DEPLOYMENT OF FAIR SHARE AND SMART START TECHNIQUE FOR OPTIMAL USE OF AVAILABLE CAPACITY IN TCP CONGESTION CONTROL

Ganeshkumar P. [1], Thyagarajah K. [2] .
[1]Department of Information Technology, [2]Department of Electrical and Electronics Engineering.
PSNA College of Engineering and Techmology, Dindigul, Tamil Nadu, India.
[2]Research Scholar, Institute of Remote Sensing, Anna University, Chennai, India
Email: [1]p_ganeshkumar@rediffmail.com

Abstract

The traditional TCP congestion control mechanism encounters a number of new problems and suffers in poor performance. In traditional TCP congestion control, slow start results in poor performance. During slow start the number of TCP segments transmitted per unit time is considerably small compared to the capacity of the Network. It increases gradually according to the RTT. In this paper, an attempt is made to increase the performance of the TCP by introducing a two new technique called Adaptive Fair Share of Congestion window size (AFSCOW) and Smart Start (SS). AFSCOW can be used for the new TCP connections emerging in the end systems having same sender and receiver. For an emerging new application in the sender the window size can be determined as the total of the current window sizes of the existing application divided by the total number of application including the new emerging application. In Fast Start the number of TCP segments transmitted is not doubled but it is increased three times the older window size which shows the optimum use network resources within minimum RTT.

Key words: TCP, Congestion Window size, Link Utilization, Bandwidth Sharing, Round Trip Time (RTT).

I. INTRODUCTION

TCP (Transmission Control Protocol) (1) was designed to provide reliable end-to-end delivery of data over unreliable networks. In theory, TCP should be independent of the technology of the underlying infrastructure. In particular, TCP should not care whether the Internet Protocol (IP) is running over wired or wireless connections. In practice, it does not matter because most TCP deployments have been carefully designed based on assumptions that are specific to wired networks. Ignoring the properties of wireless transmission can lead to TCP implementations with poor performance. In wireless and Ad hoc networks, the principal problem of TCP lies in performing congestion control in case of losses that are not induced by network congestion. Since bit error rates are very low in wired networks, nearly all TCP versions nowadays assume that packets losses are due to congestion. Consequently, when a packet is detected to be lost, either by timeout or by multiple duplicated acknowledgements (ACK), TCP slows down the sending rate by adjusting its congestion window. Unfortunately, wireless networks suffer from several types of losses that are not related to congestion, making TCP not adapted to this environment. Numerous enhancements and optimizations have been proposed over the last few years to improve TCP performance over wireless and Ad hoc Networks. These improvements include infrastructure based WLANs (2), (3), (4), (5), mobile cellular networking environments (6), (7), and satellite networks (8), (9). It is noted that the following TCP versions: Tahoe, Reno, Newreno, and Vegas perform differently in Ad hoc networks (10). However, all these versions suffer from the same problem of inability to distinguish between packet losses due to congestion from losses due to the specific features of Ad hoc networks. For a survey about TCP versions (11) is referred. By examining the TCP's performance studies over wireless and Ad hoc network the following major problems are identified: (i) TCP is unable to distinguish between losses due to route failures and network congestion. (ii) TCP suffers from frequent route failures. (iii) the contention on the wireless channel. (iv) TCP unfairness. Either cross layer proposals or layered proposal can be used to improve the TCP's performance. our work is based on layered proposal. Until now, the fairness ratio issue of the flows emerging from same sender and receiver is not explored. This motivated us to carry out research in this issue.

II. BACKGROUND

A. Fairness

When network resources are shared by multiple TCP connections, there is a question of whether each connection gets a fair share of the resources. Users (Application programs) generally care about how much bandwidth they get, so one definition of fairness would be that each connection should get the same bandwidth. Therefore to provide fairness, TCP's congestion control algorithms must be altered such that it tends to give roughly equal windows to competing connections. This paper proposes a small modification to the way TCP increases its congestion window to provide fair share of available bandwidth and also to increase the performance (# of packets transmitted per RTT). The work is carried out for the TCP connections emerging from the same node.

