Performance Analysis of TCP Variants

Darshit Saraiya and Rohan Subramanian
College of Computer and Information Science
Northeastern University
saraiya.d@husky.neu.edu subramanian.ro@husky.neu.edu

Abstract

This paper uses simulations to analyze and compare the performances of different TCP variants – Tahoe, Reno, New Reno, Vegas and SACK under varying network parameters. The simulations are carried out using NS-2 network simulator. The behavior of TCP variants under varying load factors and their effects on the throughput and packet loss is discussed in this paper. On the basis of the experiments conducted, we can see that TCP Vegas has the best performance as compared to other variants.

1. Introduction

TCP or Transmission Control Protocol is a connection oriented end-to-end protocol providing reliable data transfer. In this paper, we illustrate the performances of different TCP variants to find out which one is better under varying conditions. The different variants of TCP used in the experiments are TCP Tahoe, Reno, New Reno and Vegas.

The original TCP did not provide the desired performance even though it worked well and was well-known for its reliability. TCP Tahoe executes slow-start and congestion avoidance mechanism. TCP Reno, in addition to this, implements fast recovery algorithm. New Reno is an improved version and implements fast recovery for partial acknowledgements as well. TCP Vegas is a very effective variant of TCP. It implements packet retransmission on basis of the acknowledgements and RTT.

We've performed three experiments and analyzed the results helping us to understand the performances of different variants in real time.

2. Methodology

To perform the experiments, we've used the network simulator (NS-2) to find out the average throughput, packet loss and latency. This data is compared for all the four variants of TCP.

The network topology consists of six nodes – N1 to N6 – with a bandwidth of 10 Mbps and full duplex. The propagation delay is set to 10ms.

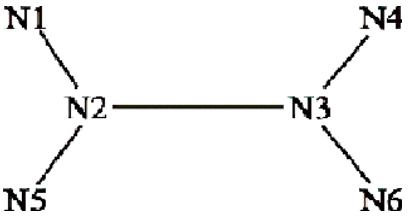


Figure 1. Network topology for NS2 simulation

For experiment 1 and 3 TCP flows are set from N1 to N4 with a sink at N4. For experiment number 2, a CBR source is added at node 2 and a sink at node 3.

Two TCP streams are added – one from node 1 to node 4 and the other from node 5 to node 6. The CBR rates are varied from 1 to 10 Mbps.

The results are represented in a graphical format and analyzed for different parameters to study the performances of different TCP variants.

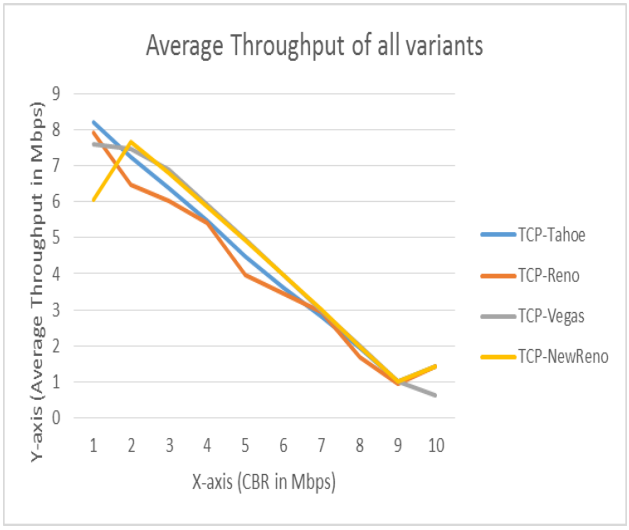
3. Results

3.1 Experiment 1:

The experiment is performed to analyze the performance of TCP variants (Tahoe, Reno, New Reno, and Vegas) under the influence of Constant Bit Rate (CBR) flow. The network topology of fig1 is used with CBR source at N2 and a sink at N3. A single TCP stream is added from N1 to a sink at N4.

In this experiment, all the TCP variants are compared by calculating the Average packet losses, throughput and latency of each TCP variant with a CBR flow whose bandwidth is varied from 1- 10 Mbps.

