COMPARATIVE ANALYSIS OF TCP CONGESTION CONTROL ALGORITHMS USING NETSIM

E.V.S.S.Vyshnavi, Assistant Professor, Department of ECE, G. Narayanamma Institute of Technology and Sciences (for Women).

Abstract

Congestion is a situation where the number of packets that a network can carry exceeds the capacity of the Network, which results in message traffic and thus slows down the data transmission rate. Congestion control is one of the most important issue in computer networks. There is a chance of network collapse if we do not use the proper congestion control algorithm. Therefore, congestion control is an effort to readjust the network performance to fluctuations in the traffic load without adversely affecting the user's perceived service quality. TCP controls the congestion by maintaining a congestion window, which indicates the maximum amount of data that can be sent into the network without being acknowledged. The main purpose of this paper is to analyze and compare the different congestion control algorithms Network Simulator tool (NetSim-Version 12).

Key Words: Congestion, Throughput, Delay, QoS.

Introduction

A computer network is a system in which multiple computers are connected to each other to share information and resources. During the last years, computer networks have experienced tremendous growth. More and more computers get connected to both private and public networks, the most common protocol stack used being TCP. Nowadays it is difficult to identify the congestion control algorithms that are currently implemented by various machines in Internet. The TCP header does not provide any information about them. Congestion control is an effort to readjust the performance of a network to fluctuations in the traffic load without adversely affecting the user's perceived service quality.

TCP controls the congestion by maintaining a congestion window, which indicates the maximum amount of data that can be sent into the network without being acknowledged. There are different congestion control algorithms for TCP protocols namely: Tahoe, Reno, BIC, CUBIC. All the algorithms suggest mechanisms for determining when to retransmit a packet and how it should update the congestion window.

Literature Survey

During the last years, computer networks have experienced tremendous growth. More and more computers get connected to both private and public networks, the most common protocol stack used being TCP. Nowadays it is difficult to identify the congestion control algorithms that are currently implemented by various machines in Internet. The TCP header does not provide any information about them. Congestion control is an effort to readjust the performance of a network to fluctuations in the traffic load without adversely affecting the user's perceived service quality.

Various studies have been conducted to explore about congestion and algorithms to control the congestion, such as Mohamed Nj and Burairah Hussin [1] made an analytical review of Network Congestion occurrence causes and the fundamentals of the existing control solutions as available and studied from some former and recent networks publications. A. Esterhuizen and A.E. Krzesinski[2] have compared all the congestion control algorithms theoretically based on different parameters. Habibullah Jamal and Kiran Sultan[3] have made a detailed analysis of how the network is effected and performance degradation of the network due to the congestion. In [4] scheme that determines the size of congestion window each time a new acknowledgment is received instead of employing slow start/congestion avoidance approach is proposed. [5] Represents exploratory study of TCP congestion control, modern implementations of TCP through extensive simulations and the performance characteristics of four representative TCP congestion control algorithms.

Implementation

Network performance is fundamentally measured in two ways - Throughput and Delay. Throughput is the measure of the number of packets that are transferred from the source to the destination successfully, calculated as bits per second. Whereas the throughput in a network is the measure of the data that is transferred from the source to the destination within a given timeframe.

Delay in a network is the measure of time taken for data to reach the destination from the source. In a network, many factors affect the average throughput and average delay of the network. Packet loss and delay are related to throughput. Minimizing these factors can increase the throughput of the network thus increasing the performance of the network.

As the number of applications in a network increase, the bandwidth available will be divided for the different applications. Since the bandwidth for each application decreases, the throughput also decreases. And the throughput of the network also depends on the type of topology chosen as it indicates the number of devices connected to each other which leads to increase traffic and decreases the speed of communication.

Similarly for the network delay, if there are fewer applications in the network, then there would be more bandwidth or faster connection. More bandwidth leads to faster communication between the source and destination or among the devices. This means less delay. Therefore, it can be aid that delay drives throughput.

In the network chosen, there are different link speeds and different numbers of applications or users in the network. Link speeds are chosen as 30Mbps and 60Mbps. The number of applications in the network is chosen as 6 applications and 12 applications. Totally three combinations are selected. They are

1. Number of applications - 6 and Link speed - 30 Mbps

2. Number of applications - 6 and Link speed - 60 Mbps

3. Number of applications - 12 and Link speed - 30 Mbps

Steps

- **Step-1:** A network is built using routers, switches and end devices like PC's (wired and wireless).

