Performance Analysis of TCP Variants

Chenxi Yuan
College of Computer and Information Science
Northeastern University, MA
yuan.ch@husky.neu.edu
NUID 001738048

Leyi Qiang
College of Computer and Information Science
Northeastern University, MA
qiang.l@husky.neu.edu
NUID 001907268

***Abstract*—In this paper, we will analyze the difference of different TCP variants (Taheo, Reno, NewReno, and Vegas) in large and congested network. In the first experiment, we analyze the performance of TCP variants under the influence of various load conditions. In the second experiment, we will analyze the fairness between different TCP variants. In the third experiment, we will analyze the influence of the queuing discipline used by nodes on the overall throughput of flows.**

Keywords—TCP, TCP variants, Queuing disciplines, Taheo, Reno, NewReno, Vegas.

I. Introduction

TCP is one of the most important protocols of the Internet protocol Suite. The original TCP, proposed in 1974, provided a reliable protocol for end-to-end communication. Although the original TCP works reliably, it does not provide an acceptable performance in large and congested networks. Therefore, people attempted to modify the original TCP to provide a better mechanism to avoid congestion. In this paper, we will analyze several TCP variances (Taheo, Reno, NewReno, and Vegas). We will analyze how each TCP performs under congestion, how do they affect each other under the same network, and how queuing disciplines, like DropTail and Random Early Drop (RED), treat the packets in a queue.

II. Methodology

A. The network topology

We will use a single network topology for all three experiment.

B. Tools

- NS2: it is a discrete event simulator targeted at network research. We use NS2 to simulate the TCP behavior in a pseudo network.
- Perl: it is a script programming language. WE use Perl to run NS2 configuration file and to parse and organize data.
- Gnuplot: it is a command-line graphing tool. We used Gnuplot to generate graphs.

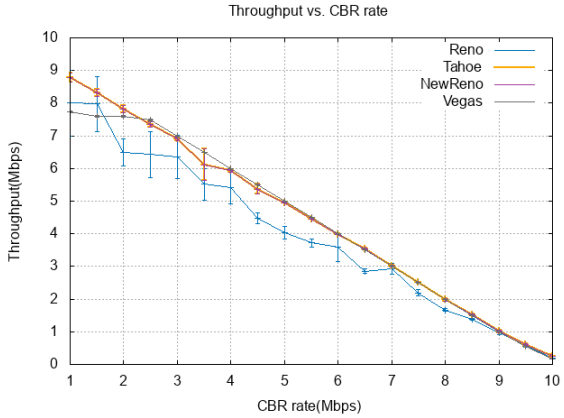
III. Experiments

A. *Experiment 1: TCP Performance Under Congestion*

In the first experiment, we analysis the performance of Tahoe, Reno, New Reno and Vegas under different CBR load. The topology contains six nodes (N1 – N6). Band width between each node is set to 10 Mbps. The delay between each node is set to 10ms. A TCP stream and FTP application are attached to N1 to a sink agent N4. CBR source is attached to N2 to sink N3. The performance between each TCP variant is valued in three aspects: Throughput, Latency and Packets Drop Rate. The result shows Reno's throughput is lower than others and Vegas has better performance on latency and packets drop rate.

Throughput

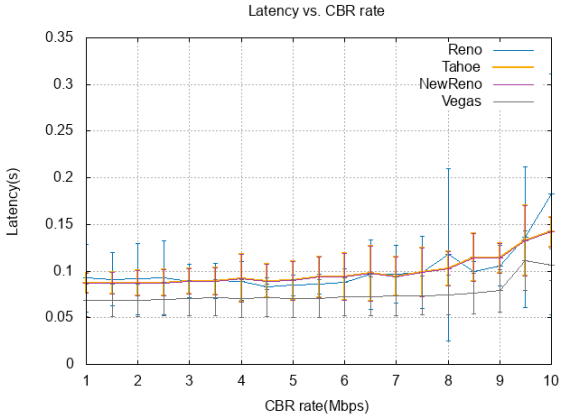
Figure "Throughput vs. CBR rate" shows the throughput performance in increased CBR rate under different variants. The figure shows the overall throughput are similar between Tahoe, New Reno and Vegas, and Reno is lower than the others.



