A Survey on Congestion Control Mechanisms in High Speed Networks

K. Satyanarayan Reddy[†] and Lokanatha C. Reddy[††]

Research Scholar, Dept. of CS School of Science & Technology, Dravidian University, Kuppam-517425, A.P., India. [†]
Professor, Dept. of CS School of Science & Technology, Dravidian University, Kuppam-517425, A.P., India. [††]

Summary

In this paper an effort has been made to study various Congestion control techniques used for reducing/easing the level of congestion and subsequently avoiding the congestion of the wired communication networks in general and High Speed Networks in particular. Many authors have suggested several congestion control techniques [2], [3], [4], [8], [12], [14], [15], [23], [29], [32] & [33] and studied their behavior under various network conditions, for a range of parameters also under heterogeneous networking environments. A special effort has been made to study the problems associated with the TCP congestion control mechanisms and the several solutions that have been proposed to improve its performance. This analysis tries to study the limitations of the suggested solutions, based on various parameters and propose algorithms to overcome these limitations for the High Speed Networks.

Key words:

TCP, Congestion Control, Throughput, Performance, High Speed Networks.

1. Introduction

High Speed Networks [27] refer to the networks supporting high data rates like high speed LAN's and Ethernets. The range of data rates may vary from few Mbps to Gbps. To achieve a greater degree of performance from the High Speed Networks, the end systems must regulate their flow of data for using the network resources efficiently without overloading the systems, which results in congestion and throughput collapse.

Network congestion refers to a situation in which the network resources are overloaded quite often, i.e., the total demand for a network resource exceeds its capacity. Current advancements in technology only add to the problem of congestion - for example, an increase in the buffer capacity increases packet delays and the period of the delay can be so long that by the time packets reach the destination, the sending source might have timed out its timer and retransmitted the copies of the packets thus choking the network with duplicates (which ultimately are dropped by the receiver and in the process such duplicates occupy considerable amount of network resources and processing during the transmission). Similarly, increased link speed increases the possibility of congestion because of mismatch in link speeds at the point of interconnection of a High Speed Network and a Low Speed Network.

For sending data to a bottleneck link, the sending source uses two rate control techniques for adjusting the data rates namely Open loop & Closed loop. Open loop control technique is useful when the traffic characteristics are defined precisely and the performance requirements are known well in advance, thus the network reserves the available resources for the connections.

Closed loop technique is used when the network resources cannot be reserved or traffic characteristics are not defined in precise terms. In this case the network resources are shared fairly and efficiently amongst the various users. The performance of the systems using the closed loop technique mainly depends on the feedback delay].

1.1 Packet-Oriented Networks:

In packet oriented networks [1], like the internet, the data transfer between the end systems occurs in fixed or variable units of packets of limited size. The intermediate nodes, between the end systems, called as routers which are equipped with queues (buffers) used for storing the packets in transition temporarily and then forwarding them in the direction of destination when the link is free. Since packet-oriented networks have the inherent property that they can get congested locally. So congestion control has to be performed for improving the overall network performance, by controlling the load produced by all the data streams in the network.

Based on the current load conditions of the network, the congestion control is done by adapting efficiently the sending rate of each source of the data streams, thus reducing or even preventing the congestion also allowing

Manuscript received January 5, 2008
Manuscript revised January 20, 2008

a high utilization of the available bandwidth of the network.

1.2 Congestion Control in Packet-Oriented Networks:

In packet-oriented networks, two fundamental types of congestion-control mechanisms [2] can be distinguished regarding the role of the network protocol:

(1). In packet oriented networks the network protocols and routers play important roles. Network protocols frequently inform the sending sources about the current load conditions in the network. The sources store current load conditions of the network in congestion control variables which are used for controlling the congestion. This leads to high utilization of Bandwidth [9], [13] and increase in performance. This advantage of such a congestion-control mechanism is combined with two disadvantages like

First, the congestion-control information transferred by the network protocol requires some additional overhead. There is a trade-off between the frequency/ overhead and the benefit that can be expected if such a congestion-control mechanism is performed.

Second, the upper-layer protocols working on top of the network protocol are limited in their flexibility, as they have to evaluate and react on the congestion-control information supported by the network protocol.

