International Journal of Soft Computing 1(2) : 83-90, 2006

Interaction Between Polynomial Congestion Control Algorithms and Queue Management Schemes in Wired TCP Networks

[1]M.Chandrasekaran and [2]R.S.D.Wahida Banu
[1]Deptartment of ECE, Govt. College of Engineering, Salem 636 011, India
[2]Department of CSE, Govt. College of Engineering, Salem 636 011, India

Abstract: In this study, the effect of queue management schemes on various congestion control algorithms used in wired TCP networks is compared with that of the newly proposed Polynomial Congestion Control Algorithms. The parameter used for comparison is the total throughput. First the polynomial congestion control algorithms are introduced and analyzed. They generalize the Additive Increase Multiplicative Decrease algorithms and provide additive increase using a polynomial of the inverse of the current window size and provide multiplicative decrease using the polynomial of the current window size. There are infinite numbers of TCP-compatible polynomial algorithms of different order. This study analyses the performance of two such models for the wired TCP networks. Simulations are done using ns2. The results show that the proposed congestion control algorithms perform better than TCP/Tahoe, TCP/Reno, TCP/New Reno and TCP/Fast algorithms. The effects of varying the buffer size on these algorithms are also studied.

Key words: Queue Management, Polynomial Congestion control, TCP, AIMD, ns2, RED and Drop Tail

INTRODUCTION

Modern day computer networks, routers and switches often use First-In-First-Out (FIFO) buffers to multiplex packets from different flows. Also the Internet strongly demands for Quality of Service (QoS) and fairness among flows. Hence they rely on congestion control algorithms to probe and gain bandwidth, avoiding congestion and achieving system fairness[1] .

The rapid growth of the Internet has sparked the demand of several applications, which require the stability of the Internet. The stability of Internet depends on transmission errors, bandwidth sharing of sources that use common bottleneck links, Round Trip Time (RTT) and mainly due to congestion. TCP/IP is the standard network protocol stack on the Internet[2]. TCP's end-to-end congestion control mechanism reacts to packet loss by adjusting the number of outstanding unacknowledged data segments allowed in the network[3]. Such algorithms are implemented in its protocol, TCP[4-6]. In the existing algorithms, increasing the congestion window linearly with time increases the bandwidth of the TCP connection and when the congestion is detected, the window size is multiplicatively reduced by a factor of two[1,8].

This study presents and analyzes a new class of window based nonlinear congestion control algorithms for Internet Transport Protocols and applications such as Internet audio and video for which the rate reduction techniques will degrade the quality[9,10]. The Additive Increase and Multiplicative Decrease (AIMD) algorithms are generalized as Polynomial Congestion Control Algorithms and the proposed algorithms are analyzed in a simulated wired TCP network.. The performance is compared with the existing TCP congestion control algorithms such as TCP/Tahoe, TCP/Reno, TCP/New Reno and TCP/Fast.

There are extensive research works on the congestion control strategies and Queue management schemes[1,11]. But very little work has been done on the joint dynamics and interactions between different congestion control strategies with different queue management schemes.

This study investigates the interaction of various queue management schemes such as Drop Tail, Random Early Discard (RED), Fair Queuing (FQ) etc., with the various congestion control algorithms. These results are compared with that of the proposed Polynomial congestion control algorithms. This study also compares the effect of varying the buffer size on the total throughput of the Congestion control algorithms.

WINDOW BASED CONGESTION CONTROL FOR TCP NETWORKS

TCP is a connection-oriented protocol[3] that maintains a congestion window that controls the number of outstanding unacknowledged data packets in the network. Sending data consumes slots in the window of

Corresponding Author: M.Chandrasekaran, Deptartment of ECE, Govt. College of Engineering, Salem 636 011, India