The results of the experiment are graphically plotted as follows:



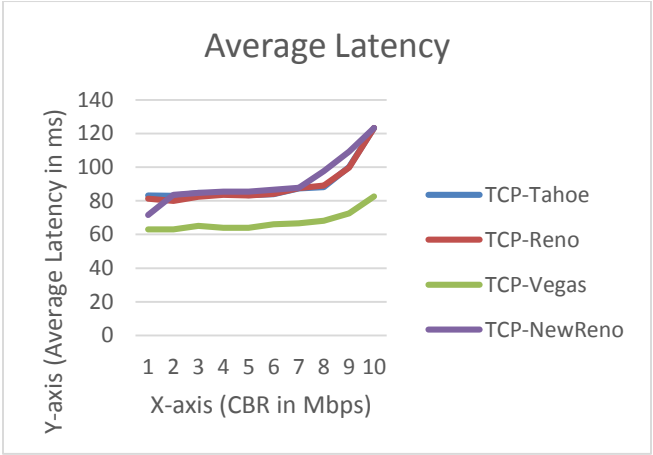
Formula Used:

Average Throughput = 8 * Number of packets received/ Duration of experiment

From the graph of Average throughput of all variants it is evident that the throughput for TCP Vegas is the highest among all variants. This is because Vegas induces congestion to learn the available network capacity, a Vegas source anticipates the onset of congestion by monitoring the difference between the rate it is expecting to see and the rate it is actually realizing. Vegas' strategy is to adjust the source's sending rate (congestion window) in an attempt to keep a small number of packets buffered in the routers along the transmission path. Hence, due to this unique property Vegas performs the best among all.

Formula Used:

Average Latency = Sum of Packet Latency / Number of Packets



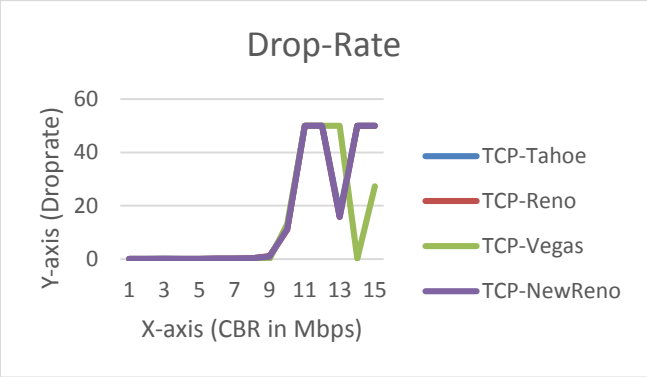
From the graph of Average Latency it is evident that latency of Vegas is lowest among all the variants. TCP Vegas measures and controls the amount of extra data in network where "extra data" is the data, source would not transmit trying to match the available bandwidth of the network. If source is sending too much extra data it causes long delays. TCP Vegas uses congestion avoidance to avoid this situation.

When TCP Vegas transmits a packet, it reads and stores the system clock with that packet id. When an

ACK is received, TCP Vegas reads the clock again to compute the RTT.

Formula Used:

Average Packet Drop Rate = 100 * (Packets Dropped/ Packets Sent)



From the graph it is evident that TCP Vegas has the fewest packet drops. This is because TCP Vegas overcomes the problem of requiring enough duplicate acknowledgements to detect a packet loss. It has the mechanism to detect congestion even before packet losses occur. Hence, due to this unique property TCP Vegas has packet drops.

Around the bottleneck capacity when CBR is 10-15 Mbps the packet drop is suddenly very high for all the variants. When CBR is set to 14Mbps the packet drop for Vegas is almost zero and for other variants is almost 50 percent.

