Random Early Detection (RED)

Goal of the Algorithm

- To detect incipient congestion that has persisted for a long time
- To avoid a bias against bursty traffic
- To achieve global synchronization that results from notifying all connections to reduce their windows at the same time.

Algorithm

The main idea is that we take care of the slowly increasing queue size by early dropping the packets. There are 2 queue size levels, min and max.

When, the Avg queue size increases beyond the min level, we drop a packet with a certain probability, which is proportional to how much is the Avg queue size above the min threshold.

Also, to ensure that we do not wait for too long before dropping a packet, we increase the count since the last dropped packet.

Average Queue size is calculated using a low pass filter with an exponential weighted moving average. The recursion formula for the same is mentioned below.

$$p_b \leftarrow max_p(avg - min_{th})/(max_{th} - min_{th}).$$

$$p_a \leftarrow p_b/(1 - count \cdot p_b)$$

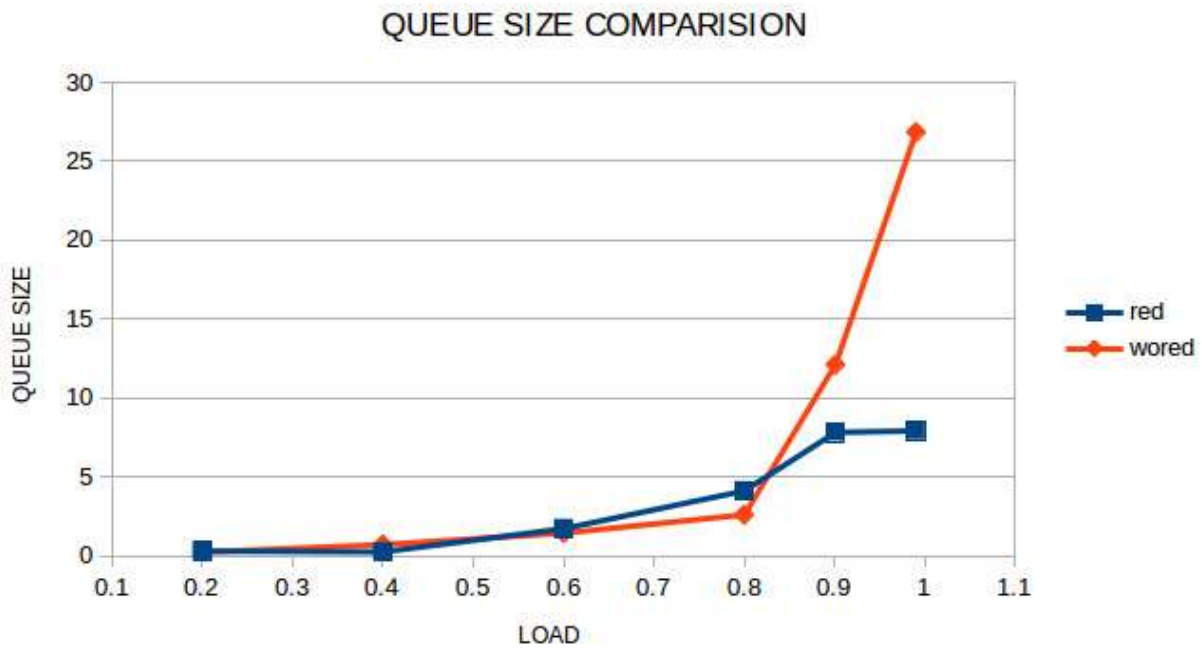
$$avg \leftarrow (1 - w_q)avg + w_q\, q$$