(2) Congestion control can be excluded from and not supported by the network protocol and the routers of the network. Then, the protocols working on top of the network protocols are responsible for the congestion control in the network. In this case, each source has to frequently collect network information, store them in its congestion-control variables, and locally perform congestion control based on values of these variables. One main problem of this approach is that the network information collected by a sender does not reflect very well the current network conditions. The result is a sub-optimal congestion control in terms of network utilization and data stream performance.

Another problem of this approach is that the source of each new data stream entering the network does not know anything about the current load conditions in the network. Therefore, such a source starts sending its data very conservatively using a small sending rate, estimates and probes the current network-load conditions by continuously increasing its sending rate; after a while in which the TCP sender has raised its local knowledge about the current network load little by little that it is able to perform a more accurate congestion control based on the so far collected network information.

In the meantime, the congestion control [17], [18] of this data stream might be also far from optimality. Besides being fair, efficient, responsive and stable; a congestion control technique must be robust against the loss of information also it must scale well with the increase in the speed of the link, the distances and the users.

This paper is organized as follows. In Section 2 we describe various congestion control mechanisms. In Section 3 we have taken up various congestion parameters, followed by comparative analysis of the congestion control algorithms in Section 4. In Section 5 we have proposed the flowchart for congestion detection & control. In Section 6 we have presented the expected results to be obtained from the experimental & simulation study. Finally conclusion of the paper has been presented in Section 7.

2. Various Congestion Control Mechanisms

Many Congestion Control Algorithms have been designed namely:

- Random Early Detection (**RED**), **DECbit**
- Back Pressure Technique
- Choke packet Technique
- Implicit Congestion Signaling
- Additive Increase and Multiplicative Decrease (**AIMD**) [3], [34]
- Explicit Congestion Notification (**ECN**) in TCP/IP
- Binary Congestion Notification(**BCN**)

RED: The gateway detects incipient congestion [12], [31] by computing average queue size and would notify connections of congestion either by dropping packets arriving at it or by setting a bit in packet headers. When the average queue size exceeds a preset threshold max_{th} it marks or drops each arriving packet with probability 1, where the exact probability is a function of the average queue size. If the average buffer occupancy is less than the preset threshold min_{th} then no packets are dropped.

RED gateways keep the average queue [5] size low while allowing occasional bursts of packets in the queue. During congestion, the probability that the gateway notifies a particular connection to reduce its window is roughly proportional to that connection's share of the bandwidth through the gateway. The disadvantage of RED algorithm [35] is that when the average queue occupancy reaches

max_{th}, the packets present in the queue and which are otherwise perfectly alright are all dropped. This happens because the drop probability increases with the increase in the average queue length.

DECbit congestion avoidance scheme [27], [28] is an early example of congestion detection at the gateway. The gateway uses a *congestion-indication* bit in packet headers to provide feedback about congestion. When the average queue length exceeds one, the gateway sets *congestion-indication* bit in the header of arriving packet. The sources use the window based flow control mechanism. They update their windows of data packets once every two round trip times. If at least half of the packets in the last window had the *congestion-indication* bit set, then the window size is decreased exponentially, otherwise it is increased linearly. The main disadvantages are the averaging of queue size for fairly short periods of time and no difference between congestion detection and indication.

Back Pressure Technique: If a node becomes congested [27] then it slows down or stops receiving the packets from the nodes from which it is receiving packets. If this restriction persists for long then packet sending nodes themselves become congested which in turn propagate the restriction on their preceding nodes. But this method is of limited utility as it can be used for the connection oriented networks supporting Hop by Hop flow control.

Choke packet Technique: A choke packet [27], [29] is a control packet generated at the congested node & this packet is transmitted back to the source node to restrict the traffic flow. As the source receives the Choke packet it has to reduce its transmission rate till it stops receiving the choke packets. But this method is crude method as a choke packet does not indicate to the sending source, the status of delivery (receipt / non-receipt) of the packets.

Implicit Congestion Signaling: When the sending source comes to know of congestion [27] at a node if the propagation delays of packets are detected that is the delay is longer than fixed propagation delay and it may ultimately lead to packet discard. But the sending node should have a mechanism to detect increased delays and packet discards.

Explicit Congestion Notification in TCP/IP (ECN): The purpose of this method is to react to congestion [4], [15], [19], [21], [23] in a controlled & in a fair manner. It especially operates over connection oriented networks. In this method the network alerts the end systems about the growing congestion within the network & the end systems. ECN allows routers to set the Congestion Experienced (CE) bit in the IP packet header as an indication of congestion to the end nodes as an alternative to dropping the packet. There are two types of ECN namely Forward Explicit congestion (FECN) and Backward Explicit congestion (BECN).