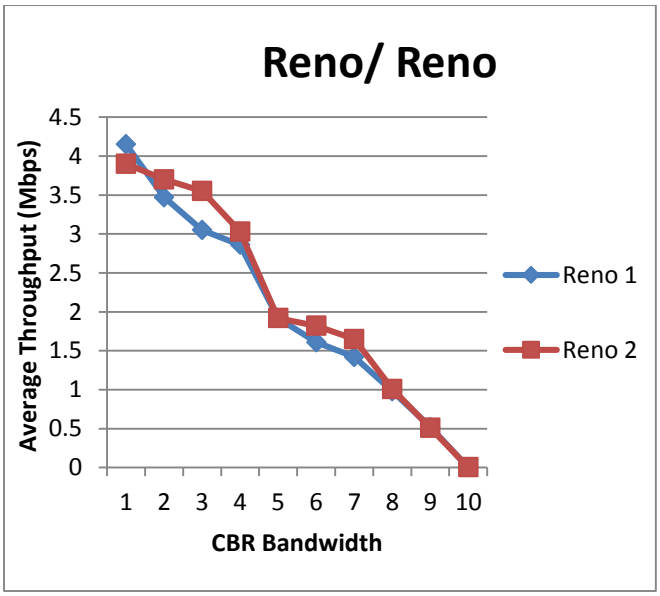
After analyzing the results of experiment 1 we come to the conclusion that TCP Vegas stands out in all aspects (Average Throughput, Average Latency and Drop Rate) among other TCP variants used.

The TCP Vegas is a modified version of Reno. It works on the proactive measures to control congestion rather than reactive measures. It uses an algorithm to check for timeouts. It also overcomes the problem of requiring enough duplicate acknowledgements to detect a packet loss. It also uses a slightly modified slow start mechanism. It has the mechanism to detect congestion even before packet losses occur, but it also retains the other mechanisms of Reno and Tahoe. Overall, the Vegas has a new retransmission mechanism, a modified slow start algorithm and congestion avoidance scheme.

3.2 Experiment 2:

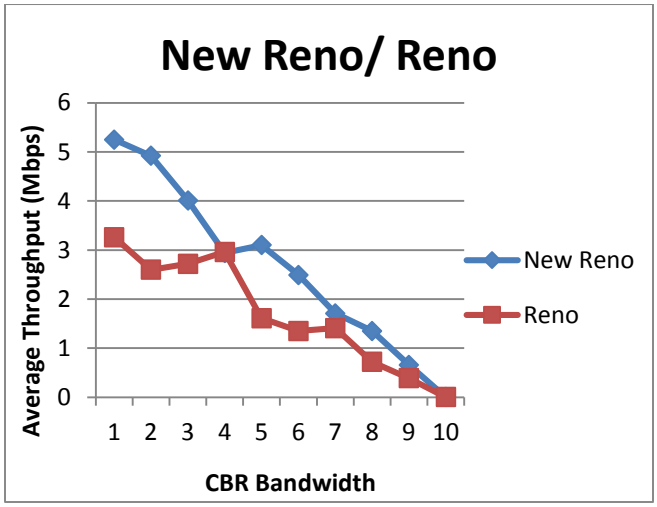
This experiment is aimed at analyzing the fairness between different TCP variants. Three flows are started for this experiment. CBR from N2, TCP from N1 to N4 and another TCP flow from N5 to N6. A sink is added at N3. Experiments are conducted for the following pairs of TCP variants – Reno/ Reno, New Reno/ Reno, Vegas/ Vegas and New Reno/ Vegas.

Reno/ Reno



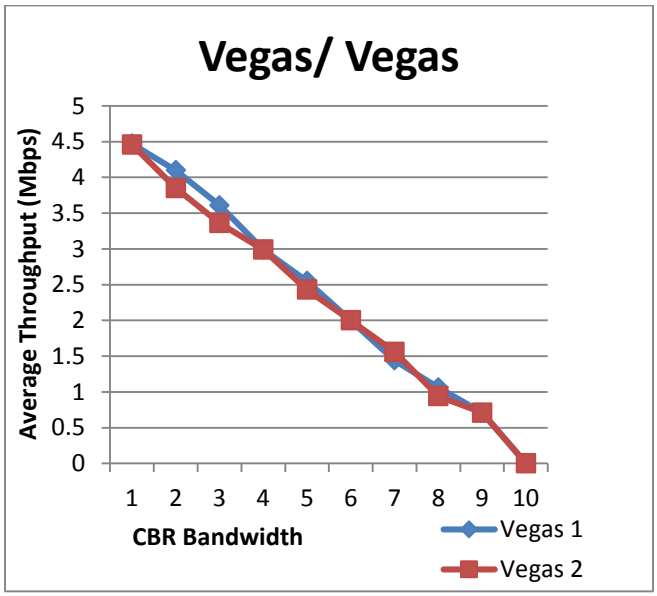
As seen from the figure above, in a combination of Reno/ Reno, the average throughput and packet loss are comparable as both the flows use the same variant. Such a combination could be used in a network where equal traffic is necessary.

