Volume 3, Issue 5, May 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Performance Evaluation of Flow Count Mechanism for Congestion Control

Dr.Yogesh Chaba[1]
[1] *Associate professor,*
GJU Hisar, India.

Poonam[2]
[2] *M.tech(cse)*
GJU Hisar, India.

Abstract - Ad-hoc network are flexible, self-configurable, easy and fast to deploy. Computer networks have long suffered from congestion so there is a growing need to control the congestion in the network. We define congestion as the loss of utility to a network user due to high traffic loads and congestion control mechanisms as those that maximize a user's utility at high traffic loads .We consider the problem of protecting well-behaved users from congestion caused by ill-behaved users by allocating all users a fair share of the network bandwidth. Fairness is achieved when equal numbers of packets are received from each node and this will be achieved by restricting the queue size and limited bandwidth .This aggregate queue orders packets based on their timestamps rather than order of arrivals. Through simulation, we show the performance of AODV, DSR and AOMDV.

Index Terms — Congestion, congestion control mechanism, Fairness, routing protocols, FIFO

I. Introduction

When the number of packets increases beyond the limit that can be handled by the network resources, the network performance degrades, and this situation is called congestion. Congestion simply means overcrowding or blockage due to overloading .It is similar to traffic jam caused by many cars on a narrow road. In Traditional congestion control policy The best-effort services based on two algorithms, FIFO (First In First Out) and Drop Tail [3]. Routing nodes add each received packet to FIFO queue, and every data packet sent to all. When the nodes in congestion, it will drop the follow-up data packets out without distinction. It is easy to achieve, but when the network congestion happened, the network nodes will suddenly discard at the large number of packets, the status of the network changes reflect the slow, easily leads to the SSO network, and reduces network utilization. [1]In the traditional congestion control policy when the network has too many packets, network resources in a demand for more than the available resources can provide, the network performance will be degraded. It is called the congestion. The main reason caused the congestion is the network resources cannot satisfy the needs of users. The resources include the cache space, the capacity of bandwidth link and the handling capacity of network nodes. The network lacks of admission and control capacity, when resources in the network is insufficient, it cannot restrict the number of users, and can only rely on lower quality of services to continue to service users. It is called the best-effort service [2]. The rest of this paper is organized as follows. Section II describes the congestion control mechanisms and different techniques to manage congestion. Section III gives a brief description of the related work. Section IV presents the simulation setup for different routing protocol. Results of simulation experiments are presented and discussed in Section V. Finally, conclusions from this study are listed in section VI.

II. Congestion Control Mechanisms

The four algorithms, Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery are described below.

A) Slow Start-

Slow Start, a requirement for software implementations is a mechanism used by the sender to control the transmission rate, otherwise known as sender-based flow control. This is accomplished through the return rate of acknowledgements from the receiver. In other words, the rate of acknowledgements returned by the receiver determines the rate at which the sender can transmit data. When a connection first begins, the Slow Start algorithm initializes a congestion window to one segment, which is the maximum segment size (MSS) initialized by the receiver during the connection establishment phase. When acknowledgements are returned by the receiver, the congestion window increases by one segment for each acknowledgement returned. Thus, the sender can transmit the minimum of the congestion window and the advertised window of the receiver, which is simply called the transmission window. Slow Start is actually not very slow when the network is not congested and network Response time is good. For example, the first successful transmission and acknowledgement of a segment increases

the window to two segments. After successful transmission of these two segments and acknowledgements completes, the window is increased to four segments. Then eight segments, then sixteen segments and so on, doubling from there on out up to the maximum window size advertised by the receiver or until congestion finally does occur. At some point the congestion window may become too large for the network or network conditions may change such that packets may be dropped. Packets lost will trigger a timeout at the sender. When this happens, the sender goes into congestion avoidance mode as described in the next section