- **Step-2:** The devices are connected accordingly. How congestion is taking place and how much throughput is being received is analyzed by varying number of applications within the same network. IP Addresses and related properties for the nodes and routers are provided by the software internally.

- **Step-3:** Link properties are given as uplink speed-30mbps, downlink speed-30mbps, propagation delay-5µs. These link metrics are changed for all the links in the network.

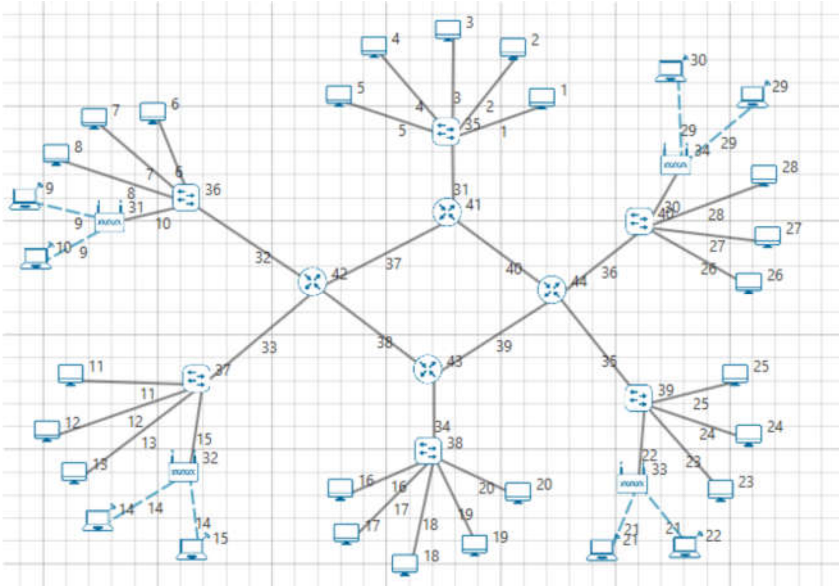

Fig.1 Network Topology

- **Step-4:** Uplink speed and downlink speed for the links between the routers is given as 100mbps.
- **Step-5:** Firstly, 6 applications are given with an arrival time of 400ms.
- **Step-6:** The type of application taken is CBR (constant it Rate) means packets of constant size are generated at constant inter arrival time.

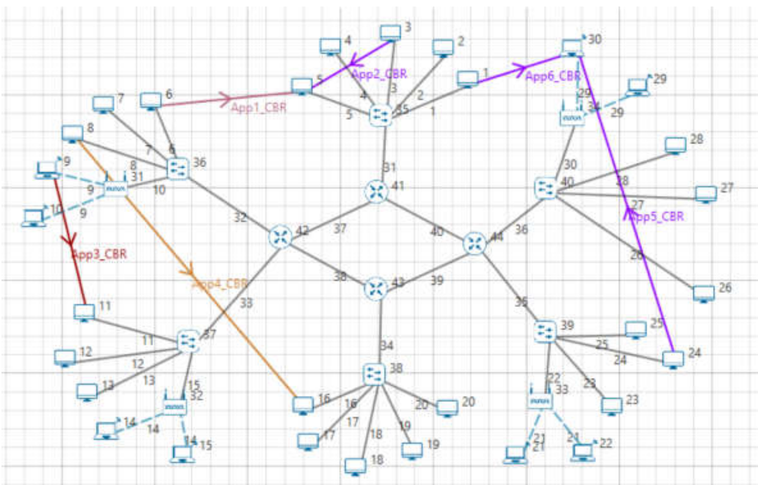

Fig. 2 Topology for 6 applications

- **Step-7:** Each algorithm Tahoe, Reno, BIC, CUBIC are applied at the congestion control algorithm option in the application window to all the 6 applications and also to the routers, throughput is measured for all the algorithms.