At the beginning, when CBR rate is small, Vegas has lower throughput than others, because during slow start, Vegas has lower rate of increasing window size due to its different slow start mechanism. When CBR rate is between 3-8 Mbps, each variant start getting into congestion avoidance stage. Both Reno and NewReno are start to using fast recovery and fast retransmit. While Reno cannot handle multiple packets lost, its window size drops frequently, that's why we can see Reno's throughputs have multiple spikes on the graph and stays lower than others during the mode. Since Tahoe's treats multiple duplicate ACKs as timeout and will reduce the window size to 1, the performance during this stage is lower than New Reno and Vegas, and we can see a spike during that period.

Overall it is hard to tell which one has the best performance over throughput, Vegas has slightly better performance after slow start. However, NewReno and Tahoe are getting close to Vegas after congestion becoming larger.

Latency

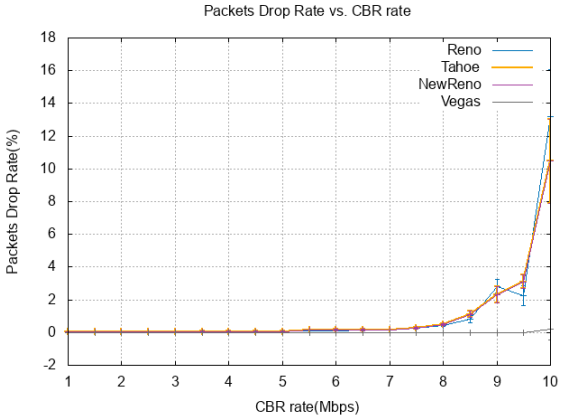


The latency is calculated based on Roundtrip Time; figure "Latency vs. CBR rate" shows the latency of each variant under different CBR rate. As we can see Vegas has the lowest average latency, because the way Vegas detect timeouts is based on round-trip time, and Vegas is calculating RTT to avoid latency and packets drop; However, since the standard deviation is very large during this experiment, we cannot say Vegas' latency performance is absolutely better than others.

Packets Drop Rate

We count total packets received and dropped from the TCP stream to get packets drop rate. The figure below shows the performance of each TCP variant. From the graph we can see there's no big difference between Reno, Tahoe and NewReno, and it is clear that the drop rate of Vegas is very low and almost 0 even under high congestion situation.

The reason Vegas has lowest packets drop rate is because it is using RTT as a way for detecting congestion, while others are detecting congestion after packets get dropped. When CBR rate increase, Vegas will detect the increase of RTT, and start reducing window size. That's why Vegas can keep 0 drop rate easily.



B. Experiment 2: Fairness Between TCP Variant

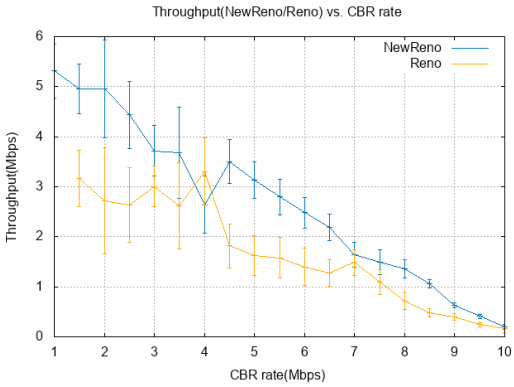
This experiment checks fairness between different TCP variants. The topology is based on experiment 1, in addition we add another TCO variant on N5 to a sink N6.

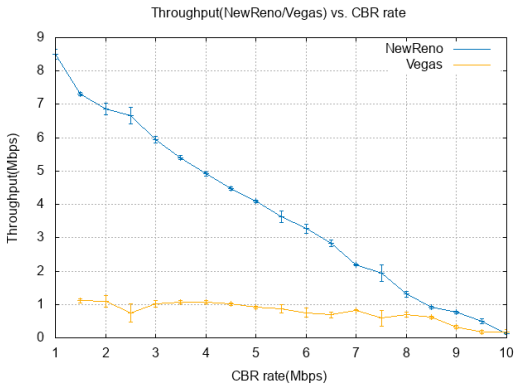
The result shows many TCP variants are fair to each other based on latency and packets drop rate; yet there is still some obvious unfairness when we do the throughput performance test.