Table 1: Polynomial Algorithms for Various Values of k and l

Value of k	Value of l	Increase Rule (I)	Decrease Rule (D)	Rule Behavior
MIMD Model				
0	1	$W_{t+R} \leftarrow W_t + 1$	$W_{t+\delta t} \leftarrow W_t - \beta W_t$	AIMD
-1	1	$W_{t+R} \leftarrow W_t + W_t/\alpha$	$W_{t+\delta t} \leftarrow W_t - \beta W_t$	MIMD
-1	0	$W_{t+R} \leftarrow W_t + W_t/\alpha$	$W_{t+\delta t} \leftarrow W_t - 1$	MIAD
0	0	$W_{t+R} \leftarrow W_t + 1$	$W_{t+\delta t} \leftarrow W_t - 1$	AIAD
PIPD Model				
0	0	$W_{t+R} \leftarrow W_t + 2$	$W_{t+\delta t} \leftarrow W_t - 2$	AIAD
1	1	$W_{t+R} \leftarrow W_t + (\alpha/W_t) + (\alpha/W_t)^2$	$W_{t+\delta t} \leftarrow W_t - (\beta W_t) - (\beta W_t)^2$	PIPD
0	1	$W_{t+R} \leftarrow W_t + 2$	$W_{t+\delta t} \leftarrow W_t - (\beta W_t) - (\beta W_t)^2$	AIPD
1	0	$W_{t+R} \leftarrow W_t + (\alpha/W_t) + (\alpha/W_t)^2$	$W_{t+\delta t} \leftarrow W_t - 2$	PIAD

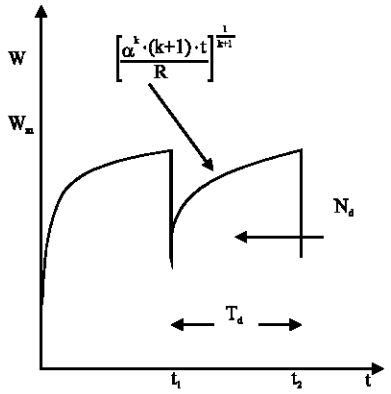

Fig. 1: Window vs time curve of the polynomial algorithm

the sender and the sender can send packets only as long as free slots are available[6]. On start-up, TCP performs slow-start, during which the rate roughly doubles each round-trip time to quickly gain its fair share of bandwidth. In steady state, TCP uses the AIMD mechanism to detect additional bandwidth and to react to congestion. When there is no indication of loss, TCP increases the congestion window by one slot per round-trip time. In case of packet loss, indicated by a timeout, the congestion window is reduced to one slot and TCP reenters the slow-start phase. Packet loss indicated by three duplicate ACKs will reduce the window size to half of its previous size.

The AIMD algorithm may be expressed as given[8,12,13].

$$\text{I: } W_{t+R} \leftarrow W_t + \alpha; \alpha > 0$$
$$\text{D: } W_{t+\delta t} \leftarrow (1-\beta) W_t; 0 < \beta < 1 \qquad (1)$$

Where

- I→Increase in window as a result of the receipt of one window of acknowledgement in a round-trip-time (RTT) and
- D→Decrease in window size on detection of congestion by the sender
- W_t→Window size at time t
- R→RTT of the flow and
- αand β→Increase and Decrease Rule constants.

POLYNOMIAL CONGESTION CONTROL ALGORITHMS

The properties of polynomial congestion control algorithms are discussed. Note that the window adjustment policy is only one component of the congestion control protocol derived from polynomial algorithms. Other mechanisms such as congestion detection (loss, ECN etc.), retransmissions (if required), estimation of Round-trip-time etc., remain the same as TCP[12].

The AIMD rules are generalized as polynomial rules in the following manner

$$\text{I: } W_{t+R} \leftarrow W_t + (\alpha/W_t)^k + (\alpha/W_t)^{2k} + (\alpha/W_t)^{3k} + \ldots; \alpha > 0$$
$$\text{D: } W_{t+\delta t} \leftarrow W_t - (\beta W_t)^l - (\beta W_t)^{2l} - (\beta W_t)^{3l} - \ldots; 0 < \beta < 1 \qquad (2)$$

Considering only the first order terms, the rules are as follows. These rules are named as MIMD model.

$$\text{I: } W_{t+R} \leftarrow W_t + (\alpha/W_t)^k$$
$$\text{D: } W_{t+\delta t} \leftarrow W_t - (\beta W_t)^l \qquad (3)$$

The following are the rules resulting in considering the first two terms of Eq. 2 and these are named as PIPD model.

$$\text{I: } W_{t+R} \leftarrow W_t + (\alpha/W_t)^k + (\alpha/W_t)^{2k}; \alpha > 0$$
$$\text{D: } W_{t+\delta t} \leftarrow W_t - (\beta W_t)^l - (\beta W_t)^{2l}; 0 < \beta < 1 \qquad (4)$$

The various polynomial algorithms for MIMD and PIPD models using the first and second order polynomial rules and for various values of k and l are given in the Table 1.