B. Additive Increase Multiplicative Decrease.

TCP's congestion control increases the window additively (adding one packet at a time in the absence of loss), and decreases it multiplicatively (cutting it in half when a loss is detected).This means that if the resources are allocated unfairly to start with, all connections increase their allocation at the same rate, but a connection with a larger share will decrease more quickly than one with a smaller share. Therefore an additive increase multiplicative decrease policy tends to make the allocation which does not provide fair share. This issue is addressed in this paper. With the growth of the web, more and more traffic takes the form of short-lived connections. Often the entire amount of data to be sent fits on the network path (the bandwidth-delay product), meaning it could be transferred in one round-trip time. But the source has no way of knowing the capacity of the network beforehand, so to avoid congestion it gradually increases its window from one packet over several round-trip times. This is a frustrating under use of the network, and is a hard problem. To overcome this drawback this new start technique is used where the congestion window size is incremented. Because of this, obviously the capacity of link will be utilized to a maximum extend within small number of RTT.

III .AFSCOW

In traditional TCP congestion control, slow start results in poor performance because TCP spends more time in slow start. When time out occurs, the congestion window is initialized to one segment and the congestion threshold is set to half the present value. Each time an ACK is received, the congestion window is increased by one segment until the congestion window reaches the congestion threshold. This behavior is to slow, so that most of the connection can not use the maximum speed.

In this paper, we assume that there are more than one application which has the same sender and receiver end systems. If time out occurs for one or more applications then we are going to fix the congestion window size as the congestion window size of an application which has the small window size divided by number of applications for which timeout had occurred instead of starting from one segment. For example, consider two applications with window size of 512 and 32, within the same sender If time out occurs 32 is divided by 2 and window size of both applications are set to 16. By using the above method the number of segments transmitted in a particular number of RTT can be increased.

A. Performance Metric

The jain's fairness index is used to find out the fairness ratio. It is necessary that the available bandwidth must be shared equally among the competing flows. To verify the throughput fairness, the jain's fairness index can be calculated as follows

To assign a fairness index to a set of throughput (x_1, x_2, x_3 . . . . $x_{n)}$

$$f(x_1, x_2, x_3 \ldots . x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n\left(\sum_{i=1}^{n} x_i^2\right)} \quad [1]$$

The value of the fairness index will lie in between 0 and 1, where by the value 1 means that the measurement values $x_1 . . . . x_n$ are absolute equal. If only k of the n flows receive equal throughput and the remaining n-k users receive zero throughput, the fairness index is k/n.

B. Comparison of Existing and Proposed System

Based On Number of Rtt's

By assuming that the timeout occurs at Rtt7, according to the existing method the congestion window size is decreased to 1 and starts increasing from there as shown above. Since the timeout occurrence is considered for 2 application the window size for the two application starts from 1. According to this scheme the total number of segments transmitted for 7 Rtt's is equal to 1022

Total number of RTT=15

Application 1starts Timeout Occurred (assumption)

↓ ↓

{Rtt 8}

1 2 4 16 32 64 128 256 512

1 2 4 8 16 32

↑

Application 2 starts

Existing System

Ap1- 1 2 4 8 16 32 64 128

Ap2- 1 2 4 8 16 32 64 128

Total no of segments transmitted= 510

Table1. Fairness index of Existing system

Application program	No. of transmitted segments	Fairness index
1	758	0.8301
2	286	

Proposed System

Ap1 8 16 32 64 128 256
512 1024
AP2 8 16 32 64 128 256
512 1024

Total no of segments transmitted = 3056.

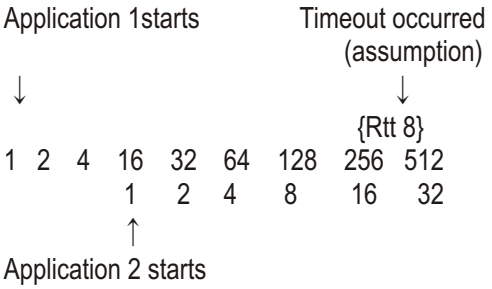
Table 2. Fairness index of Existing system

Application program	No. of transmitted segments	Fairness index
1	2543	0.9896
2	2071	