ECN cannot be relied upon to completely eliminate packet losses as indications of congestion, and therefore would not allow the end nodes to interpret packet losses as indications of corruption instead of congestion. Similarly, ECN does not eliminate the need for Fast Retransmit and Retransmit Timeout mechanisms to detect dropped packets, and therefore does not eliminate the need for the Limited Transmit procedure.

Binary Congestion Notification (BCN): In TCP/IP based networks, congestion [20], [32], [33] is indicated by dropping packets at congested routers. Packets are dropped when the queue of the router reaches its limit (drop tail scheme). To handle congestion situations before packets actually get dropped several proposals for using binary congestion control have been made. With such an approach, routers with capability of detecting incipient congestion can just mark the arriving packets as congested instead of discarding them. The destination copies the value of the congestion bit of the received packets into the acknowledgement packets sent back to the source. The source then changes its transmission window in accordance with the value of the congestion bit.

3. Parameters under study:

As we know, the following are the major metrics for measuring the Network Performance

Fairness,
Latency,
Jitter,
Packet Loss,
Throughput,
Link/Channel Capacity (Bandwidth),
Link utilization,
Availability and
Reliability.

Fairness: Fairness measures or metrics are used in networks to determine whether users or applications are receiving a fair share of system resources. There are several mathematical and conceptual definitions of fairness. The Jain's equation [28] states $fairness = \frac{(\sum_{i=1}^{n} x_i)^2}{n.\sum_{i=1}^{n} x_i^2}$. This equation rates the fairness of a set of values. The result ranges from $\frac{1}{n}$ (worst case) to 1 (best case). This metric identifies underutilized

channels and is not unduly sensitive to typical network flow patterns.

Latency: A common measure of latency [6] is the Round Trip Time (RTT), the time between dispatch of a packet from source and receipt of an acknowledgement that it has reached its destination but in general it is

$$\text{Latency} = \text{RTT} + W_t + P_t$$

W_t: wait time at queues at routers.

P_t: packet processing time at receiving host & generate Acknowledgement.

Jitter: A short-term variation in the rate at which packets travel across a network is called jitter. The jitter [19], [27] can be of two types viz. delay jitter & latency jitter. Variation in the time it takes for packets to reach their destination is delay jitter. The corresponding variation in the latency is latency jitter.

Packet loss: Packet loss [27] is the fraction (usually expressed as a percentage) of packets dispatched to a given destination host during some time interval for which an acknowledgement is never received. Such packets are referred to as being lost.

$$\text{Packets Loss\%} = \frac{\text{No. of unacknowledged packets}}{\text{Total No. of packets transmitted}} X\ 100$$

TCP uses the fraction of lost packets to gauge its transmission rate: if the fraction becomes large then the transmitting host will reduce the rate at which it dispatches packets. As a rule of thumb, a network with a packet loss of 5-15% is said to be severely congested, and one with a higher rate is likely to be unusable for most practical purposes.

Throughput: Throughput [7] is the rate at which data flow past some measurement point in the network. It can be measured in bits/sec, bytes/sec or packets/sec.

Throughput is measured by counting the traffic over an interval and a care must be taken to choose this interval appropriately. A long interval leads to averaging out transient bursts and lulls in the traffic. A shorter interval will record these temporary effects, even if they are not important in the context of the measurement.

Link capacity: Maximum Throughput [19] which a link can offer for transferring bits reliably.

Utilized Link capacity: It is defined as the current traffic load excluding the traffic from host,

Available Capacity = Link Capacity − Utilized Capacity

Achievable Capacity: It is the fraction of the available capacity which can utilized.

Access Rate: It is the Maximum data rate.

Link Utilization: It is defined as simply the throughput (as defined above) divided by the access rate and expressed as a percentage.

$$\text{Link Utilization} = \frac{\text{Throughput}}{\text{Access Rate}}$$

The Availability: It is the fraction of time during a given period when the network is unavailable.

Reliability: It is related to both availability and packet loss. It is the frequency with which packets get corrupted (due to network malfunction); as distinct from being lost.

But note that when calculating the packet loss (above) it is conventional to include corrupted packets as well as lost ones.

In this work, we are considering the following parameters for the purpose of comparison of various congestion control mechanisms:

Fairness, Latency, Jitter, Packet Loss, Throughput, Link Capacity (Bandwidth) & Link utilization.