New Reno/ Reno



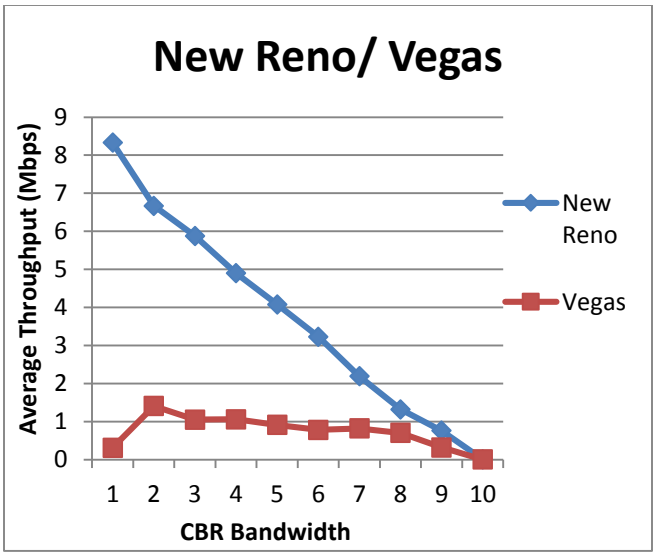
As seen from the figure, in a combination of New Reno and Reno, the former has a better throughput utilization of the channel as compared to Reno. This is primarily because, New Reno implements fast recovery even for partial acknowledgements. As a result, New Reno will have an advantage and so is unfair as compared to Reno/ Reno.

Vegas/ Vegas



The graph shows that the average throughput and utilization of the bandwidth is comparable. TCP Vegas uses a very efficient bandwidth and as a result, it detects congestion very early giving out high throughput. The same can be inferred from the graph. This combination can be used where equal traffic flow is required in a network.

New Reno/ Vegas



TCP Vegas is more efficient and has a better performance. But, in combination with New Reno, its performance takes a hit. It doesn't receive a fair bandwidth as Vegas reduces its sending rate before New Reno. New Reno takes up a majority of the available bandwidth.

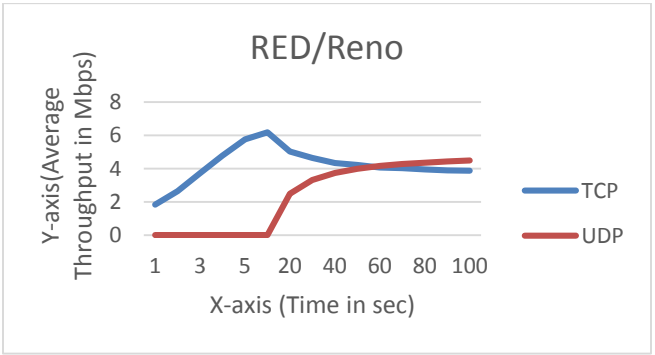
3.3 Experiment 3:

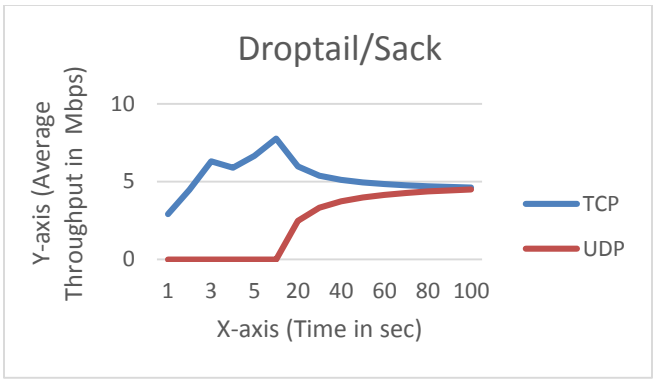
The experiment is performed to analyze influence of different queuing mechanisms like DropTail and Random Early Detection (RED). These mechanisms are applied on TCP variants SACK and Reno. The network topology of fig 1 is used with one TCP flow from N1 to N4 and CBR/UDP flow from N5 to N6. The CBR/UDP flow is started once the TCP flow is ready.