Simulation Parameters

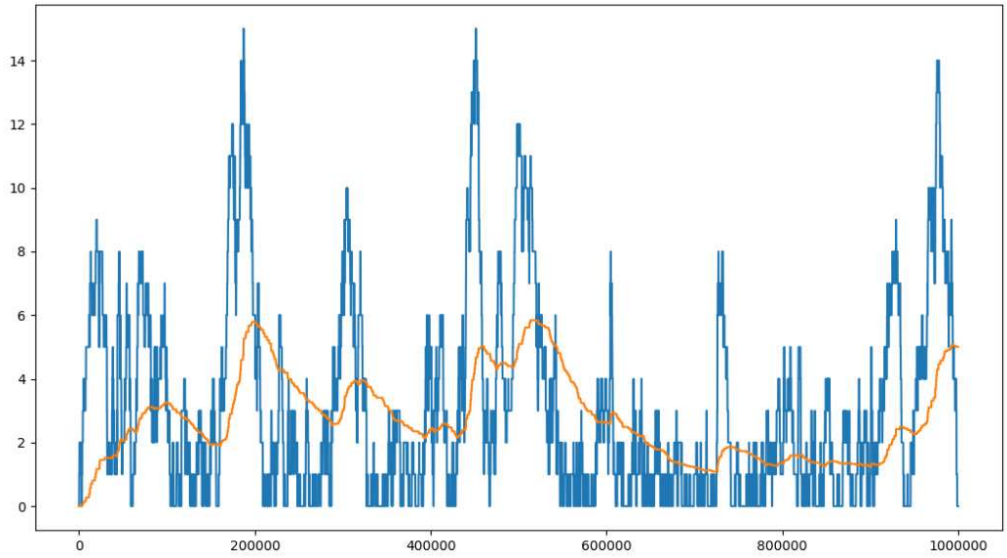
$W_q = 0.002$, $max_p = 1/50$

$min_{th} = 5$ packets, $max_{th} = 15$ packets, (Assumption – Uniform packet size)