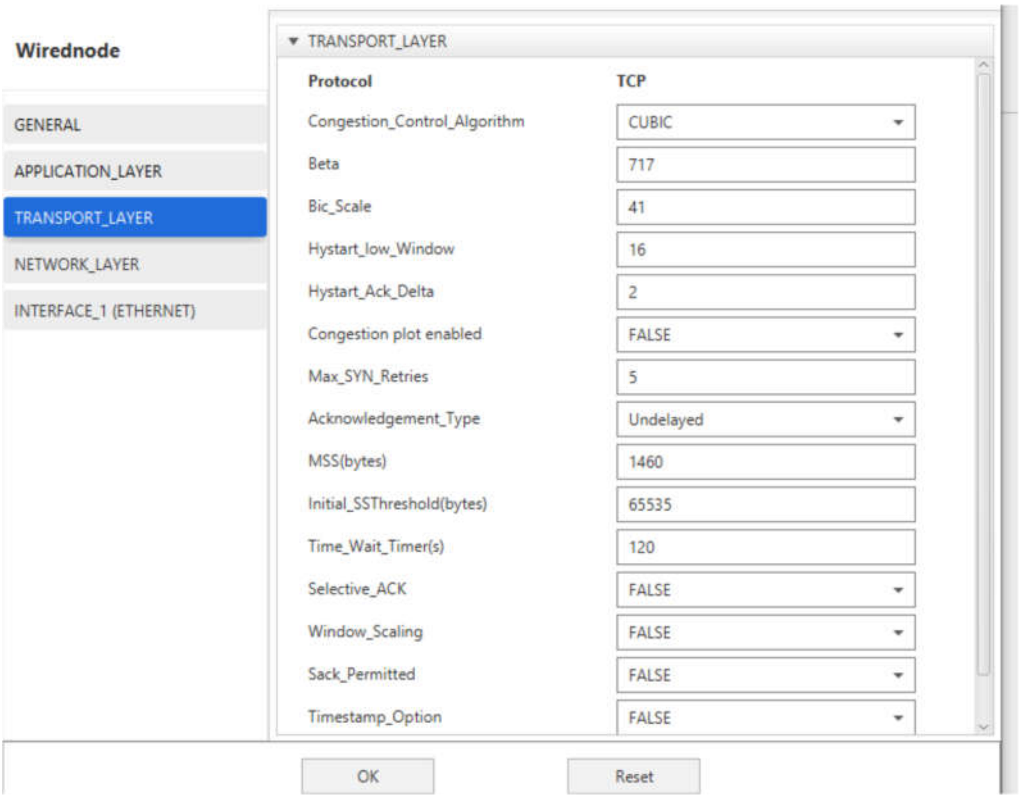

Fig. 3 Transport layer window 16

• **Step-8:** Then again uplink speed and downlink speed are changed to 60ms and checked the throughput. Since the given numbers of applications are less, the throughput received will be more and throughput is also increased due to increase in speed in link properties.

• **Step-9:** The throughput received for speed 60ms is little more than throughput received for 60ms speed.

• **Step-10:** The same process is repeated for 12 applications. The number of applications in the network increased, hence the congestion is increased. Therefore, the throughput received here is less than the throughput received in 6 applications.