By choosing different values of αand β in Eq. 3 and 4, they became the members of the polynomial family. Polynomial increase and decrease algorithms of different orders can be formed by including the higher order terms

in Eq. 2 and all possible algorithms that may be used for the window size adjustment for the congestion avoidance results.

The Algorithms represented by Eq. 3 and 4 are implemented in ns-2 by modifying the source code *tcp.cc* for the TCP congestion control. Both these algorithms are implemented to study the variation of window size and the resulting throughput with respect to time. The algorithm begins in the slow-start state[6]. In this state, the congestion window size is doubled for every window of packets acknowledged. Upon the first congestion indication, the congestion window size is cut in half and the session enters into the polynomial congestion control state.

In this state the congestion window size is increased by $[(\alpha/W_t)^k + (\alpha/W_t)^{2k} + (\alpha/W_t)^{3k} + \ldots.]$ for each new acknowledgement received, where W_t is the current congestion window size. The algorithm reduces the window size when congestion is detected. Congestion is detected by two events: (i) triple-duplicate ACK and (ii) time-out. If by triple-duplicate ACK, the algorithm reduces the window size by $[(\beta W_t)^1-(\beta W_t)^{2l}-(\beta W_t)^{3l}\ldots..]$. If the congestion indication is by time-out, the window size is set to 1[6,12].

SIMULATION AND ANALYSIS

In this section, the results of our ns-2[14] simulation of the two polynomial algorithms in wired TCP networks are presented. This study investigates the connections running the TCP-compatible polynomial algorithms MIMD and PIPD represented by Eq. 3 and 4.

The wired TCP network simulations use the topology shown in Fig. 2. It consists of 6 connections sharing a bottleneck link where all connections have an almost identical round-trip propagation delay.

Each polynomial flow uses a modified TCP with AIMD algorithm replaced by the polynomial family; other

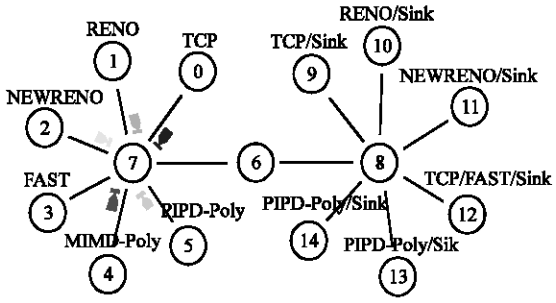

Fig. 2: Simulation topology

mechanisms like slow-start and time-out remain unchanged. Each source always has data to send, modeled using ns's FTP application. The various activities due to the above simulation are recorded using the trace facility of the network simulator and the results of these simulations are used to plot the graphs shown in Figures.

The results are obtained by assuming the two buffer management schemes-Drop Tail and Random Early Detection (RED)-at the bottleneck gateway[15,16]. Figure 3 show the variation of window size over a simulation period of 100 sec for TCP/Tahoe, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD models using Drop Tail buffer management strategy. Figure 5 show the window size variation for the same set of TCP variants using RED buffer management.

Figure 4 show the throughput of the TCP/Tahoe, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD models using Drop Tail buffer management strategy and Figure 6 show the throughput variation for the same set of TCP variants using RED buffer management The throughput (λ) is modeled using the Eq. 5.

$$\lambda = \frac{c \cdot s}{R \cdot \sqrt{p}} \qquad (5)$$

where s is the segment size, R *is* the round trip time, p is the packet loss rate and c is a constant value commonly approximated as. The graphs show that the throughput of MIMD and PIPD models are very high when compared with that of TCP/Tahoe, TCP/Reno, TCP/New Reno and TCP/Fast and hence gain more bandwidth when interacting with them.

The results show that the congestion window size and the resulting throughput of TCP/Reno and TCP/New Reno oscillate between the minimum and maximum values. This is because they try to respond quickly to the changing network situations. This generates a lesser throughput. TCP/Tahoe behaves in fair manner and TCP/Fast performs well. The two proposed polynomial models MIMD and PIPD have an average minimum congestion window size of 10 and 20 respectively for the entire simulation period. This show that these two models are aggressive in gaining the bandwidth and PIPD model is more aggressive.