.b) Congestion Avoidance

During the initial data transfer phase the Slow Start algorithm is used. However, there may be a point during Slow Start that the network is forced to drop one or more packets due to overload or congestion. If this happens, Congestion Avoidance is used to slow the transmission rate. However, Slow Start is used in conjunction with Congestion Avoidance as the means to get the data transfer going again so it doesn't slow down and stay slow. In the Congestion Avoidance algorithm a retransmission timer expiring or the reception of duplicate ACKs can implicitly signal the sender that a network congestion situation is occurring. The sender immediately sets its transmission window to one half of the current window size (the minimum of the congestion window and the receiver's advertised window size), but to at least two segments. If congestion was indicated by a timeout, the congestion window is reset to one segment, which automatically puts the sender into Slow Start mode. If congestion was indicated by duplicate ACKs, the Fast Retransmit and Fast Recovery algorithms are invoked .As data is received during Congestion Avoidance, the congestion window is increased. However, Slow Start is only used up to the halfway point where congestion originally occurred. This halfway point was recorded earlier as the new transmission window. After this halfway point, the congestion window is increased by one segment for all segments in the transmission window that are acknowledged. This mechanism will force the sender to more slowly grow its transmission rate, as it will approach the point where congestion had previously been detected.

c)Fast Retransmit

When a duplicate ACK is received, the sender does not know if it is because a segment was lost or simply that a segment was delayed and received out of order at the receiver. If the receiver can re-order segments, it should not be long before the receiver sends the latest expected acknowledgement. Typically no more than one or two duplicate ACKs should be received when simple out of order conditions exist. If however more than two duplicate ACKs are received by the sender, it is a strong indication that at least one segment has been lost. The sender will assume enough time has lapsed for all segments to be properly re-ordered by the fact that the receiver had enough time to send three duplicate ACKs. When three or more duplicate ACKs are received, the sender does not even wait for a retransmission timer to expire before retransmitting the segment (as indicated by the position of the duplicate ACK in the byte stream). This process is called the Fast Retransmit algorithm and was first defined in [8]. Immediately following Fast Retransmit is the Fast Recovery algorithm.

d) Fast Recovery

Since the Fast Retransmit algorithm is used when duplicate ACKs are being received, the sender has implicit knowledge that there is data still flowing to the receiver. Why?

The reason is because duplicate ACKs can only be generated when a segment is received.

This is a strong indication that serious network congestion may not exist and that the lost segment was a rare event. So instead of reducing the flow of data abruptly by going all the way into Slow Start, the sender only enters Congestion Avoidance mode. Rather than start at a window of one segment as in Slow Start mode, the sender resumes transmission with a larger window, incrementing as if in Congestion Avoidance mode. This allows for higher throughput under the condition of only moderate congestion [9].

e) Techniques used to manage congestion-

The basic techniques may be used to manage congestion.

1. End-system flow control- This is not a congestion control scheme, but a way to prevent the sender from overrunning the buffers of the receiver.
2. Network congestion control- In this scheme, end systems throttle back in order to avoid congesting the network. The mechanism is similar to end-to-end flow controls, but the intention is to reduce congestion in the network, not the receiver.
3. Network-based congestion avoidance - In this scheme, a router detects that congestion *may* occur and attempts to slow down senders before queues become full.[5]

III. Related Work

Although many congestion control algorithm have been proposed in the literature for wireless network, the design of the algorithms are challenged by having to support different levels of services, fairness and implementation complexity. Many researchers have compared their proposal schemes on different congestion control schemes, but there is no common, simple and standardized congestion control scheme to make their comparisons with Addisu Eshete and Yuming Jiang[7] This paper presents S-SFQ which is a single queue design and implementation of the well-known Start-time Fair Queueing (SFQ). This