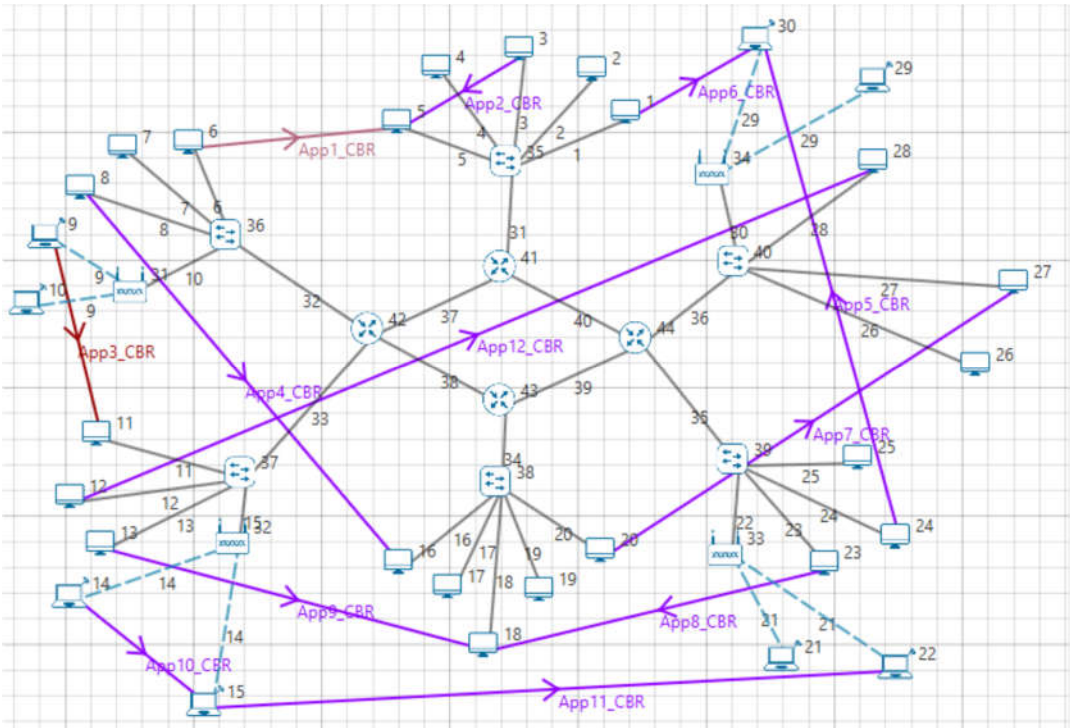

Fig. 4 Topology for 12 applications

Analysis of Each Algorithm For 6 Applications

The analysis of each algorithm like Tahoe, Reno, BIC, CUBIC with 6 applications and different link speed like 30Mbps and 60 Mbps are generated and are few of the tables given below are for reference

- **Tahoe**

Table 1: Analysis of Tahoe with link speed 30Mbps for 6 applications

Application Id	Payload generated (bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	9101640	7.281312	2997936.728	1269.8663
2	36500000	22996460	18.397168	2181024.402	216.450997
3	36500000	3014900	2.41192	4610993.011	4524.264952
4	36500000	11554440	9.243552	3835964.487	820.004852
5	36500000	1959320	1.567456	4827901.232	7133.185011
6	36500000	1084780	0.867824	4695243.495	11654.85295

- **Reno**

Table 2 :Analysis of Reno with link speed 60Mbps for 6 applications

Application Id	**Payload generated (bytes)**	**Payload Received(bytes)**	**Throughput (Mbps)**	**Delay (Microsec)**	**Jitter (Microsec)**
1	36500000	20647320	16.51786	1760506	523.2095
2	36500000	34699820	27.75986	436614.6	318.168
3	36500000	3089360	2.471488	4454054	4410.573
4	36500000	22333620	17.8669	3155239	521.56
5	36500000	1149020	0.919216	4657584	12425.32
6	36500000	1819160	1.455328	5129933	6829.289

- **BIC**

Table 3: Analysis of BIC with link speed 30Mbps for 6 applications

Application Id	**Payload generated (bytes)**	**Payload Received(bytes)**	**Throughput (Mbps)**	**Delay (Microsec)**	**Jitter (Microsec)**
1	36500000	13734220	10.987376	2307397.66	806.362507
2	36500000	21083860	16.867088	2695896.04	304.245828
3	36500000	2782760	2.226208	4413263.03	4994.319388
4	36500000	12474240	9.979392	3867125.41	799.544724
5	36500000	2229420	1.783536	4437301.34	6266.935701
6	36500000	1022000	0.8176	5992293.48	11840.31102

- **Cubic**

Table 4: Analysis of Cubic with link speed 30Mbps for 6 applications

Application Id	**Payload generated (bytes)**	**Payload Received(bytes)**	**Throughput (Mbps)**	**Delay (Microsec)**	**Jitter (Microsec)**
1	36500000	12618780	10.095024	2308196.6	842.02125
2	36500000	21664940	17.331952	2502815.4	262.06661
3	36500000	2826560	2.261248	4705426.6	4843.9778
4	36500000	18794580	15.035664	3150339.8	377.08295
5	36500000	2057140	1.645712	3877593.7	6780.5219
6	36500000	1140260	0.912208	5936585.8	10590.441

