Performance Analysis of TCP Variants

Archana Sampath
Sampath.ar@husky.neu.edu
College of Computer and Information Science
Northeastern University

ABSTRACT

The main aim of this project is to differentiate between the different TCP variants by comparing their throughput, latency and drop-rate under different sets of conditions. The variants the we are going to be using are namely:

1. TAHOE
2. RENO
3. NEW RENO
4. VEGAS
5. SACK

The performances of these variants are our points of comparison where we compare them based on congestion control patterns, fairness and their behavior under different queuing algorithms. The experiments have been performed by varying the parameters like CBR bandwidth, time and the queuing algorithms.

INTRODUCTION:

TCP (Transmission Control Protocol) is one of the most important protocols that is being used in the IP networks. The importance of TCP is great since it's the only protocol that allows the sender and receiver to communicate with each other by successfully connecting to each other and sending streams of data to and fro. The main reason why TCP was developed was to enable a successful interconnection between two hosts for reliable message sending techniques. Over the years the TCP has been modified in order to cater to the growing number of people connecting to each other using TCP and not forgoing the main reason why it was developed in the first place.

In this project we will be closely studying the behavior of the different variants under different conditions, namely their behavior when there is traffic on the network, their behavior when the bandwidth of their link is varied between 1 and 10, their behavior when the timing of the TCP flow is varied, their behavior when the queuing algorithms are changed. Another major test that is going to be done is the computation of the fairness between the variants. In the real world scenario we can have multiple TCP variants that will be sharing the same links with a limited bandwidth allotted to them. It is up to us to check and see of each of the variants is being fair to the other variant that it's sharing its link and bandwidth with. These experiments are done to compare the performances of the different TCP variants and that will be done by deriving graphs that compare the following parameters:

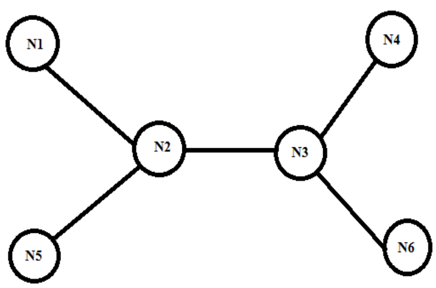
1. Throughput: Is the rate at which a packet successfully reaches the receiver.
2. Latency: Is the time taken by a packet to reach the destination by the total number of packets received by the destination.
3. Drop rate: The number of packets that have been dropped while it moves from the sender to the receiver by the time taken by the packet.

Graphs are plotted for the outputs for each of the variants and then a comparison in made to come to a conclusion about the best TCP among all of the variants. The TCP with the better attributes among all is selected to be the better TCP among all the other variants.

METHODOLOGY:

For simulating the entire network in a near real-world simulation, we would be making use of the NS-2 Network simulator. The simulator lets us simulate all the variants and their traffics, along with the start times and end times of the different flows namely TCP and CBR (Constant Bit Rate). The queuing algorithms and the queue size can also be changed. The bandwidth of the different links can also be changed. Since we are going to use these variations in our experiment we are going to make use of the NS-2 Network Simulator.

The network that we would be simulating for all our experiments is as shown below.



All these links are a full-duplex links with a bandwidth of 10 mbps.

For all our experiments we will be using the same network but with a few tweaks made here and there. The main reason why we are using this network simulation is because the variations that we need to make to our network is easily possible to make in this network simulator.

To analyze the performance of the variants under congestion we are going to use this network and implement and TCP flow which starts from N! and which has a sink at N4 moving through N2 and N3. Along with the TCP flow we also have a CBR flow whose source is at N2 and its sink is at N3. We start the CBR flow first and then we start the TCP flow and with varying bandwidths from 1 to 10 Mbps we calculate run the .tcl script .On running the .tcl script we get a .tr file. The .tr file is nothing but a trace file which contains the packet information of all the packets that are being sent and that have been received by the destination node. Once we get that we use a script in any other programing language to extract the information and to calculate the throughput, latency and the drop rate of the different variants. This experiment is repeated with varying bandwidth for all the variants.

For the second experiment we use the same network topology the only difference being that instead of having only one TCP flow of one variant, we will now have two TCP flows of two variants. One TCP flow is from the node N2 to N4 and the other is from node N5 to N6, there is also a CBR flow between the nodes N2 and N3. The main motive of this experiment is to check if there is fairness between the TCP variants since there are two present in the network. The same as before we use the CBR bandwidth change to get the variations in the .tr files. The .tr file entries are then parsed and the throughput, latency and the drop-rate are then compared to get the results.