After the occurrence of the timeout, instead of starting the congestion window size from 1, the congestion window size is started from average window size of the application which is having the lower window size. Here Average window size represents congestion window size of an application which has the small window size divided by number of applications running in the sender. The time out occurs at 7th Rtt, during this stage window size of application 1 and 2 are 256 and 16 respectively. Out of these two window sizes 16 is the least window size, 16/2 = 8 can be used as the window sizes of the two applications to start the transmission after timeout as shown. According to this scheme the total number of segments transmitted for 7 Rtt's is equal to 3056. Table 1 and 2 shows the fairness index of existing system and proposed system. Number of transmitted segments in existing system for application program 1 is 2.65 times greater than that of proposed system. But in case of proposed system it is only 1.23 times greater. The fairness index in existing system is 08301 as compared to the proposed system for which the fairness index is 0.9896. From this it is clear that fairness index of proposed system is higher than that of existing system.

C. Comparison of Existing and Proposed System Based on Occurrence of Timeout on Exceeding the Maximum Load (512 Segments)

Here it is assumed that the link connected to the sender can carry only 512 segments, if the segments is increased beyond this limit then timeout occurs. That is the maximum load are capacity is assumed to be 512 segments. During Rtt 8 since the total number segments transmitted are 512+32= 544, which is greater than 512, timeout occurs according to the assumption.



Existing System

AP1 1 2 4 8 16 32 64 128
256
AP2 1 2 4 8 16 32 64 128
256

Total no of segments transmitted = 510.

Table 3. Fairness index of Existing system

Application program	No. of transmitted segments	Fairness index
1	1014	0.9157
2	542	

After the occurrence of time out the congestion window size of existing scheme is increased as usual staring from 1. When the total CWS is 512 the timeout occurs. This is shown in bolded numbers.

Proposed System

AP1 16 32 64 128 ***256*** 64
128 ***256*** 128
AP2 16 32 64 128 ***256*** 64
128 ***256*** 128

Total no of segments transmitted= 1120

Table 4. Fairness index of Existing system

Application program	No. of transmitted segments	Fairness index
1	1142	0.9365
2	670	

The congestion window size of the proposed scheme is shown above. The representation of Cws in bolded letter shows the occurrence of timeout. The timeout occurs there because according to the assumption maximum capacity is only 512 segments. From the comparison it is clearly revealed that the total number of segments transmitted in proposed scheme is far better than that of the existing scheme. This increases the performance of TCP. Table 3 and 4 shows the fairness index of existing system and proposed system. Number of transmitted segments in existing system for application program 1 is 1.87 times greater than that of proposed system. But in case of proposed system it is only 1.70 times greater. The fairness index in existing system is 0.9157 as compared to the

proposed system for which the fairness index is 0.9365. From this it is clear that fairness index of proposed system is higher than that of existing system.

IV. Simulation and Results

The proposed technique is implemented in ns2 Simulator. Performance of the TCP is compared with the proposed technique and existing scheme. Number of node is taken as 30. The TCP statistics for a simulation run of 100 seconds time period, number of nodes 30, Node placement – Uniform, Mobility Random way point. Free space propagation is used Source and destination pair is considered as node 5 and node 25 respectively. Channel capacity is chosen as 2 Mbit/sec. Dynamic source routing and IEEE 802.11a is used as the routing and MAC protocol. FTP is used as the application layer protocol. Packet size of 512, 1024, 2048 bytes are used. Simulation is conducted for 100 seconds. 10 simulation runs are made and the average value is taken as the data point for result. The comparison between the existing and proposed technique is shown in the pie chart in Fig. 1 and 2.