Table 5: Analysis of Cubic with link speed 60Mbps for 6 applications

Application Id	Payload generated(bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	21283880	17.0271	1843799	472.8965
2	36500000	35197680	28.15814	347276.6	288.978
3	36500000	3048480	2.438784	4612088	4468.359
4	36500000	20753900	16.60312	2595558	528.0032
5	36500000	1779740	1.423792	4351481	7925.58
6	36500000	1255600	1.00448	5294351	9540.373

The analysis of each algorithm with 12 applications and 30Mbps link speed are generated and are given below

- **Tahoe**

Table 6: Analysis of Tahoe with link speed 30Mbps for 12 applications

Application Id	Payload generated (bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	9130840	7.304672	3818975	1121.454
2	36500000	23234440	18.58755	1874779	283.1872
3	36500000	1207420	0.965936	4434941	11738.02
4	36500000	9768860	7.815088	4574647	1026.126
5	36500000	930020	0.744016	4431864	15311.64
6	36500000	611740	0.489392	5802164	19933.76
7	36500000	13167740	10.53419	3629347	667.1205
8	36500000	7135020	5.708016	4688603	1522.727
9	36500000	1045360	0.836288	4990001	13482.7
10	36500000	442380	0.353904	5774631	28576.98
11	36500000	14191200	11.35296	3479720	596.6652
12	36500000	6120320	4.896256	4716282	1830.078

- **Reno**

Table 7: Analysis of Reno with link speed 30Mbps for 12 applications

Application Id	Payload generated (bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	9468100	7.57448	3740970	1077.269
2	36500000	25230260	20.18421	1690074	224.2642
3	36500000	1257060	1.005648	4040138	11279.57
4	36500000	10646320	8.517056	4163587	925.0044
5	36500000	778180	0.622544	4674697	18483.21
6	36500000	643860	0.515088	5495807	18663.85
7	36500000	9335240	7.468192	4133025	1081.928
8	36500000	12335540	9.868432	3842092	741.581
9	36500000	8660720	6.928576	4134912	1202.025
10	36500000	921260	0.737008	4787663	15112.58
11	36500000	658460	0.526768	5919840	19112.06
12	36500000	13808680	11.04694	3954717	634.5314

- **BIC**

Table 8: Analysis of BIC with link speed 30Mbps for 12 applications

Application Id	Payload generated (bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	12650900	10.12072	3823345	785.2148
2	36500000	19926080	15.94086	1762194	412.8618
3	36500000	1514020	1.211216	5135866	9181.228
4	36500000	11849360	9.479488	3917387	863.6712
5	36500000	633640	0.506912	4166836	22714.06
6	36500000	541660	0.433328	6156640	21514.98
7	36500000	8824240	7.059392	3930570	1234.318
8	36500000	10606900	8.48552	4076955	997.7343
9	36500000	8792120	7.033696	4134740	1251.807
10	36500000	865780	0.692624	4727319	16412.69
11	36500000	670140	0.536112	4818230	19052.03
12	36500000	17641180	14.11294	3046468	507.2304

- **Cubic**

Table 9: Analysis of Tahoe with link speed 30Mbps for 12 applications

ApplicationnId	Payload generated(bytes)	Payload Received(bytes)	Throughput (Mbps)	Delay (Microsec)	Jitter (Microsec)
1	36500000	12738500	10.1908	3629499	701.3942
2	36500000	22543860	18.03509	1646129	299.6712
3	36500000	1487740	1.190192	4957720	9458.533
4	36500000	10117800	8.09424	4193951	982.5005
5	36500000	534360	0.427488	4034382	26220.26
6	36500000	681820	0.545456	5414069	17822.47
7	36500000	13516680	10.81334	3575680	719.1204
8	36500000	11704820	9.363856	4252041	799.9029
9	36500000	8079640	6.463712	4319468	1318.055
10	36500000	1089160	0.871328	4418765	12973.25
11	36500000	481800	0.38544	5859040	25769.97
12	36500000	12339920	9.871936	3900718	749.3723

On comparing the throughput in the topology with 6 applications and with the different link speeds i.e., 30Mbps and 60Mbps, the graph of average throughput of the network in different types of congestion control algorithms is plotted.

- **Analysis:** As the link speed increases, the average through also increases i.e., a greater number of packets are transmitted.