For the third experiment we need to use the same network topology as the first experiment with the exception in the CBR flow which is not between N5 and N6. The queuing algorithms are changed between Drop tail and RED and this time the time is varied between the variants RENO and SACK. At first we start the TCP flow and check where it becomes stable. After that we start the CBR flow and check for the performance of the variant. WE calculate the throughput and the latency for the different variants and plot them in a graph to see which queuing algorithm is the most effective.

These are the methodologies that we are going to be implementing in order to get the performance evaluation of the different variants.

EXPERIMENT 1:

The main motive of this experiment was to compute or evaluate the performance of the TCP variants, namely TCP TAHOE, RENO, NEWRENO, VEGAS. As mentioned before we will be using the network topology as given earlier. There is a TCP flow from N1 to N4 and there is a CBR flow from N2 to N3.

The CBR flow is started first and then the TCP flow is started. The variations in the performance are made by changing the bandwidth of the link that carries the CBR traffic namely the link N2 and N3. So on varying the bandwidth we can get to know how the different variants will behave during the congestion period. We need to come up with the variant that has the best throughput, the least latency and the least drop-rate.

The graph showing the comparisons of the throughput of the variants is shown below

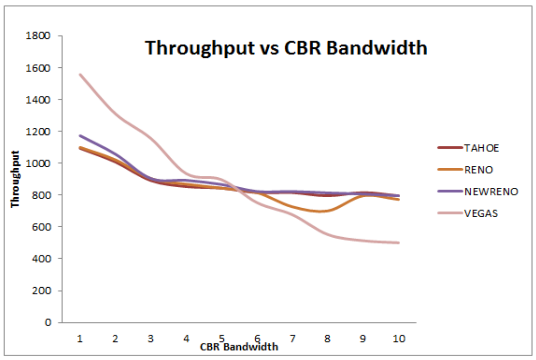


Fig 1.1: Throughput vs CBR Bandwidth

According to the graph above we can see that VEGAS has the best average throughput since it starts at a much higher throughput as compared to the other variants and towards the end due to the congestion window limitation the throughput decreases. But as an average throughput the VEGAS variant shows the better performance.

The graph for the latency of the variants is shown below

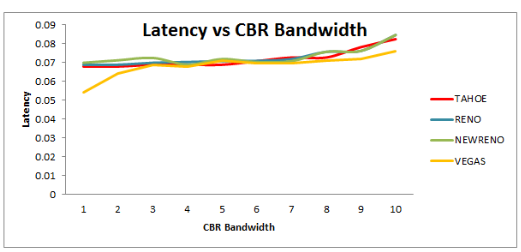


Fig 1.2: Latency vs CBR Bandwidth

According to the graph shown the least latency is shown by TCP VEGAS as compared to the other variants. We can observe that the latency starts low but then becomes almost as equal to the other variants but then towards the end we can see that the latency decreases.

The graph showing the comparisons of the drop-rate of the variants is shown below

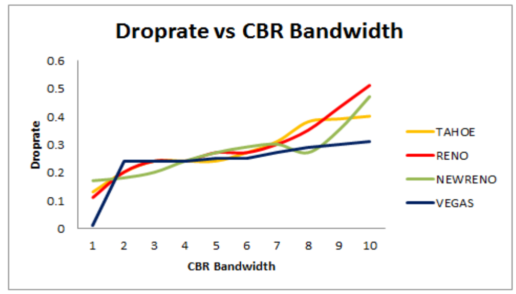


Fig 1.3: Droprate vs CBR Bandwidth

According to the graph above we can see that VEGAS show the least drop-rate as compared to the other variants.

According to the experiments that I have made. I have varied the environment to a great extent to see if the variants are behaving the same way in an uncontrolled environment as they are behaving in a controlled environment. I have varied the queue sixe, the bandwidth of the TCP links, played around with the time, where I start the CBR flow before the TCP flow to see if the behavior of the variants is the same. I have come to a conclusion that apart from very minor changes in the behavior of the variants under the uncontrolled environment they all exhibited an almost same behavior. So I can conclude this experiment by saying that VEGAS is a better variant as compared to the other variants since its throughout is better that the other variants. The latency is comparatively low and the drop-rate is also comparatively low so I can confidently say that VEGAS is a better variant when it comes to handling congestion in the network.

EXPERIMENT 2:

The main objective of this experiment is to see how the different TCP variants behave when they are pitted against each other. When we consider a real world scenario we need to account for the presence of various combinations of variants that are present in the network. Keeping this in mind we are doing an experiment to see the fairness of the variants when another variant is present in the system.

The experiment is performed using the same network configuration, the only difference being that instead of one TCP flow we will have two TCP flows of the two variants, one from N1 to N4 and the other from N5 to N6 and along with that we will also be having the CBR flow from N2 to N3.