The simulation topology shown in Fig. 2 is again used to investigate the performance of the TCP variants due to the variation of buffer size. The traced results are used to evaluate the total throughput of each connection

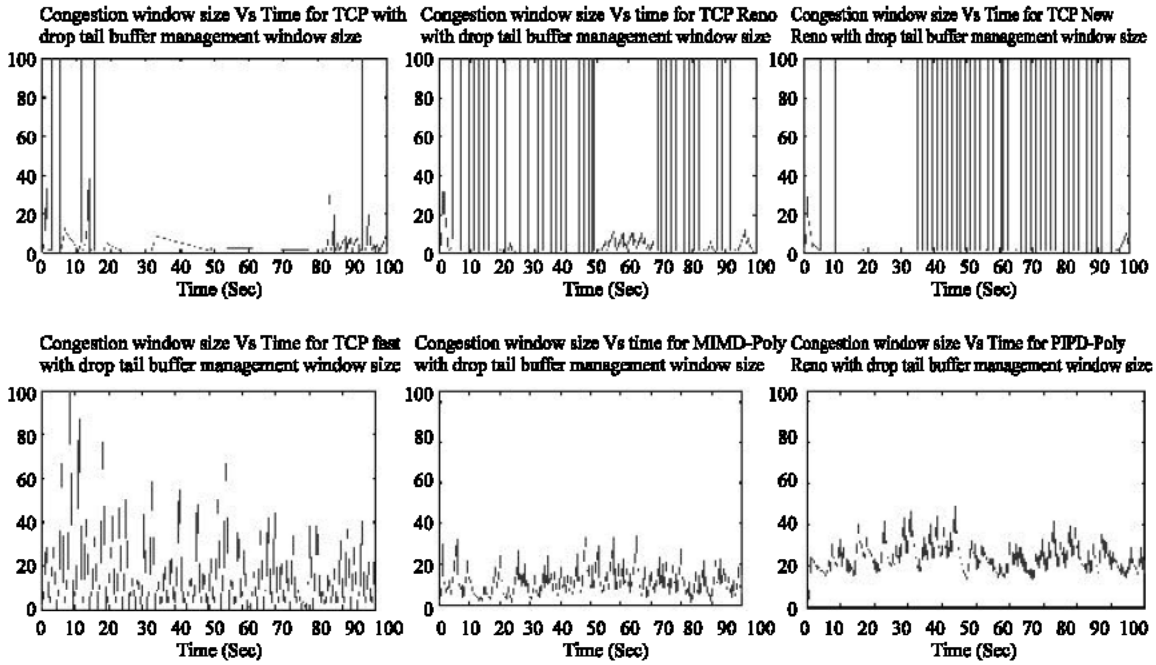

Fig. 3: Window size vs. Time of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using Drop Tail buffer management