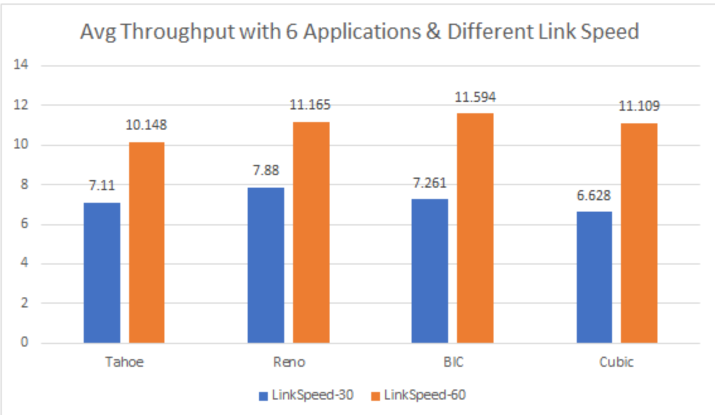

Fig. 5 Average throughput with different link speed

On comparing the throughput of the above topologies i.e. with 6 applications and 12 applications in the network with the same link speed i.e. 30Mbps, the graph ofaverage throughput of the network in different types of congestion control algorithms is plotted.

- **Analysis:** As the number of applications are increased the average throughput of all the algorithms are reduced. To control the congestion number of applications in a network should be less.

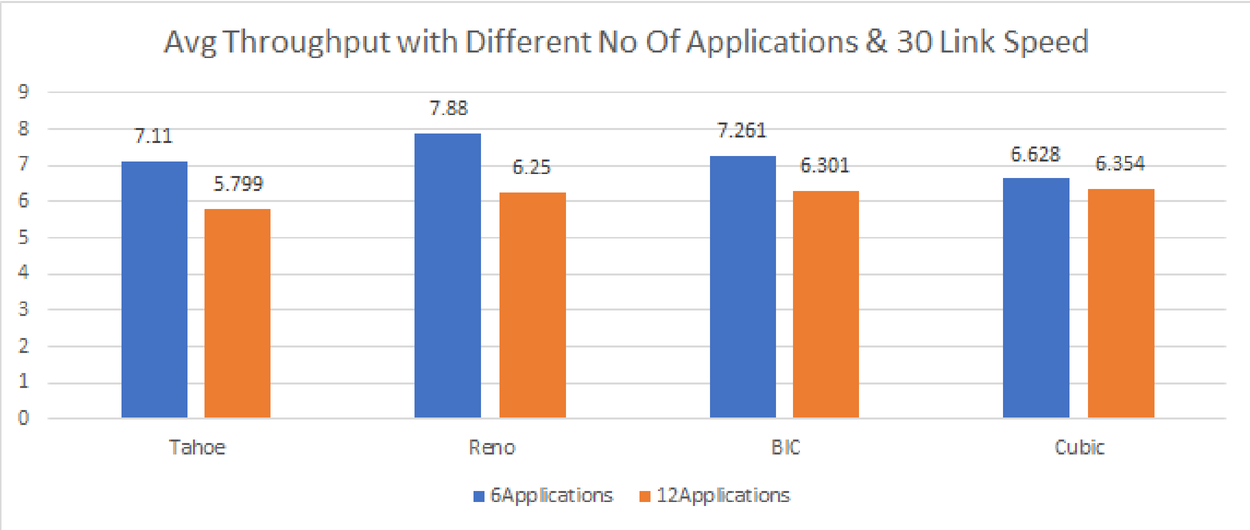

Fig. 5 Average throughput with different number of applications

On comparing the throughput in the topology with 6 applications and with the different link speeds i.e., 30Mbps and 60Mbps, the graph of average throughput of the network in different types of congestion control algorithms can be plotted as

- **Analysis:** As the speed of link is increased then the time required for the packet is reduced and thus delay is reduced. Higher the link speed lesser is the delay.