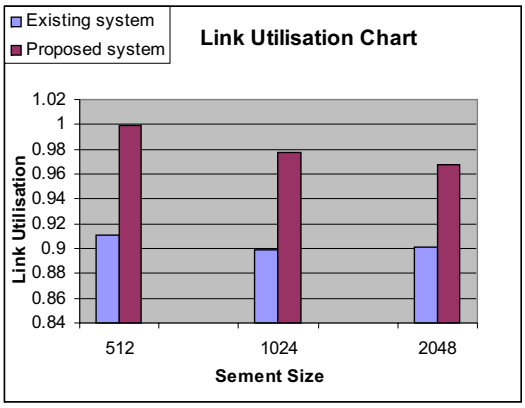


Fig. 1. Link Utilization Chart

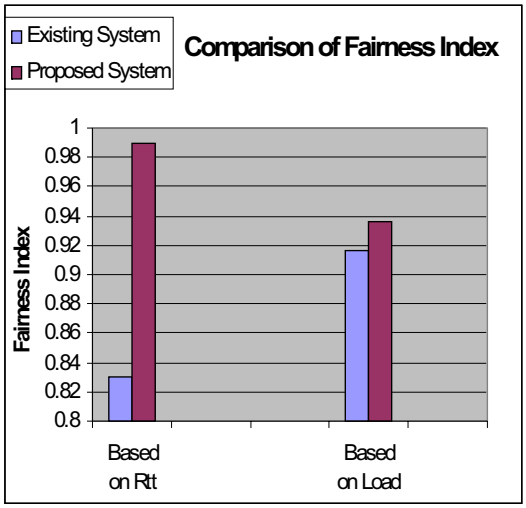


Fig. 2. Comparison of Fairness Index.

Link utilization = offered load/ maximum capacity. Link utilization is computed for packet size of 512, 1024, and 2048 bytes. If the link utilization is 1 then, the capacity of the link is used efficiently to 100 percentage. In the link utilization chart it can be seen that the link utilization of proposed system is greater than that of proposed system. Fairness index comparison chart reveals that the fairness ratio of proposed system is greater than that of existing system. Therefore the simulation result shows the same result with that of the numerical example discussed in section 3.

V. Smart Start (SS)

The slow start and congestion avoidance algorithms must be used by a TCP sender to control the amount of outstanding data being injected into the network. To implement these algorithms, two variables are added to the TCP per-connection state. The congestion window (cwnd) is a sender-side limit on the amount of data the sender can transmit into the network before receiving an acknowledgment (ACK), while the receiver's advertised window (rwnd) is a receiver-side limit on the amount of outstanding data. The minimum of cwnd and rwnd governs data transmission. In slow start after, the first successful transmission and acknowledgement of a TCP 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. In slow start even though, the window size is increased faster the connection takes much long time to reach the maximum utilization of bandwidth available in the link. To overcome this, a procedure such as increasing the window size either in the multiples of 3 or 4 is proposed rather than increasing the window size in multiples of 2 for each acknowledgement.

A. Assumptions

RTT=500 unit, Acknowledgement is received for each RTT, Bandwidth used is calculated by Used bandwidth = Sender Window size (SWS) * 3. The units of all the parameter are treated as general (it may be ms for RTT, # of bytes for Sender Window size, Kbps/Mbps for Used Bandwidth)

In the table 5 sender window size is increased by 2 for each RTT ie) Sender Window Size = Previous Sender Window Size *2. if the unit of RTT is considered as ms, then from the table it can be proved that that it takes 8 seconds till the TCP flow reaches 100MB/s at a RTT of 500ms.

Table 5. SWS in Multiples of 2

Sl.No.	Time (RTT)	Sender Window Size	Used Bandwidth
1	0	1	3
2	500	2	6
3	1000	4	12
4	1500	8	24
5	2000	16	48
6	2500	32	96
7	3000	64	192
8	3500	128	384
9	4000	256	768
10	4500	512	1536
11	5000	1024	3072
12	5500	2048	6144
13	6000	4096	12288
14	6500	8192	24576
15	7000	16384	49152
16	7500	32768	98304
17	8000	65536	196608

Table 6. SWS in Multiples of 3

Sl.No.	Time (RTT)	Sender Window Size	Used Bandwidth
1	0	1	3
2	500	3	9
3	1000	9	27
4	1500	27	81
5	2000	81	243
6	2500	243	729
7	3000	729	2187
8	3500	2187	6561
9	4000	6561	19683
10	4500	19683	59049
11	5000	59049	177147

In the table 3 sender window size is increased by 3 for each RTT ie) Sender Window Size = Previous Sender Window Size *3. If the unit of RTT is considered as ms, then from the table it can be proved that that it takes 5 seconds till the TCP flow reaches 100MB/s at a RTT of 500ms.In the table 7 sender window size is increased by 4 for each RTT ie) Sender Window Size = Previous Sender Window Size *4. If the unit of RTT is considered as ms, then from the table it can be proved that that it takes 4 seconds till the TCP flow reaches 100MB/s at a RTT of 500ms.