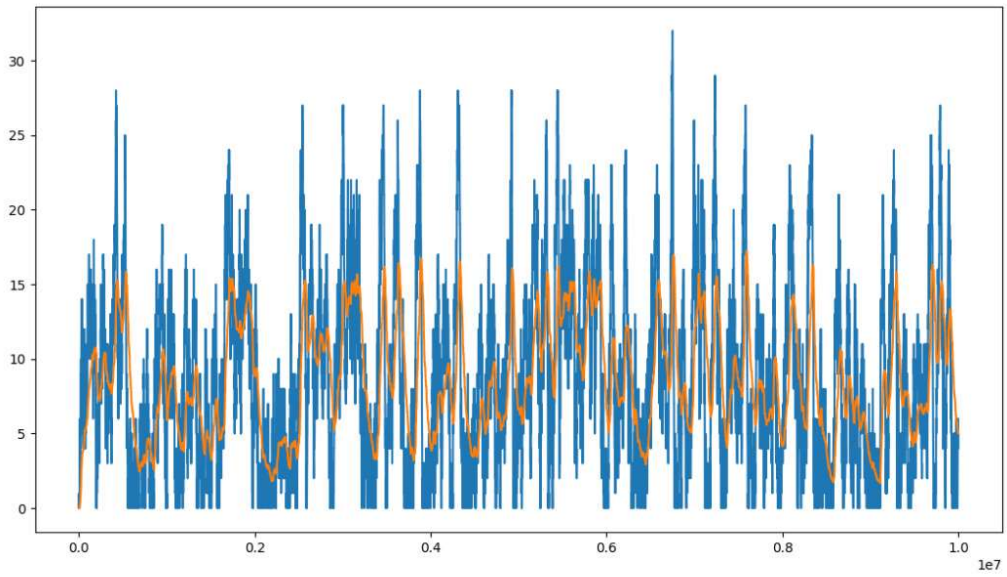
Observation



Comparison: **RED** and **w/o RED**

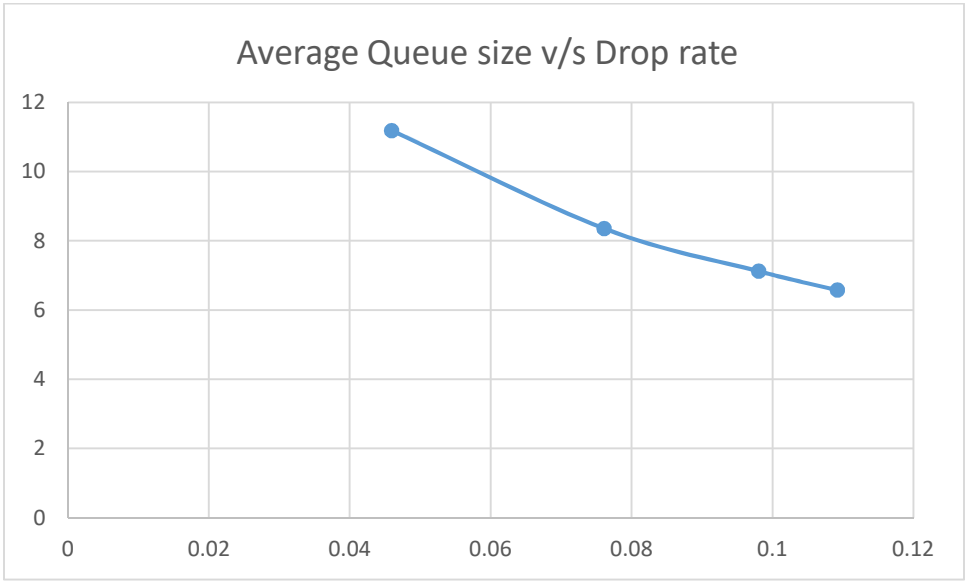


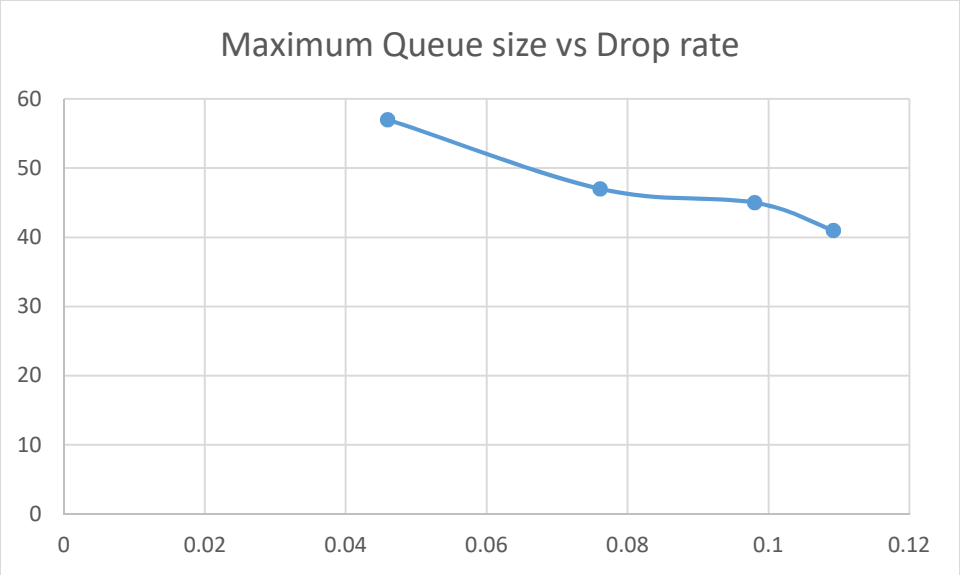
Queue Size v/s **Time**



Queue Size v/s **Time (Increased simulation time)**

- As we can clearly see from the above graph that for higher loads, RED drastically improves the Queue Size.
- This also improves the throughput of the switch for RED.
- However, at lower loads, there is not much performance difference, this is because at low loads queue sizes are normally very small.





- The above plots are at a load of 0.99.
- As expected, the Average Queue size is decreasing with increase in Drop rate.
- Also, the Buffer size we need should be equal to the Maximum Queue size over a large time. We ran the simulation for $3*10^7$ iterations. So the maximum queue size is a good first approximation. Thus we can confidently say the buffer size we need is 60*(average packet size).

Conclusion

We can clearly see that the RED algorithm improves the average waiting time in the queue as well as increases the throughput of the overall system while dropping a small fraction of packets. By dropping all the marked packets, this action provides an upper bound on the average delay at the gateway. Also, the probability that RED chooses a particular connection to notify it to reduce its windows size is roughly proportional to that connection's share of the bandwidth at gateway.