4. Comparative Analysis:

The comparative analysis of all the congestion algorithms based on the parameters discussed above are presented in tabular form in the **Appendix – A**.

5. Proposed Flowchart for Congestion Detection & Control:

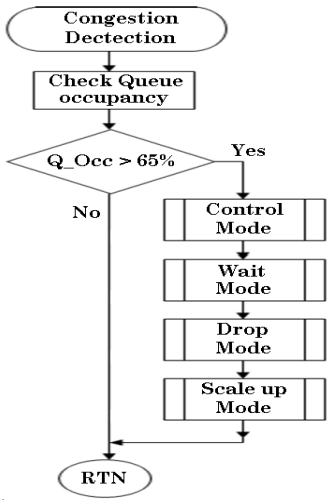

Figure 1. Congestion Control.

This module for the Congestion Detection [8], [10], [11], [16], [23] will work at the switch level. This module continuously monitors the level of queue occupancy to detect the likely congestion. This is done as follows:

a. Calculate the Total percentage of Queue occupancy.
b. Check whether % of queue occupancy is greater than 65.
c. If the total queue occupancy is found to be more than 65% , Control Mode module is called which calculates the % of queue occupancy of each sending source & sends messages to the source to reduce their current transmission rates by a factor (1, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$,…..) depending on severity of congestion and % of queue occupied by the source.
d. Then the module enters in Wait mode, wherein it waits for pre-calculated time duration, to check whether the source reduces the transmission rate.
e. If a source fails to comply then the module calls the Drop Mode module and from the priority queue all the packets of the non-behaving source are removed and the bandwidth is added to the total available bandwidth.
f. Then the Scale up module is called which allows the new sources to get connected to the network.

For the proposed algorithm we are making the following assumptions:

5.1. Network Traffic Classification

5.1.1 Traffic from Behaving sources:

All the Sender nodes that transmit the packets as per the agreed terms of Quality of Service (**QoS**) [22], [25] & [26] and during congestion, the nodes which reduce their current sending rates accordingly after receiving the choke packets from congested node are called the Behaving sources.

5.1.2 Traffic from Non-Behaving sources:

All Sender nodes that do **NOT** transmit the packets as per the agreed terms of **QoS** even after receiving the RM or Choke packets from the congested node for reducing their current sending rate are called the non-Behaving sources. Such non-behaving nodes keep on transmitting more and more packets which may lead to worsening of network congestion due to high percentage of queue occupancy and bandwidth requirements thus not allowing the genuine users to get connected to the Network.

5.2 Queues

The Input Queue [5], [31] at a Router / switch is a priority queue where as the Output queue is a general queue. It is assumed that the incoming packets are accommodated into the Priority Queue**,** to take care of the packets that are coming from the non-behaving sources and which need to be dropped based on the factor of percentage of Queue occupancy.

The packets from non-behaving sources will be DROPPED only in case of severe Congestion. Otherwise the sources are required to reduce their packet transmission rate.

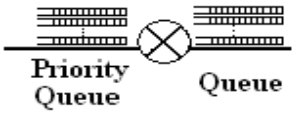

Figure 2. Queues.

5.3 Bandwidth Management:

We propose to manage the network bandwidth using the Dynamic Programming Algorithm [24], [30] assuming that Network bandwidth is to be allocated amongst ‘n’ number of hosts, which are willing to connect (to communicate with the other nodes) to the network.

Let $B_1, B_2, \ldots, B_n$ be the bandwidth requirements of the ‘n’ hosts respectively and let the total throughput T be expressed as *sum* of the individual throughputs of the ‘n’ hosts as follows:

Maximize $T(B_1, B_2, \ldots, B_n) = t_1(B_1) + t_2(B_2) + ... + tn(B_n)$;

with $T(t_i, B_i) = TAB - BRI$

where **TAB:** Total Available Bandwidth is given by the formula $TAB = T(B_n) - T(t_{i-1}, B_{i-1})$ and **BRI:** Bandwidth requested by i^{th} node, $t_j(B_j) = T(t_i, B_i)$ is the **throughput** of the link 'j' with bandwidth requirement of B_j for j=1,2,…,n. The constraint on bandwidth can be defined as:

$$B_1+B_2+\ldots+B_n \leq B \text{ ; where } B_i \geq 0 \text{ for } i=1,2,\ldots,n$$

where B is the total link Bandwidth.