aggregate queue orders packets based on their timestamps rather than order of arrivals. SFQ can fairly approximate the fairness, both simple and weighted, of per flow queues.In addition, we discuss in detail the adverse effect of packet loss synchronization problem common in such aggregate queues. The contribution of this paper is twofold. First, we provide a design and implementation of SFQ based on a single shared queue, rather than a rack of queues. This in turn calls for a non-FIFO queue that can sort packets based on their encoded timestamps. This is because all flows share the queue and the policy for flow scheduling must be different from the order of packet arrivals. Upon arrival, packets are assigned start tags computed by the underlying SFQ algorithm. The tag determines the packet position in the queue and its transmission priority. The second important contribution is that we identify and discuss a recurring problem in the context of this work called loss synchronization that has a potential to obliterate the desirable qualities of the underlying scheme. We find that this problem commonly arises in shared queues with no specialized (randomized) buffer management. Thompson and Nagle [10] represent the two points of Implementation in congestion control. The first is at the source, where flow control algorithms vary the rate at which the source sends packets .Of course flow control algorithms are designed primarily to ensure the presence of free buffers at the destination hosts but we are more concerned with their role in limiting the overall network traffic the second point of implementation is at the Gateway .congestion can be controlled at gateways through routing and queuing algorithms. Queuing algorithms, which control the order in which packets are sent and usage of gateway's buffer space, do not effect congestion directly, in that they do not change the total traffic on the gateways outgoing line. Queuing algorithm can be thought of as allocating three nearly independent quandaries' bandwidth(which packet get transmitted),promptness (when do these packets get transmitted),and buffer space(which and when packet get discarded by the gateway).Nagle proposed a Fair Queuing algorithm in which Gateway maintain separate queue for packets from each individual

IV. Simulation Setup

This evaluation will be done through simulation on various network parameter such as varying queue length and number of sender increased. Check the performance of congestion control mechanisms and how mechanism behaves when we increase number of sender and usages. This includes the platform i.e. UBUNTU and the tools such as ns2 (Network Simulator version 2), NAM (Network Animator) and Trace graph. Then the core implementation is discusses how the three protocols i.e. AODV, DSR and AOMDV were simulated and implemented. First the platform i.e. UBUNTU was set up in a virtual environment. Then ns2 was setup on the platform on which the above said protocols were implemented. Ns2 requires a script file to be run on it. These script files are written in a language called TCL (Tool Command Language).

TABLE I
Simulation parameters

setval(chan)	Channel/WirelessChannel	# channel type
set val(pro)	Propagation/TwoRayGround	# radio-propagation model
set val(ant)	Antenna/Omni Antenna	# Antenna type
set val(ll)	LL	# Link layer type
set val(ifq)	Queue/DropTail/PriQueue	# Interface queue type
set val(ifqlen)	70	# max packet in ifq
set val(netif)	Phy/WirelessPhy	# network interface type
set val(mac)	Mac/802_11	# MAC type
set val(nn)	4	# no of mobile nodes
setval(rp) # routing protocol		DSR,AODV,AOMDV
set val(x)	800	
set val(y)	800	

A. Drop Packets:-

Drop packets are those packets which are dropped during the simulation. Dropped packets are generated during simulation but not received by the receiver. Graph 5.1 shows that drop rate of DSR protocol is very low as compared to AODV and AOMDV. Drop rate increased by increasing the queue length. AODV and AOMDV have nearly same drop rate. At queue

length 70 both the protocol has same dropped packets which are app to 155. AOMDV have slightly better drop rate as compared to AODV.

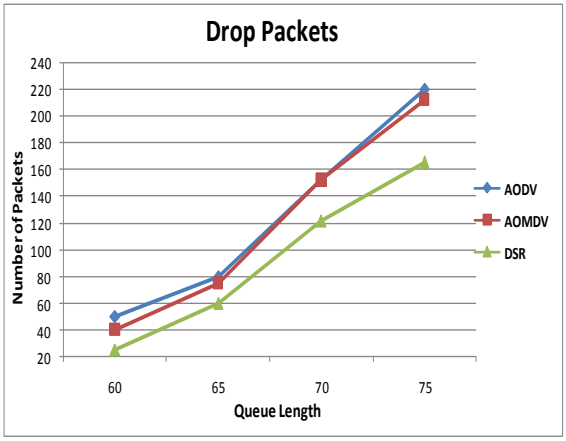

Fig. 5.1 no. of drop packets without mechanism

1. Drop Packets By Using Flow Count Mechanism:-
As shown in graph 5.2 when flow count mechanism applied it encounters in great reduction of drop packets
At queue length 60 all the three protocols have nearly same which is below 5.By increasing the queue length drop rate in DSR is low as compared to AODV and
AOMDV. In the entire three protocols drop rate in AODV is very high. Drop rate very much decreased by applying the flow control mechanism because it restrict the no of packets which are buffered .It stops the senders to generate the packets after a limit .