Throughput

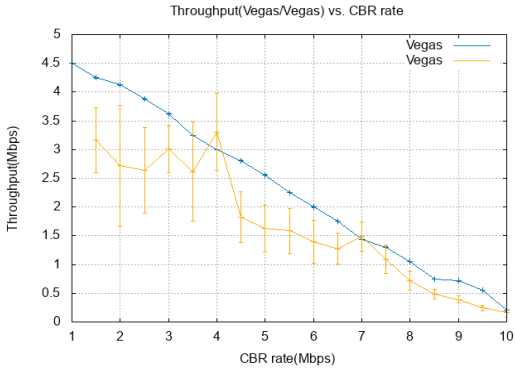
Figure below shows there's fairness in the condition of Reno vs. Reno. NewReno gives a better throughput performance over Reno and Vegas.

In the New Reno vs. Reno and New Reno vs. Vegas situation, during fast retransmit mode, New Reno retransmit one lost packet per RTT until all lost packet get retransmitted. It will keep retransmission window full and at high throughput. For Reno and Vegas, it retransmit packets only after every 3 duplicate ACKs.

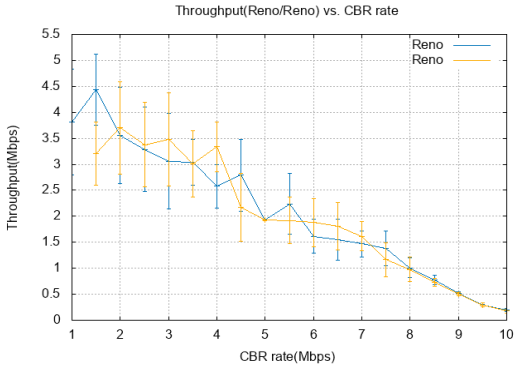




Interestingly, Vegas doesn't show fairness during the test against another Vegas variant, the result shows Vegas performs badly while another TCP variant is in presence. The reason is due to one Vegas takes the main bandwidth of TCP stream and the others will not get much bandwidth after it.



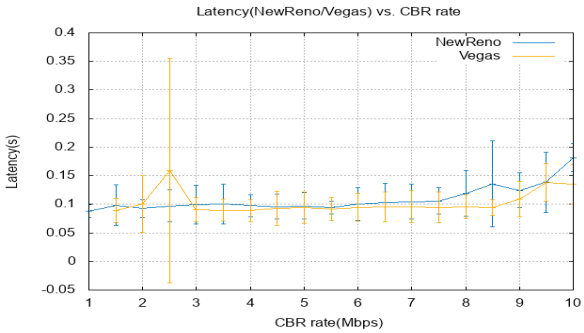
Reno and Reno shows fairness throughput between each other from graph shows below.

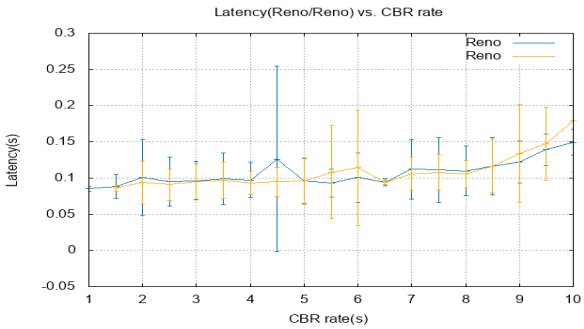


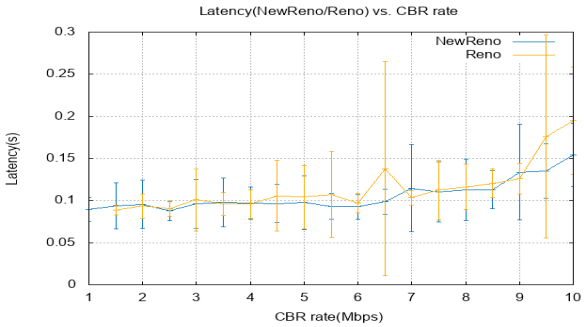
Latency

During latency performance test, Vegas shows less delay than Reno due to the way Vegas avoid congestion. Overall, the result shows fairness between each TCP variants.