The algorithm using the **Dynamic Programming** approach will manage the bandwidth by allowing Scalability (both Scaling/de-scaling) under the following scenario [12], [14]:

a). From Congestion to No Congestion:

The nodes are allowed to **increase** their packet sending rate i.e. Source Rate [13], [19], [20] as ($\frac{1}{128}, \frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}$, 1) during this phase new hosts may request for getting connected to the network and the permission is granted if required Bandwidth is available and on **QoS** negotiations.

b). From No Congestion to Congestion:

The hosts are requested to **decrease** their packet sending rate i.e. Source Rate [13], [19], [20] as ($\frac{1}{128}, \frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}$, 1) during this phase new nodes are *not allowed* to request for getting connected to the network unless existing connected node's packets have been dropped for misbehavior.

Further, if the already connected nodes do not accede to the request, till some time, then the packets from such sources are dropped based on the criteria of % **of queue occupancy** for dropping the packets. This process of dropping of the packets continues till the congestion Eases and ultimately Congestion clears.

6. Expected Results:

The proposed model is expected to

a. Optimize the Bandwidth and make the bandwidth available to the Behaving sources under Congestion situation and also when there is No Congestion.

b. Maximize the Throughput for the Behaving sources under Congestion situation and also when there is No Congestion.

c. Meet the QoS demands of the Network Traffic during Congestion situation and also when there is No Congestion.

d. Reject/drop all the packets from the Non-behaving source, during congestion, and packets from the behaving sources are accepted and accommodated in queue for onwards transmission.

e. Allow scaling up i.e. allocating Bandwidth to a new host which agrees to behave by sending packets as per QoS agreement.

7. Conclusion:

Under severe congestion condition, the TCP congestion control algorithm goes into Slowstart mode i.e. all the sources have to drastically reduce their packet transmission rate to one packet and then again they slowly increase their transmission rate through Additive Increase & Multiplicative Decrease (AIMD).

The proposed algorithm does not make all the sources to reduce their packet transmission rate drastically, instead it makes the behaving sources to reduce the transmission rates based on percentage of their queue occupancy, also it penalizes the non-behaving sources by dropping all their packets present in the queue, and makes the bandwidth of such source available for allocation to the sources which wish to get connected to the network. Thus allowing to scale up.

8. Acknowledgements:

I wish to thank the Management and the Principal of the New Horizon College of Engineering, Panathur Post, Bangalore, India for having extended all the support and encouragement for carrying out this work.

9. References:

[1] W. R. Stevens, *TCP/IP Illustrated, Volume 1*, Addison-Wesley, Reading, MA, November 1994.

[2] V. Jacobson, "Congestion Avoidance and Control", in *Proc. ACM SIGCOMM*, pp. 314–329, August 1988.

[3] D-M. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks", *Computer Networks and ISDN Systems*, vol. 17, pp. 1–14, 1989.

[4] M. Allman and V. Paxson, "*TCP Congestion Control"*, Internet Engineering Task Force, RFC 2581, April 1999.

[5] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", Internet Engineering Task Force, RFC 2309, April 1998.

[6] T. V. Lakshman, U. Madhow, and B. Suter, "Window-based Error Recovery and Flow Control with a Slow Acknowledgement Channel: A study of TCP/IP Performance ", in *Proc. Infocom 97*, April 1997.

[7] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A Simple Model and its Empirical Validation", in *Proc. ACM SIGCOMM*, September 1998.

[8] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", http://www.aciri.org/tfrc/, June 2000.

[9] T. V. Lakshman and U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss", *IEEE/ACM Trans. on Networking*, vol. 5, no. 3, 1997.

[10] D. Bansal and H. Balakrishnan, "TCP-friendly Congestion Control for Real-time Streaming Applications", Tech. Rep. MIT-LCS-TR-806, MIT Laboratory for Computer Science, May 2000.

[11] "ns-2 Network Simulator", http://www.isi.edu/nsnam/ns/, 2000.

[12] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, Aug. 1993.

[13] D. Andersen, D. Bansal, D. Curtis, S. Seshan, and H. Balakrishnan, "System support for bandwidth management and content adaptation in Internet applications", in *Proc. Symposium on Operating Systems Design and Implementation*, October 2000.

[14] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance", in *Proc. ACM SIGCOMM '94*, August 1994.