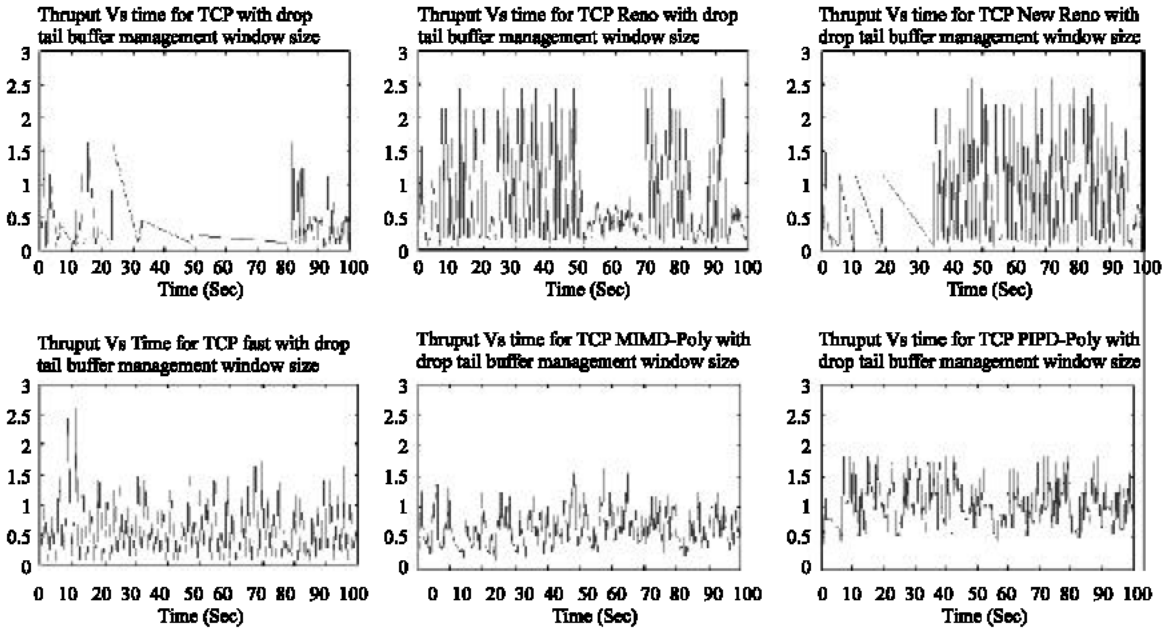

Fig. 4: Total Throughput vs. Time of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using Drop Tail buffer management

using the model given in Eq. 5. The simulation experiments are performed by considering the two Buffer management schemes-Drop Tail and RED. The buffer size is varied from 10 to 100 in steps of 10 packets and the results are plotted in Fig. 7. The total throughput of the connections running TCP/Tahoe, TCP/New Reno and

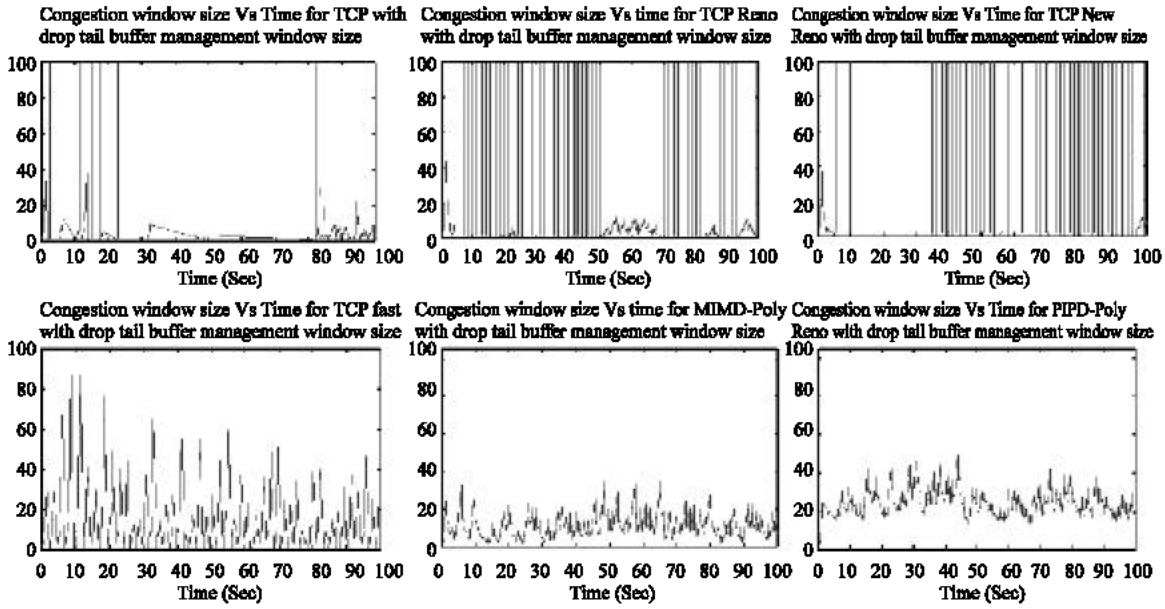

Fig. 5: Window size vs. Time of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using RED buffer management