When NewReno compare against Vegas, Vegas shows less latency during the test, it is due to Vegas uses RTT as a standard for congestion avoidance. Thus Vegas has better performance during then latency test.

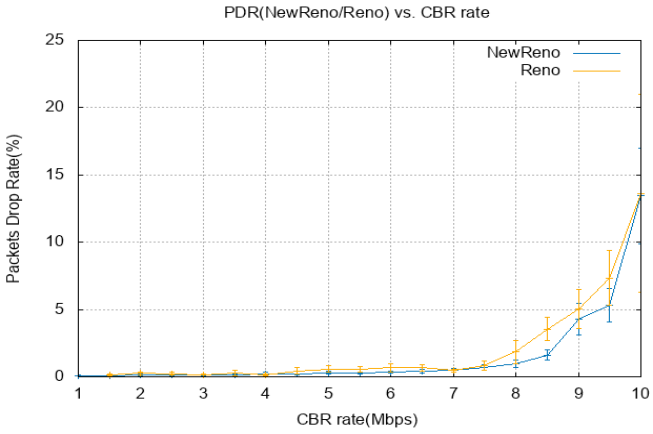


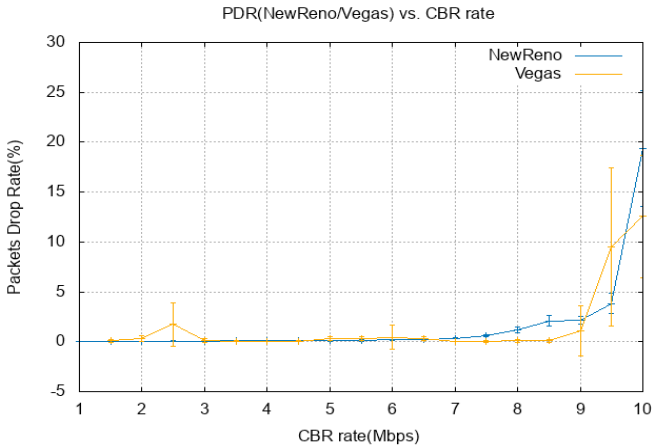


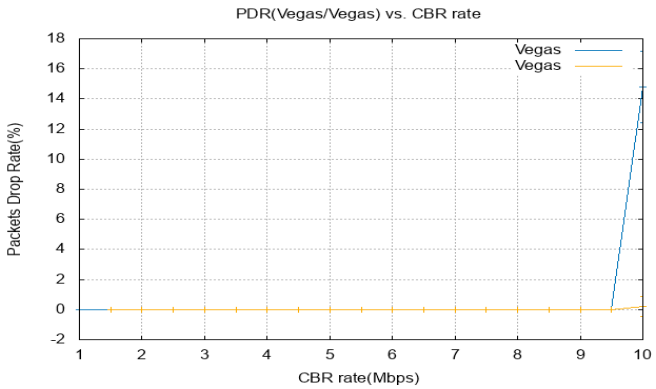


Packets Drop Rate

Packets Drop Rate between each TCP variants are also show fairness except NewReno vs. Reno, from the graph below we can see the drop rate of Reno is slightly higher than New Reno. This is because Reno will retransmit after 3 duplicated ACKs, yet at that time it may already have multiple packets dropped. Due to NewReno's fast retransmit mechanism, it avoids the problem.







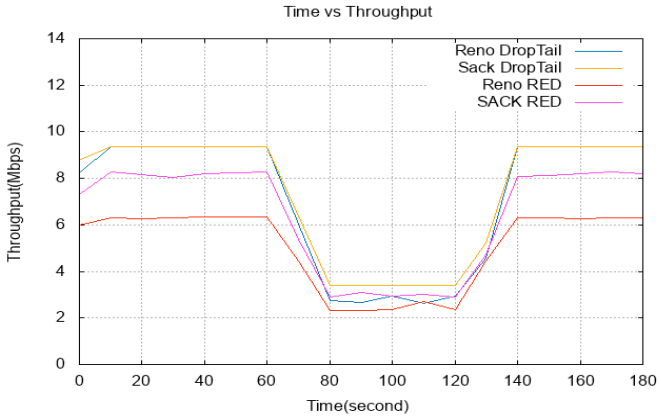
C. *Experiment 3: Influence of Queuing*

In this experiment, we will analyze how different queuing algorithms influence different TCP variants. We tested DropTail and Random Early Drop(RED) with TCP variants: TCP Reno and SACK.