[15] M. Mathis and J. Mahdavi, "Forward Acknowledgement: Refining TCP Congestion Control", in *Proc. ACM SIGCOMM*, August 1996.

[16] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A Model Based TCP friendly Rate Control Protocol", in *Proc. NOSSDAV*, July 1999.

[17] S. J. Golestani and S. Bhattacharyya, "A Class of End-to-End Congestion Control Algorithms for the Internet", in *Proc. ICNP*, 1998.

[18] R. Srikant "The Mathematics of Internet Congestion Control", Birkhauser, 2004.

[19] David X. Wei "Congestion Control Algorithms for High speed Long Distance TCP connections", Master Thesis, Caltech, 2004.

[20] Lisong Xu, Khaled Harfoush and Injong Rhee "Binary Increase Congestion Control for Fast Long-Distance Networks", IEEE Infocom, Hong Kong, March 2004.

[21] Jiantao Wang, David X. Wei Steven H. Low "Modelling and Stability of FAST TCP", in Proc. of IEEE Infocom, March 2005.

[22] Weibin Zhao, David Olshefski and Henning Schulzrinne "Internet Quality of Service: an Overview", Columbia University, Research Report CUCS-003-00, February 2000.

[23] C. Jin, D. X. Wei and S. H. Low "FAST TCP: Motivation, Architecture, Algorithms, Performance", IEEE Infocom, Hong Kong, March 2004.

[24] G. L. Nemhauser "Introduction to Dynamic Programming", John Wiley, New York, 1966.

[25] P. Ferguson and G. Huston. "Quality of Service: Delivering QoS in the Internet and the Corporate Network*",* Wiley Computer Books, New York, NY, 1998.

[26] R. Guerin and V. Peris. "Quality-of-service in packet networks: Basic mechanisms and directions", *Computer Networks*, 31(3):pp.169–189, February 1999.

[27] W. Stallings "High-Speed Networks & Internets Performance and Quality of Service", 2nd edition, Pearson Education, 2004.

[28] Jain, R., Chiu, D.M., and Hawe, W. "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems", DEC Research Report TR-301, 1984.

[29] Ao Tang, Jiantao Wang, Steven H. Low, "Understanding CHOKe: Throughput and Spatial Characteristics", IEEE/ACM Trans. Netw. 12(4): 694-707, 2004.

[30] Karp, R., Koutsoupias, E., Papadimitriou, C., Shenker, S., "Optimization problems in congestion control", Foundations of Computer Science, 2000. Proceedings 41st Annual Symposium on Volume , Issue , 66 – 74, 2000.

[31] R. N. Shorten, D.J.Leith, "On queue provisioning, network efficiency and the Transmission Control Protocol",

IEEE/ACM Transactions on Networking, Volume 15, Issue 4, pp. 866 – 877, Aug. 2007.

[32] Ramakrishnan, K.K., and Jain, R., "A Binary Feedback Scheme for Congestion Avoidance in Computer

Networks", *ACM Transactions on Computer Systems*, V.8, N.2, pp. 152-181, 1990.

[33] Dorgham Sisalem, Henning Schulzrinne, "Congestion Control in TCP: Performance of Binary Congestion Notification Enhanced TCP Compared to Reno and Tahoe TCP", 1996. Proceedings of International Conference on Network Protocols, pp. 268 – 275, 1996.

[34] Dmitri L., Hayder R., "Increase-Decrease Congestion Control for Real-Time Streaming: Scalability", INFOCOM 2002. Proceedings of Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies.. IEEE Issue Volume 2, pp. 525-534, 2002.

[35] May, M. Bolot, J. Diot, C. Lyles, B. INRIA; "Reasons not to deploy RED", IWQoS '99. Seventh International Workshop on Quality of Service pp. 260-262, 1999.

Authors:

†K.. Satyanarayan Reddy received his M.Sc.(Maths) & M.Phil.(Maths) Degrees from Nagpur University, Maharashtra and M.Tech. (Computer Applications) from Indian School of Mines, Dhanbad, Jharkhand in 1987, 1988 and 2000 respectively. He is currently associated with Information Science & Engineering Department of New Horizon College of Engineering, Bangalore, Karnataka State, India. He is a Research Scholar in the Dept. of Computer Science at Dravidian University, Kuppam, AP, India and is pursuing his Ph.D. Degree in Computer Science. His current areas of research are Congestion Control in High Speed Networks and Data Communications.