Table 7. SWS in Multiples of 4

Sl.No.	Time (RTT)	Sender Window Size	Used Bandwidth
1	0	1	3
2	500	4	12
3	1000	16	48
4	1500	64	192
5	2000	256	768
6	2500	1024	3072
7	3000	4096	12288
8	3500	16384	49152
9	4000	65536	196608

Table 8. Comparison of Time Vs SWS of 2,3,4.

Sl.No	SWS ^	Time *
1	2	8
2	3	5
3	4	4

SWS^ represents the magnitude at which the sender window size is increased.

Time * represents the maximum time taken by the connection to attain 100Mbps

By examining the data's given in table 8, it can be understood that, if the sender window size is increased in multiples of 3 0r 4 then the connection can attain the maximum link utilization in 3 or 4 times faster than that of the existing system.

VI. Conclusion

In this paper, we have presented a new technique called Adaptive Fair Share of Congestion window size (AFSCOW) for TCP to speed up short transfers such as web page downloads. Another technique called Smart Start (SS) technique is used to speed up the sending rate of the sender TCP to optimally use the available capacity of the link. These techniques enable the TCP connection to reduce the overhead of slow start by increasing the congestion window size fastly. It allows to increase the use of capacity in network to a optimum level. The key results are: Large improvements in the utilization of network bandwidth, improvement of TCP performance (Number of transmitted segments divided by RTT), Fair sharing of bandwidth related to the application which belongs to the same sender and receiver, before the references.

REFERENCES

[1] Transmission Control Protocol, 1981 RFC 793

[2] L. Andrew, S. Hanly, and R. Mukhtar, 2003, CLAMP: Differentiated capacity allocation in access networks, in Proc. of IEEE International. Performance Computing and Communications Conf., Phoenix, AZ, USA, pp. 451–458.

[3] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, 2006, A comparison of mechanisms for improving TCP performance overwireless links, IEEE Transactions on Networking, vol. 5, no. 6, pp. 756–769.

[4] H. Balakrishnan, S. Seshan, E. Amir, and R. Katz, 1995, Improving TCP/IP performance over wireless networks, in Proc. of ACM MOBIHOC, Berkeley, CA, USA, , pp. 2–11.

[5] A. V. Bakre and B.R. Badrinath, 1997 "Implementation and performance evaluation of indirect TCP," IEEE Transactions on Networking, Vol. 46, no. 3, pp. 260–278

[6] K. Brown and S. Singh, 1997, M-TCP: TCP for mobile cellular networks," ACM SIGCOMM Computer Communication Review, vol. 27,no. 5, pp. 19–43.

[7] H. Balakrishnan, S. Seshan, and R. Katz, 1995, Improving reliable transport and handoff performance in cellular wireless networks," ACM Wireless Networks, vol. 1, no. 4, pp. 469–481.

[8] Y. Tian, K. Xu, and N. Ansari, 2005, TCP in wireless environments: problems and solutions," IEEE Communications Magazine , vol. 43, no. 3, pp. S27-S32.

[9] J.-Y. L. Boudec, 2005, Rate adaptation, congestion control and fairness: a tutorial. http://ica1www.epfl.ch/ PS_files/LEB3132.pdf.

[10] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, 2003, The impact of multihop wireless channel on TCP throughput and loss, in Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03), vol. 3, pp. 1744-1753, San Francisco, Calif, USA.

[11] Luqing Yang , Winston K.G. Seah , Qinghe Yin, 2003 ,Improving fairness among TCP flows crossing wireless ad hoc and wired networks, Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing, pp 171-179 , Annapolis, Maryland, USA.

Ganeshkumar P.

Research scholar - Anna University, Chennai. He is a member of IEEE , ISTE and CSI. He had published several papers in International journals, IEEE international conferences and national conferences. Currently he is working as assistant professor in PSNA College of Engineering and Technology. His area of interest includes Distribution systems, Computer Network and Ad hoc Network.