Configuration: We start the TCP flow under the queuing algorithm first and wait for TCP to be steady. We start CBR flow 75 seconds after TCP starts, and we end CBR flow 65 seconds before we end TCP flow.

Calculation: We take the duration of 10 seconds, and calculate the throughput and latency for the 10-second duration.

Throughput:



Before CBR is started, DropTail provides a better throughput than RED. DropTail also provides the same throughput to Reno and SACK. RED does not provide the same throughput to Reno and SACK. From graph, we can tell that RED with Reno has the worse throughput when the network is not congested.

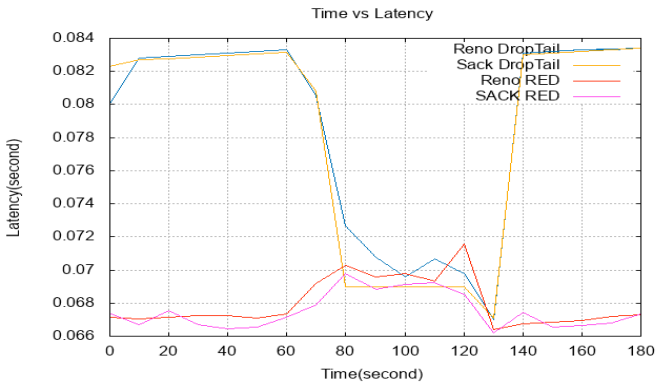
When CBR is started, all of them almost drop to a steady throughput at the same time. Their adaptability of the frequent changes of network congestion are similar.

When CBR stays constant, RED with Reno still has the worst throughput. Both DropTail and Red work better with SACK when the network is congested. .

When CBR ends, both RED and DropTail go back to the state before CBR starts

For the throughput, we can draw the conclusion that DropTail provides a better throughput than RED.

Latency:



Before CBR is started, RED possesses a better latency than DropTail. Both RED and DropTail provide an equal latency to the different type of TCP variants.

When CBR is started, DropTail's latency drops quite a lot for both TCP variants. RED's latency increases a little bit for both TCP variants.

When CBR stays constant, both DropTail and RED perform similarly as their latencies are nearly the same.

When CBR ends, Both RED and DropTail go back to the state before CBR starts.

For experiment 3, we concluded that DropTail provides a better throughput than RED all the time, and RED provides a better latency rate than DropTail when the network is congested.

When CBR flow is created, TCP flow drops quite a lot for both queuing algorithms and both TCP variants, though some have better performance than the other.

When latency is considered more important than throughput, RED is a good idea while dealing with SACK, as RED provides a better latency when the network is not congested. However, when throughput is more important, RED is not a good idea while dealing with SACK.

IV. Conclusion

We analyze the performance of different TCP variants in three experiments above. The conclusion for each experiment:

For experiment 1, we concluded that Vegas TCP have a better drop rate prevention when the network is congested, though Vegas TCP has the worst throughput in the four TCP variants (Taheo, Reno, NewReno, and Vegas) we tested.

For experiment 2, we concluded that many TCP variants are fair to each other based on latency and packets drop rate; yet there is still some obvious unfairness when different TCP variants exists in the same network.

For experiment 3, we concluded that DropTail has a better throughput than RED, but DropTail latency is worse than RED. In addition, SACK is better option than Reno when RED is chosen as the queuing algorithm.

ACKNOWLEDGMENT

We would like to express our gratitude to Professor David Choffnes for his help to us to understand TCP variants and how to conduct network experiments.

REFERENCES

[1] Cerf, V., and R. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. COM-22, No. 5, pp 637-648, May 1974.

[2] V. Jacobson. Congestion Avoidance and Control. In SIGCOMM, 1988.

[3] Fall, K. and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno and SACK TCP", Computer Communication Review, July 1996, ftp://ftp.ee.lbl.gov/papers/sacks.ps.Z.