2. Drop Packets By Using Flow Count Mechanism:-
As shown in graph 5.2 when flow count mechanism applied it encounters in great reduction of drop packets
At queue length 60 all the three protocols have nearly same which is below 5? By increasing the queue length
Drop rate in DSR is low as compared to AODV and
AOMDV. In the entire three protocols drop rate in AODV is very high. Drop rate very much decreased by applying the flow control mechanism because it restrict the no of packets which are buffered .It stops the senders to generate the packets after a limit .

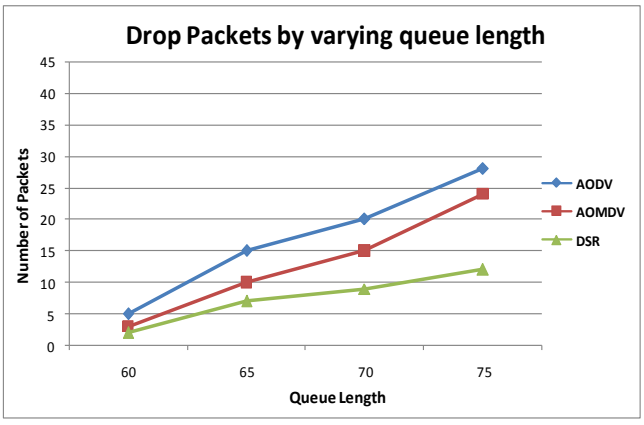

Fig : 5.2 Number of Dropped Packets with Mechanism

2. Packet Delivery Ratio:-
The graph showing the packet delivery ratio of three
Protocols are illustrated as follows .as shown in the graph the packet delivery ratio of the DSR is much better when the queue length is larger. AODV and AOMDV has approximately same packet delivery ratio but DSR not perform well as

compared to aodv and AOMDV on smaller queue length, as queue length increases the performance of AODV, AOMDV decreases but DSR performance increases without using the Mechanism .

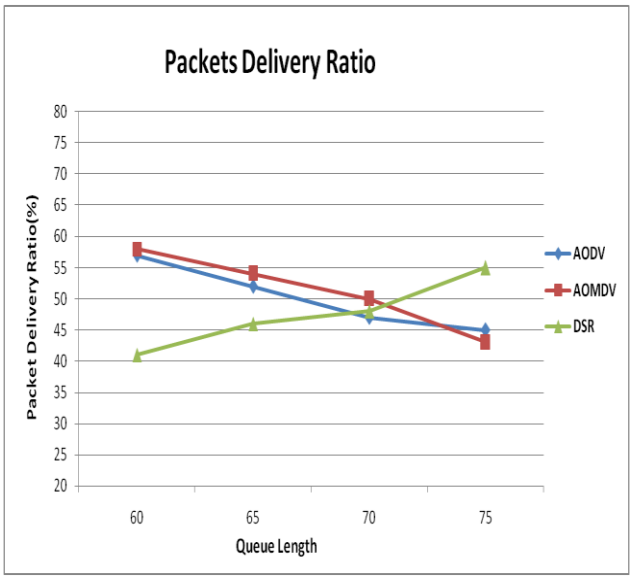

Fig: 5.2 Average Packet Delivery Ratio

3. Packet Delivery Ratio Using Flow Count:-

The graph 5.3 showing Packet Delivery ratio of the protocol is illustrated as follows. From the graph the Packet Delivery Ratio of DSR is better than AODV and AOMDV at larger queue length. On queue length 70 DSR, AODV and AOMDV shows approximate same result after queue length 70 the loss rate increased in AODV and AOMDV. So these two protocols not sustain their performance at larger queue length. By better utilization of bandwidth DSR performance increases by increasing the queue length .This is because of restricting the queue length and flow count mechanism .Active node can monitor available bandwidth, and make use of its active calculating ability to inform sources send-control rate and thus control the flow of data within the channel transmission rate to minimize the occurrence of congestion.