In order to come up with the fairness result we will be again varying the bandwidth of the CBR node and calculate the throughput, latency and the drop rate of the variants.

The graph showing the throughput of the first variant combination is shown below

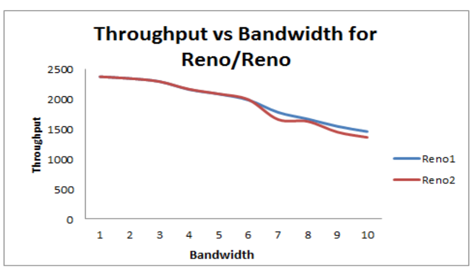


Fig 2.1: Throughput vs Bandwidth for Reno/Reno

According to the graph we can see that there is a very good amount of fairness between the two variants. So we can say that Reno/Reno acts fairly in the presence of each other.

The graph showing the latency of the combination is shown below

From the graph we can see that the latency for both the variants is also not so different from each other. So we can say that they are fair to each other

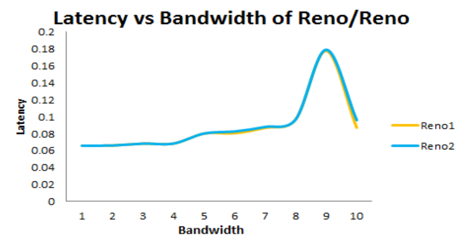


Fig 2.2: Latency vs Bandwidth of Reno/Reno

The graph showing the throughput for the second set of variants i.e. New Reno/Reno is shown below.

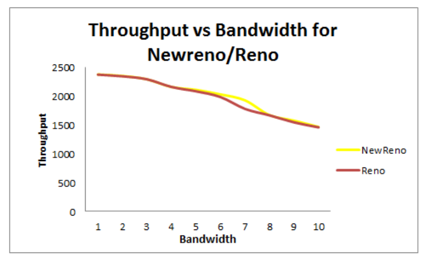


Fig 2.3: Throughput vs Bandwidth for Newreno/Reno

From the graph we can see that there is again a very good fairness between the two variations. At one point New Reno gets a better throughput but then it becomes equal to Reno again.

The latency graph for the two variants are shown below.

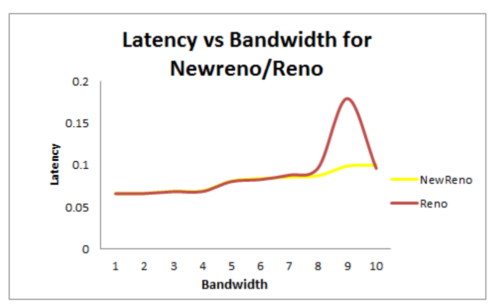


Fig 2.4: Latency vs Bandwidth for Newreno/Reno

From the figure we can see that the latency for new Reno is greater that the latency for new Reno since the fast re-emit feature of Reno will make it resend the packets after 3 acks are received as a result of which its latency increases.

The graph for the throughput for the 3 variant combination i.e. Vegas and Vegas is shown below.

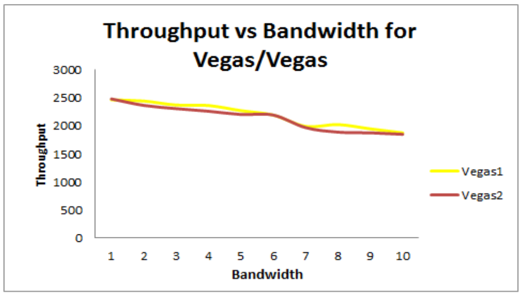


Fig 2.5: Throughput vs Bandwidth for Vegas/Vegas

According to the graph we can say that there is a fair amount of fairness between the two variants.

The graph for the latency is as shown below

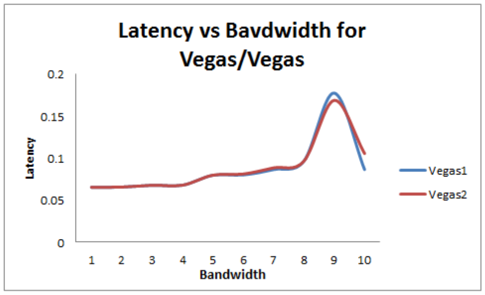


Fig 2.6: Latency vs Bandwidth for Vegas/Vegas

According to the graph we can see that there is fair amount of fairness between the two variants when it comes to Vegas and Vegas as well.

The graph for the last variant pair i.e. New Reno/Vegas is as shown below.