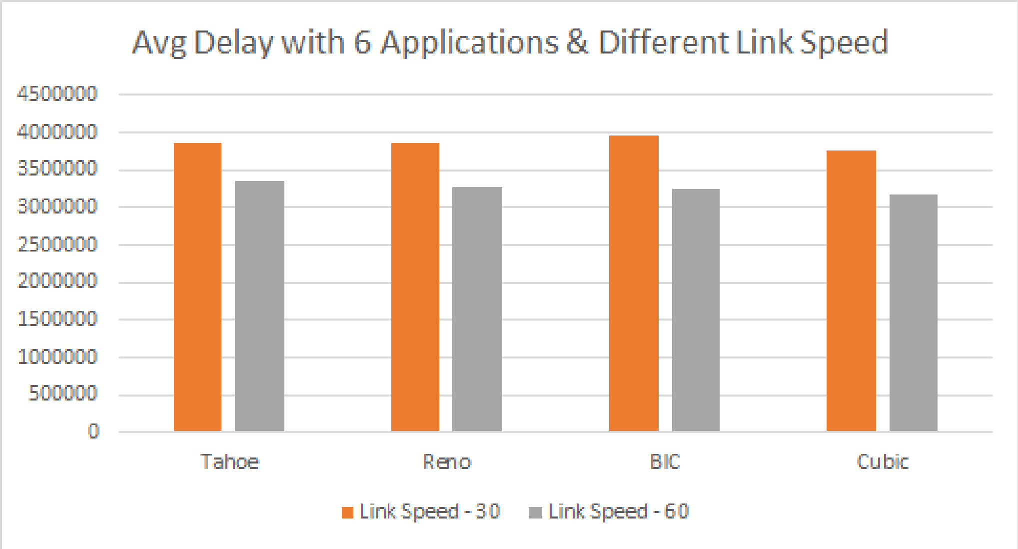

Fig. 6 Average delay with different link speed

On comparing the delay of the above topologies i.e., with 6 applications and 12 applications in the network with the same link speed i.e. 30Mbps, the graph of average delay of the network in different types of congestion control algorithms are plotted.

- **Analysis:** If the number of Applications in a network is increased then the delay inthe network also increases.

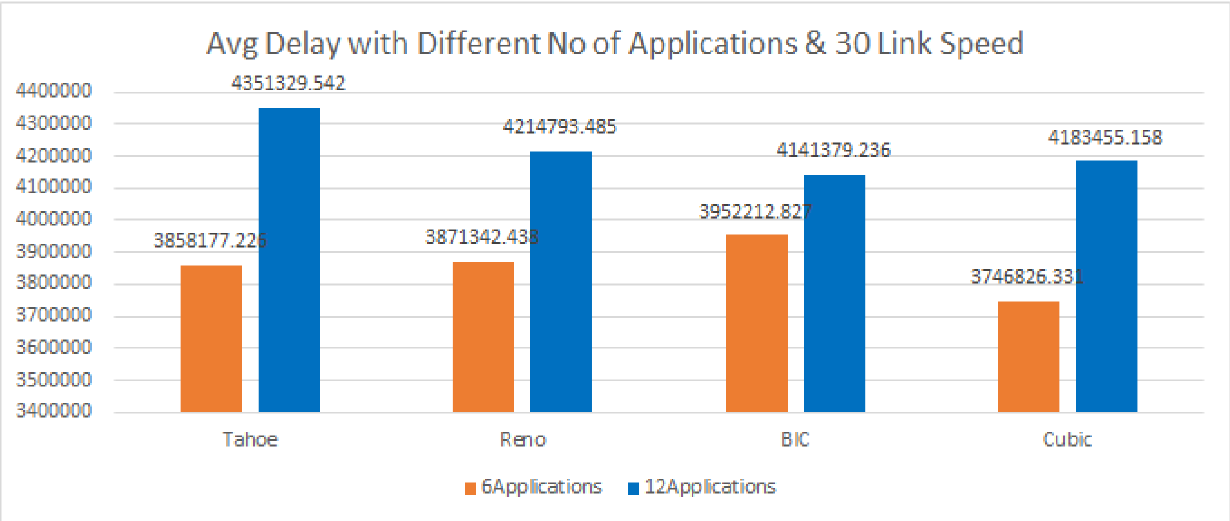

Fig. 7 Average delay with different no. of applications

- **Comparing All the Algorithms Based on Average Throughput**

On comparing the average throughput for all the algorithms cubic is having thehighest throughput.