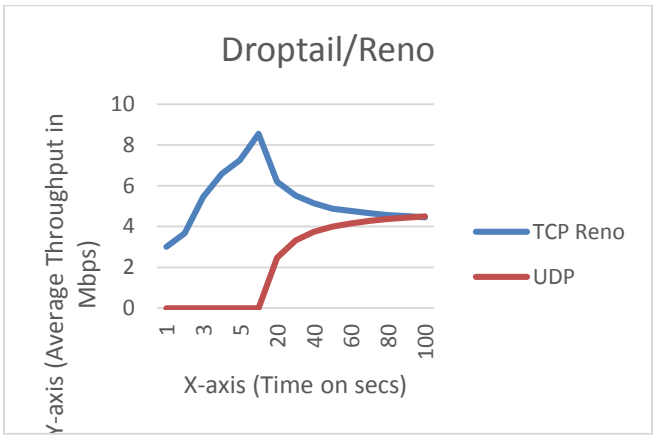
DropTail is a simple queue mechanism that is used by the routers when packets are needed to be dropped. In this mechanism each packet is treated identically and when queue is filled to its maximum capacity the newly incoming packets are dropped until queue has sufficient space to accept incoming traffic.

Random Early Detection (RED) is a congestion avoidance queuing mechanism .It is active queue management mechanism. It operates on the average queue size and drop packets on the basis of statistics information. If the buffer is empty all incoming packets are acknowledged. As the queue size increase the probability for discarding a packet also increase. When buffer is full probability becomes equal to 1 and all incoming packets are dropped.

In all the experiments the CBR is set to 5 Mbps.







Does each queuing discipline provide fair bandwidth to each other?

No, DropTail does not provide fair bandwidth. When the queue is full it simply drops the incoming packets which prevents fair utilization of bandwidth. Over a course of time the average throughput of UDP and TCP is almost same.

RED provides fair bandwidth compare to DropTail. Initially TCP utilizes maximum bandwidth but when CBR is initialized, the TCP throughput reduces over a course of time and CBR utilizes the predefined bandwidth which is set to 5 Mbps.

How does the end-to-end latency for flows differ between DropTail and RED?

The number of packets dropped in DropTail are more than that of RED. This is because DropTail drops the packets when the queue is full whereas RED drops

the packets earlier than it would have to, so as to notify the source that it should decrease its congestion window sooner than it would normally have to.

Hence, the end-to-end latency for DropTail is higher compared to that of RED.

How does TCP flow react to creation of CBR flow?

From the graphs it is evident that when the CBR flow starts the throughput of TCP reduces due to congestion in the network.

Is RED a good idea while dealing with SACK?

Yes, RED is a good idea while dealing with SACK because before the start of CBR, the throughput for SACK is excellent showing full utilization of bandwidth. RED also prevents bias against bursty traffic which makes it an asset while dealing with SACK.

4. Conclusion

Comparison between the different TCP variants shows that TCP Vegas is the most efficient. The percentage of packet drops is less as compared to the other variants.

5. References

1. http://nile.wpi.edu/NS/
2. http://www.isi.edu/nsnam/ns/tutorial/index.html
3. http://www.ccs.neu.edu/home/cbw/4700/papers/tcp-sim.pdf
4. http://www.roman10.net/drop-tail-and-redtwo-aqm-mechanisms/
5. ftp://ftp.cs.princeton.edu/techreports/2000/616.pdf
6. http://www.ijcaonline.org/volume20/number6/pxc3873294.pdf