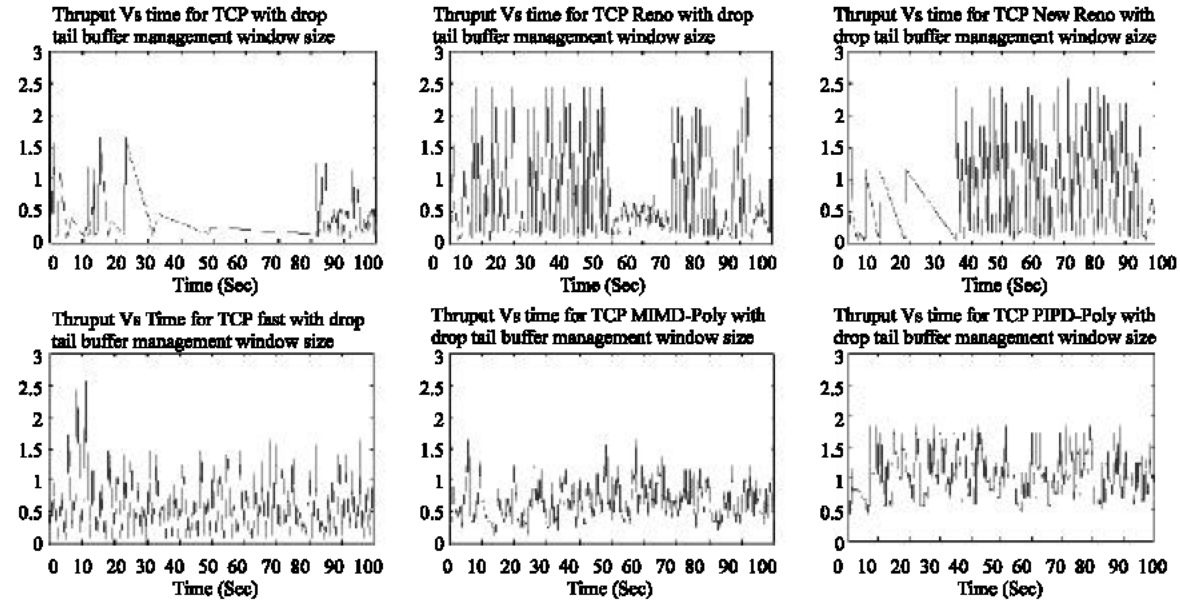

Fig. 6: Total Throughput vs. Time of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using RED buffer management

TCP/Fast algorithms show large variations, varying between 10 and 300KB. TCP/Reno show very poor performance. The two proposed algorithms perform well having an average total throughput of about 400 KB for MIMD model and 600 KB for PIPD model. This shows that the two algorithms are aggressive in gaining the bandwidth in a topology with shared bottleneck link.

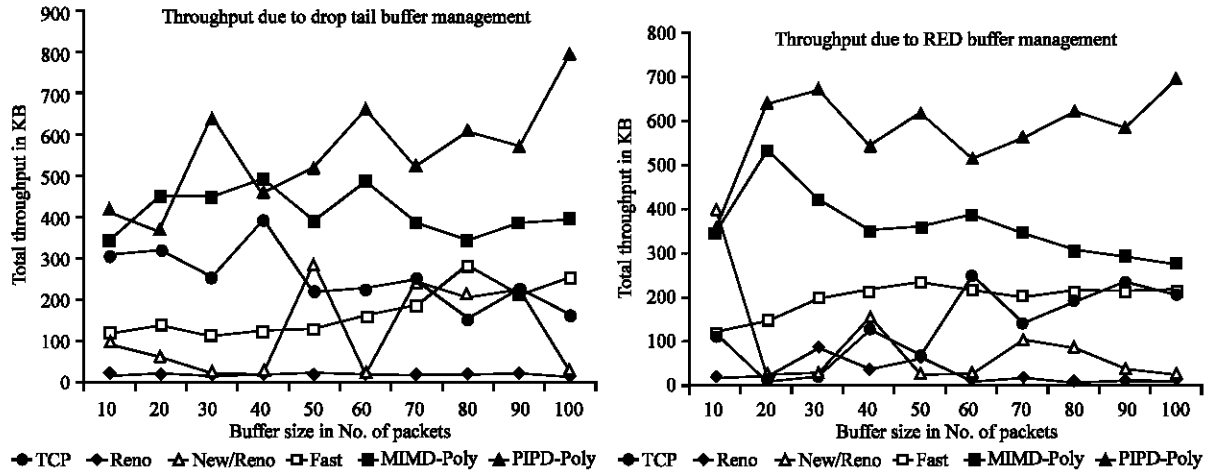

Fig. 7: Total Throughput of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using Drop Tail and RED buffer management and scheduling schemes for different buffer sizes