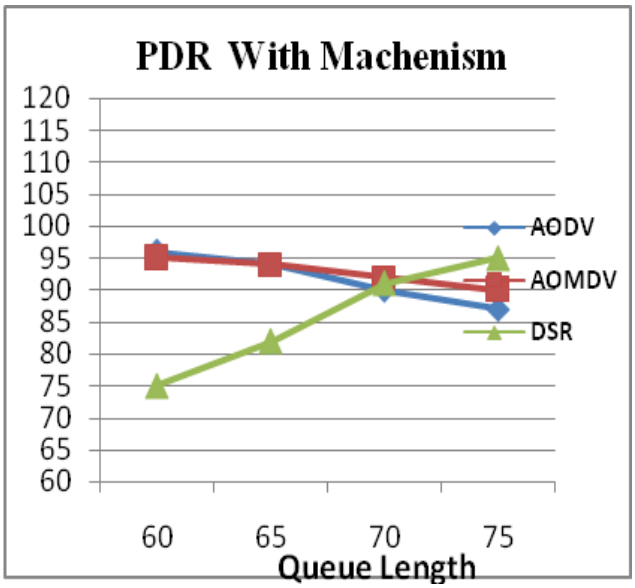

Fig: 5.3 Avg Packet Delivery Ratio With Mechanism

4. Throughput:-

Throughput is the number of packets that is passing through the channel in a particular unit of time. As it can be seen from the following graph 5.4 shows average throughput without applying the congestion control mechanism on three protocols AODV, AOMDV and DSR. The throughput of DSR protocol is better than AODV and AOMDV .These two protocols gives approximate same result. DSR protocol provides better results.

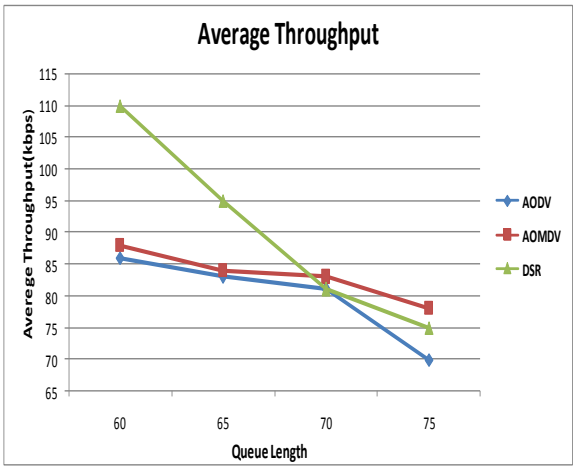

Fig. 5.4 Average Throughput

Throughput with Flow Count Mechanism:--

From graph 5.5 it has been observed that all the Protocols (AODV,AOMDV,DSR)provide better results by applying the mechanism .In all the protocols throughput of DSR is better than the other two as shown in graph AODV perform equivalent to AOMDV on higher queue length .AOMDV perform low initially but it perform better then AODV after increasing the queue length .DSR perform well throughout the queue length. The main reason behind that a forwarding node refuses to accumulate large number of packets from a flow. Once the buffering quota is reached, the node will not accept more packets from that flow before it can forward some of the buffered packets. We show that our scheme can provide a coarse control over bandwidth allocation to routing protocol. DSR protocol utilizes its bandwidth properly on small and larger queue lengths.