††Lokanatha C. Reddy earned M.Sc.(Maths) from Indian Institute of Technology, New Delhi; M.Tech.(CS) with Honours from Indian Statistical Institute, Kolkata; and Ph.D.(CS) from Sri Krishnadevaraya University, Anantapur. Earlier worked at KSRM College of Engineering, Kadapa (1982-87); Indian Space Research Organization (ISAC) at Bangalore (1987-90). He is the Head of the Computer Centre (on leave) at the Sri Krishnadevaraya University, Anantapur (since 1991); and a Professor of Computer Science and Dean of the School of Science & Technology at the Dravidian University, Kuppam (since 2005). His active research interests include Real-time Computation, Distributed Computation, Device Drivers, Geometric Designs and Shapes, Digital Image Processing, Pattern Recognition and Networks.

[**Appendix-A follows…**]

APPENDIX - A

Comparative Analysis of various Congestion Control Protocols

Metrics ▼ ▼ ▼	*Congestion Algorithms*►►►					
	BCN [32], [33]	**ECN** [15], [31]	**AIMD** [3],[27],[34]	**CHOKe** [29]	**DECbit** [28]	**RED** [5], [12], [35]
Fairness	Fairness is achieved when TCP-BCN are combined. Unfairness increases for long distance connections.	It is unfair to fragile & average flows. With Adaptive ECN (AECN) fairness improves.	It exhibits fair behavior with bulk data transfer.	Differentially penalizes non-responsive and unfriendly flows using queue buffer occupancy information of each flow **(Fair)**.	It only sends congestion signals to those users who are using more than their fair share of bandwidth.	Unable to differentiate between behaving & Non behaving flows **(Unfair).**
Latency	Latency increases for long distance connections.	It reduces the latency of TCP connection.	large queues resolve the tradeoff between efficiency and latency favouring use of small backoff factor.	To admit a packet into the queue through CHOKe requires processing.	It controls the delay by attem-pting to keep the average queue size close to one.	Large buffer sizes may cause this value to go high.
Jitter	Further research has to be carried out for heteroge-neous environment.	When used with RED gateways it reduces the packet losses.	Large buffers take care of short term bursty traffic	For bursty traffic, packet drops for UDP source are more.	It assumes all the sources to be cooperative.	For bursty traffic, packet drops are more.
Packet Loss	Probability of packets getting dropped or marked at congested switches increases proportionally to the number of switches a connection traverses.	ECN reduces the number of packets dropped by a TCP connection.	when the sum of all throughputs reaches buffer capacity *C*, losses occur	If the average queue size is greater than max_{th}, every arriving packet is dropped.	As the sources cooperate by reducing rates packet loss is avoided.	When average queue size reaches "max_{th}", the Drop rate increases.
Throughput	TCP-BCN combination gives better throughput.	Effects of ECN on bulk throughput are less clear.	reduces the number of packets in flight by half on detecting network congestion.	It effectively penalizes the UDP flow to keep the total queue size around min_{th}, and enable TCP flows to get high throughput.	Throughput of multiple users depends on the "cf" the capacity factor (being 0.5 & 0.9).	Is sensitive to the Traffic Load & Parameters.
Link/ Channel Capacity (Bandwidth)	More research is to be carried out for mixed traffic.	As ECN marks the packets so bandwidth allocation is good.	This method is not suitable for High Bandwidth Medium.	It can bind the bandwidth share of UDP flows to a vanishingly small unit, regardless of their arrival rate.	Bandwidth allocation is fair for all the users.	For bursty traffic, packet drops are more & the flows using most of the bandwidth.
Link utilization	TCP-BCN combination gives better link Utilization.	TCP/ECN achieves near 100% link utilization regardless of propagation & queuing delays	If network queues are too small, backoff action causes them to empty with reduction in link utilization.	Bandwidth share is catered Maximum to the responsive sources **(Good)**.	Good with binary feedback mechanism.	It is better if the buffer size is small **(Good).**
Queue	Packets are dropped when the queue of the router reaches its limit.	The average queue size monitors the congestion level.	It should be chosen such that the queue does not empty for significant periods of time.	The random packet selection mechanism effectively penalizes UDP flow after the average queue size reaches min_{th}	It follows regeneration cycle i.e. empty to non-empty to empty.	As the average queue size increases, packets are dropped randomly with increase in drop probability.