INTERACTION BETWEEN CONGESTION CONTROL ALGORITHMS AND VARIOUS BUFFER MANAGEMENT SCHEMES

This section analyses the interaction between the TCP Congestion control algorithms (used in the simulation experiments explained in the previous section) and the various buffer management schemes implemented in ns2. Following is the list and brief descriptions of buffer management schemes assumed for the analysis. Most of these schemes detect congestion based on queue lengths at the link, some of the schemes detect congestion based on the arrival rate of the packets at the link and some use both[1].

Drop tail: This scheme implements FIFO scheduling and drop-on-overflow buffer management which is typical of most present-day Internet routers[16].

RED (Random Early Drop): This scheme regulates the queue length to a desired value by adapting the marking probability. This marks each packet with a probability p, which is periodically updated[15].

FQ (Fair Queuing): Implements sharing of the buffer equally among all the contending sources and places the packets fairly in the per-flow queue[17].

SFQ (Stochastic Fair Queuing): Router uses a hashing function to put a flow to its correct position in the per-flow queue[1].

PI (Proportional Integral Controller): This scheme marks each packet with a probability p and the value of p is periodically updated. This is similar to RED scheme. But the update equations used for calculating the value of the probability p are different[18].

Vq (Virtual Queue): This scheme maintains a virtual queue whose capacity is less than the actual capacity of the link. When a packet arrives in the real queue, the virtual queue is also updated to reflect the new arrival. Packets in the real queue are marked/dropped when virtual buffer overflows[11].

REM (Random Exponential Marking): The marking probability depends on the sum of link prices (Congestion Control), summed over all the routers in the path[18].

GK (Gribbens-Kelly Virtual Queue): The link maintains a virtual queue with a buffer size of B'=kB and k<1, where B is the buffer size of original queue. When the virtual queue overflows, all the packets in the real queue and all future incoming packets are marked till the virtual queue becomes empty again[18].

DRR (Deficit Round Robin): The flows arrive at the router are queued in input buffer and wait for an en-queue action. The en-queue finds the right queue for each flow according to its IP source and destination address pair. Then the packets of flows in the correct queue are released to the output buffer according to the round robin rule[17].

SRR (Stochastic Round Robin): This is similar to the weighted round robin, but adjusted to account for variable sized packets, when surplus (unused bandwidth) is carried on to the next round[1].

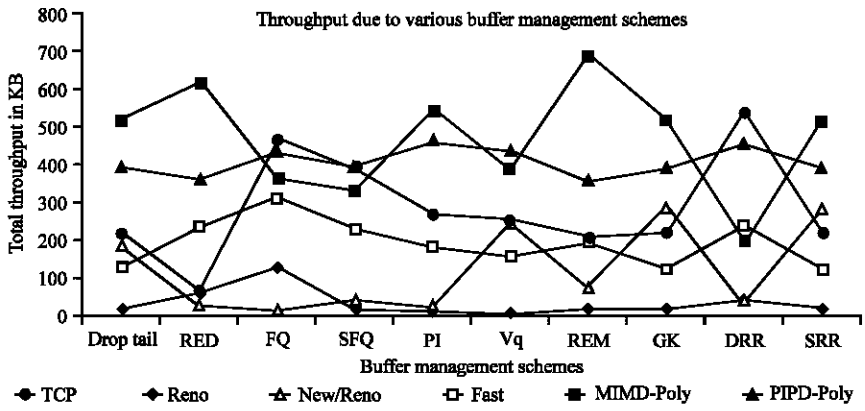

Fig. 8: Total Throughput of TCP, TCP/Reno, TCP/New Reno, TCP/Fast, MIMD and PIPD using 10 different buffer management and scheduling schemes

The network topology shown in Fig. 2 is used for analyzing the throughput variations due to all the above buffer management and scheduling schemes. Many variation of these schemes are available and the above ten schemes are chosen for analysis. The simulation experiments are conducted by assuming each of these schemes one by one and the results are recorded. The Total throughput of each connection running the same six TCP variants assumed in the simulations are plotted and the graph is shown in Fig. 8. Except for the FQ buffer management scheme, the two proposed models gain a larger share of the bottleneck bandwidth for all the buffer management and scheduling schemes.