According to the graph we can see that there is a huge difference between the throughputs for the two variants. We can see that throughput for New Reno is better that the throughput for Vegas when both are on the same network. The reason being the Vegas has a slow start mechanism and by that time the New Reno would have already occupied the bandwidth hence the unfairness

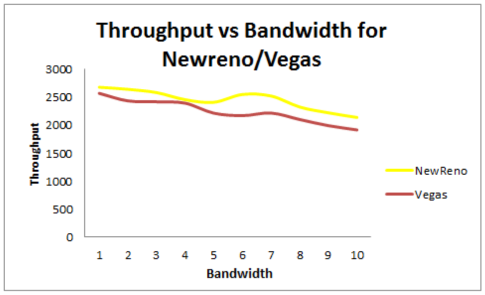


Fig 2.7: Throughput vs Bandwidth for Newreno/Vegas

The latency for the variants is as shown below

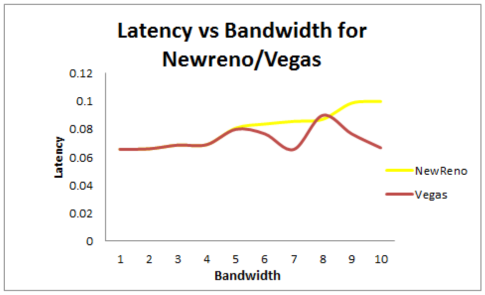


Fig 2.8: Latency vs Bandwidth for NewReno/ Vegas

From the graph we can see that the latency for New Reno is higher than that of Vegas. The reason being the higher re-transmit rate for new Reno causes the latency to be higher than that of Vegas.

This concludes the second experiment.

EXPERIMENT 3:

The main aim of the third experiment is to check for the influence of queuing on the different variants. For this experiment we will be using the Reno and the Sack variants of TCP. The network topology is the same except that the CBR flow is not from N5 to N6.

In order to successfully complete this experiment we need to vary the queuing algorithm that is being used, namely RED and Drop tail. We start the TCP flow first and after it stabilizes we start the CBR flow and check the performance, namely Throughput and Latency.

The graphs are shown below:

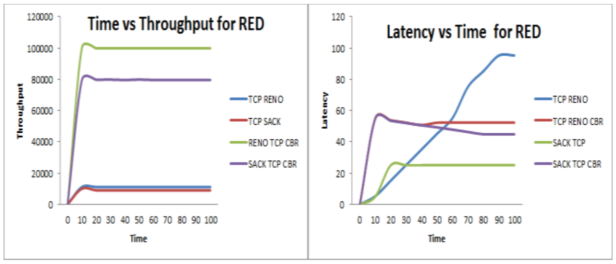


Fig 3.1: The Throughput and Latency for RED

From the graph we can see that the throughput and latency using the RED queuing algorithm is better after the CBR flow is started.

The graph for Drop tail is shown below

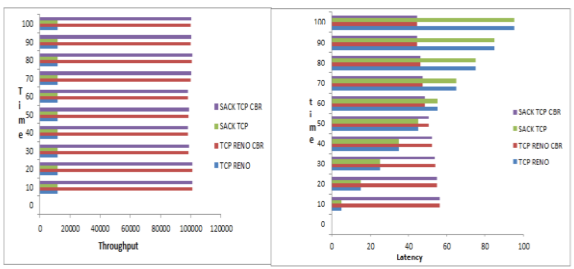


Fig 3.2: Throughput and Latency for DropTail

From the graph we can say that the variants perform better when the CBR flow is started as opposed to when there is no CBR flow.

I can conclude the experiment 3 by saying that the RED queuing algorithm provides more fairness to the variants. Since RED algorithm doesn't blindly drop the packets that arrive at it. It does a random early detection as a result of which the latency for the variants that implement the RED algorithm are is quite less as compared to the variants that implement the Drop-tail algorithm. The throughput is also high for the variants that implement the RED algorithm. When the CBR flow is created it is noticed that the TCP latency decreases and the throughput increases. According to me since SACK is nothing but selective ack it is alright to implement it with RED since if there is any collision detection then RED will take care of it.

This concludes the 3rd experiment.

CONCLUSION:

At the end of all the experiments and the tests I have come to a conclusion for each of the experiments performed.

In the first experiment I can clearly say that under congestion the TCP VEGAS variant performed better than the other variants since it's throughout was much better than that of the other variants. The drop-rate and the latency was low as compared to that of the other variants.

In the second experiment we could clearly see fairness between the variant Reno/Reno and Vegas/Vegas.

In the third experiment we could clearly see that RED algorithm for queuing is much preferred since the throughput of the variants that implemented it were higher than that of the other variants. The drop-rate and the latency were low as well.

References:

- Kevin Fall and Sally Floyd , "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP"
- http://www.isi.edu/nsnam/ns/tutorial/index.html
- http://nile.wpi.edu/NS/
- http://www.isi.edu/nsnam/ns/ns-documentation.html