Habibullah Jamal, Kiran Sultan ," Performance Analysis of TCP Congestion Control Algorithms" *INTERNATIONAL JOURNAL OF COMPUTERS AND COMMUNICATIONS*, Issue 1, Volume 2, 2008

[13] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, and R. Wang. TCP Westwood: End-to-End Congestion Control for Wired/Wireless Network. In Wireless Networks journal, volume 8, page 467, 2002.