CONCLUSION

In this study, a family of nonlinear congestion control algorithms, called polynomial algorithms are presented and analyzed. Two models-MIMD and PIPD are considered for evaluation. These polynomial algorithms generalize the familiar class of linear algorithms.

The variation of total throughput is studied assuming the different types of buffer management and scheduling schemes. The effect of varying the buffer size on the total throughput is also recorded. These results show the interaction between the proposed two models and TCP/Tahoe, TCP/Reno, TCP/New Reno and TCP/Fast The interaction of all these algorithms with the various buffer management and scheduling schemes are also presented. All the simulation results showed good performance and interactions between polynomial algorithms (MIMD and PIPD) and TCP using standard algorithms in wired networks. The algorithms MIMD and PIPD obtain higher long-term throughput than the standard algorithms for TCP/Tahoe, TCP/Reno, TCP/New Reno and TCP/Fast.

The future work will be about studying the scalability and fairness of the two models and the applicability of these models to wireless networks.

REFERENCES

1. Chi Zhang and Lefteris Mamatas, 2005. The Interaction Between Window Adjustment Strategies and Queue Management Schemes., In Proceedings of WWIC 2005, pp. 65-74,2005.
2. Bansal, D. and H. Balakrishnan, 2000. TCP-friendly Congestion Control for Real-time Streaming Applications, Tech. Rep. MIT-LCS-TR-806, MIT Laboratory for Computer Science.
3. Jorg Widmer, Robert denda and Martin Mauve, A Survey on TCP-Friendly Congestion Control.
4. Douglas E. Comer, 1991. Internetworking with TCP/IP Vol. I, Englewood Cliffs, New Jersey, Prentice Hall.
5. Stevens, W.R., 1994. TCP/IP Illustrated, Volume 1, Addison-Wesley, Reading, MA.
6. Allman, M. and V. Paxson, 1999. TCP Congestion Control, Internet Engineering Task Force, RFC 2581.
7. Van Jacobson, 1988. Congestion Avoidance and Control, ACM Computer Communication Review, Proceedings of the Sigcomm '88 Symposium in Stanford, CA, 18: 314-329,
8. Richard Yang, Y. and S. Simon Lam, 2000. General AIMD congestion control, in Proceedings of ICNP.
9. Sally Floyd and Kevin Fall, 1999. Promoting the use of end-to-end congestion control in the Internet, IEEE/ACM Transactions on Networking, 7: 458-472.
10. Sally Floyd, 1995. TCP and Explicit Congestion Notification, ACM Comput. Commun. Rev. 24: 8-23.

11. Srisankar Kunniyur and R. Srikant, 2001. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management, In Proceedings of ACM SIGCOMM 2001, San Diego, California.
12. Shudong Jin, Liang Guo, Ibrahim Matta and Azer Bestavros, 2001. A spectrum of TCP-friendly window-based congestion control algorithms, Tech. Rep. BU-CS-2001-015, Computer Science Department, Boston University, July 2001, Available at http://www.cs.bu.edu/techreports/2001-015-spectrum-tcp-friendly.ps.Z.
13. Deepak Bansal and Hari Balakrishnan, 2001. Binomial congestion control algorithms, in Proceedings of IEEE INFOCOM.
14. ns-2 Network Simulator, http://www.isi.edu/nsnam/ns/.
15. Floyd, S. and V. Jacobson, 1993. Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, 1.
16. Tejas Kokje Vijay Kakadia, Analysis of Congestion Control Strategies For TCP Variants Using Droptail and RED Queuing Disciplines Department of Computer Science University of Southern California, Los Angeles
17. Jung-Shian Li, 2002. An Evaluation of Deficit Round Robin Fair Queuing Applied for Router Congestion control, J. Inform. Sci. Engin. 18/2: 333-339.
18. Sanjeewa Athuraliya Victor H. Li, Steven H. Low, Qinghe Yin, 2001. REM: Active Queue Management, IEEE Network Magazine, 15: 48-53.