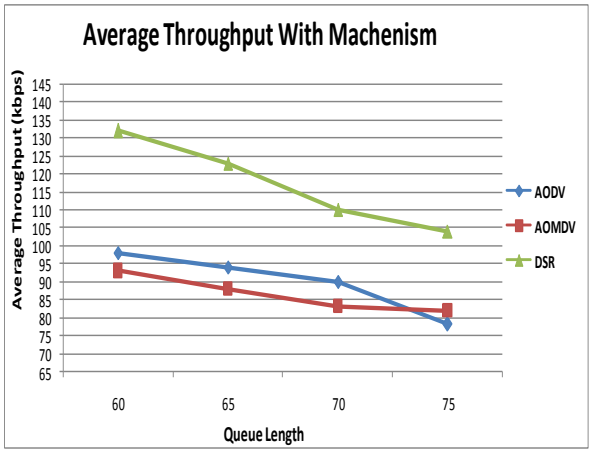

Fig. 5.5 Average Throughput with Mechanism

V. Result And Analysis

Packet Delivery Ratio, Packet drop and Throughputs are evaluated by performing simulation experiments. In these simulation experiments is compared with Flow Count Mechanism by varying the queue length 0, 60, 65, 70, 75 and queue type is FIFO.

VI. Conclusion

In this paper AODV, AOMDV and DSR routing protocol were studied. The performance evaluation parameters were Packet Delivery Ratio, Throughput. We have studied the impact of varying queue length and Flow Count. Congestion in bust networks depends on the number of active flows and the total storage in the network. Total storage included both router buffer memory and packets in flight on long links. In this dissertation, a simple flow counting algorithm is presented. The algorithm takes a few instructions per sender and uses one bit of state per flow. The algorithm provides congestion feedback by varying the number of packets per sender in proportion to the queue length. This approach has the desirable effect of reducing queuing delay, however it produces high loss rate as the number of flows increases, causing long and unfair timeout delays. When the packets arrived, packets are assigned start tags. This tag determines the packet position in queue and

transmission policy. It reduces the packet drop rate. The comparison between AODV, AOMDV and DSR have been shown with the help of some graph taken under various performance metrics .The average throughput and packet delivery ratio of DSR is better than the both. DSR perform better at larger queue length because drop packet rate is low even at large queue length. We show that our scheme can provide a coarse control over bandwidth allocation to routing protocol. DSR protocol utilizes its bandwidth properly on small and larger queue lengths. DSR protocol provide better results because it allows the network to completely self organizing and self configuring without the need of existing network .With the help of Flow control mechanism all the three protocols generates more packets. Fairness is achieved when equal numbers of packets are received from each node. The router itself insures that the bandwidth is allocated fairly. So the unfriendly flow cannot affect the flow and the limited effect is there. The current works has been limited with constant pause time and fix simulation area with CBR traffic.

Reference

[1] CHONG LIU, YANJUAN ZHENG,"*Analysis and Research on the Traditional Congestion Control Policy and Active Networks Congestion Control Policy"*

[2] Carlo Tarantola, *"Dynamic Active Networks Services"*, Proceedings of the 2004 IEEE International Conference On Mobile Data Management, pp. 46-47, June 2004.

[3] A. Kortebi, S. Oueslati, and J. Roberts, *"Cross-protect: implicit service differentiation and admission control",* in High Performance Switching and Routing, 2004, pp. 56–60.

[4] Elizabeth M. Belding-Royer Charles Perkins,*"Ad hoc on demand distance vector (aodv)routing".* IETF Internet Draft, 2003.

[5] David Maltz David Johnson.*" Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks"*. Ad Hoc Networking, pages 139–172, 2001.

[6] Mahesh K. Mariha, Samir R. Das,et al. *"On-demand Multipath Distance Vector Routing in Ad-Hoc Networks"*. IEEE Proceedings of the Ninth International Conference on Network Protocols Page: 14, ISSN:1092 1658,2001

[7] On the Flow Fairness of Aggregate Queues Addisu Eshete and Yuming Jiang Center for Quantifiable Quality of Service in Communication Systems (Q2S)Norwegian University of Science and Technology,Trondheim, Norwayaddisu.eshete@q2s.ntnu.no ymjiang@ieee.org

[8] Van Jacobson, *"Modified TCP Congestion Control Avoidance Algorithm*". end-2-end-interest mailing list, April 30, 1990.

[9] W. Stevens." *TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms,*" January 1997, RFC 2001.

[10] Thompson and Nagle's,"*Analysis andsimulation of Fair Queuing Algorithm"* published in internetworking :Research and Experience,vol. 1,3-26,1990