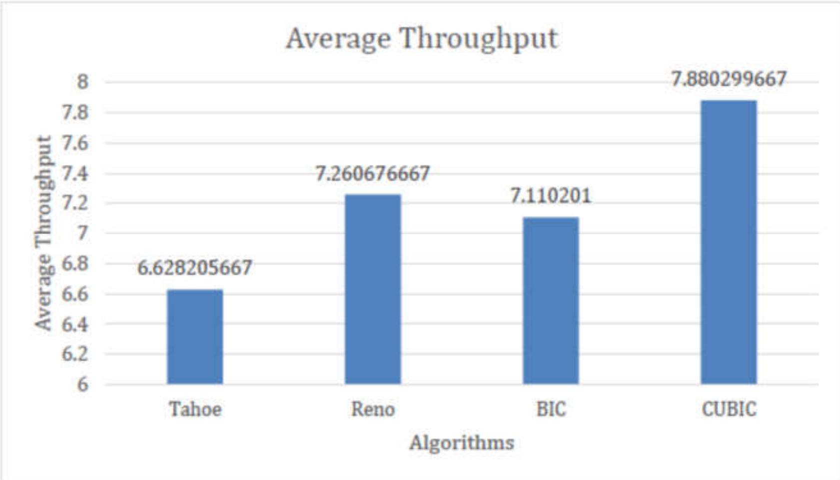

Fig. 8 Average Throughput for different congestion control algorithms

- **Comparing All the Algorithms Based on Average Delay**

On comparing the average delay for all the algorithms cubic is having the lowest delay.

Fig. 9 Average Delay for different congestion control algorithms

Future Scope & Conclusion

In this Paper, we have made simulations to understand the congestion control algorithms and can be implemented for real-time networks. In our simulation we considered IEEE 802.11 standard, we can extend our study to all other IEEE standards. Here we have compared Tahoe, Reno, BIC, CUBIC algorithms as Netsim can support these algorithms only, we can compare other new algorithms like Vegas.

Some of the algorithms show better response and some of them show poor responsiveness to changing network conditions and network utilization. Although there are various protocols and algorithms that have been used, referring to figure 5.5 and figure 5.6 we can conclude that TCP CUBIC is more effective by providing high throughput and less delay on average compared to Tahoe, Reno and BIC.

REFERENCES

[1] Mohamed Nj and Burairah Hussin, "Understanding Network Congestion Effects on Performance - Articles Review", JATIT & LLS, Vol.92. No.2, 31st October 2016.

[2] A. Esterhuizen and A.E. Krzesinski, "TCP Congestion Control Comparison", University of Stellenbosch, 7600 Stellenbosch, South Africa, Volume 3, 2006

[3] Habibullah Jamal and Kiran Sultan, "Performance Analysis of TCP Congestion Control Algorithms", International Journal of Computers and Communications, issue 1, Volume 2, 2008.

[4] M. Allman, V. Paxson and W. R. Stevens, "TCP Congestion Control," RFC 2581, 1999.

[5] IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834, p-ISSN: 2278-8735.Volume 5, Issue 6 (Mar. - Apr. 2013), PP 74-79.

[6] Singh, G., Kumar, D. and Kaur, A., 2009, "Simulation based comparison of performance metrics for various TCP extensions using NS-2", IEEE Sponsored International Conference on Innovative Technologies, vol. 9, pp. 106-110.

[7] K. Fall and S. Floyd, "Simulation Based Comparisons of Tahoe, Reno and SACK TCP," ACM SIGCOMM Computer Communication Review, Vol. 26, No. 3, July 1996.

[8] Bogdam Moraru, Flavius Copaciu, Gabriel Lazar, Virgil Dobrota, "Practical Analysis of TCP Implementations: Tahoe, Reno, New Reno", Technical University of Cluj-Napoca, January 2003.

[9] Hui Zhang, Zhengbing Bian, "Evaluation of Different TCP Congestion Control Algorithms Using Ns-2", Spring 2002.

[10] J. Padhye, V. Firoiu, D. F. Towsley and J. F. Kurose, "Modeling TCP Reno performance: a simple model and its empirical validation", *IEEE/ACM Trans. Netw*, vol. 8, pp. 133-145, Apr. 2000.