Analyzing the Efficiency and Performance of SW in High-Speed Data Networks

Introduction

Background:

In high speed data networks the flow control is important along with congestion control and error corrections. To ensure reliability many flow control mechanism are used we can see that each one is a modification of other. Now we are going to see the easy protocol also the main building of other mechanism that is top and wait protocols.

Stop-and-wait ARQ:

Introduction:

It is the simplest protocol and easy to implement. In stop and wait protocol, Sender sends one data packet and then waits for its acknowledgement.
Sender sends the next packet only after it receives the acknowledgement for the previous packet.

Lets see some of the problems and how it is overcoming them,

Problems:

1.Problem of Lost Data Packet:

Solution:

Time out timer helps to solve the problem of lost data packet.
After sending a data packet to the receiver, sender starts the time out timer. If the data packet gets acknowledged before the timer expires, sender stops the time out timer.
If the timer goes off before receiving the acknowledgement, sender retransmits the same data packet. After retransmission, sender resets the timer.

2.Problem of Lost Acknowledgment:

Solution:

Sequence number on data packets help to solve the problem of delayed acknowledgement. Consider the acknowledgement sent by the receiver gets lost. Then, sender retransmits the same data packet after its timer goes off.

The sequence number on the data packet helps the receiver to identify the duplicate data packet. Receiver discards the duplicate packet and re-sends the same acknowledgement.

3.Problem of Corrupted packets:

Solution:

If receiver receives a corrupted data packet from the sender, it sends a negative acknowledgement (NAK) to the sender. NAK requests the sender to send the data packet again.

Mathematical Visualization:

The transmitter sends a single information frame of nf bits to the receiver and spends tf time units in doing so. It then stops and waits to receive an ACK from the receiver. If no ACK is received within a given time out, tout, period, the frame is resent by the transmitter and once again it stops and waits for the ACK. If an ACK is received within the given tout period then the transmitter checks the received frame for errors during the next tproc time units. If
errors are detected then the ACK is ignored and the frame is resent by the transmitter after tout expires and once again the transmitter stops and waits for the ACK . If no errors are detected in the ACK frame then the transmitter transmits the next frame in its queue. The receiver is always waiting to receive information frames. When a new frame arrives, the receiver checks it for errors during the next tproc time units. If errors are detected
then the information frame is ignored and the receiver continues to be in the wait state otherwise it initiates the transmission of an ACK frame of na bits, which takes ta time units. Under the above mentioned conditions, the ACK message cannot be received before tprop +tproc +ta +tprop +tproc units of time pass after sending out an information frame, where tprop represents the one-way propagation time between transmitter and receiver. It is, therefore, necessary to set the time out period tout to be greater than or equal to 2(tprop + tproc)+ta for reliable communication.

Efficiency Calculation:

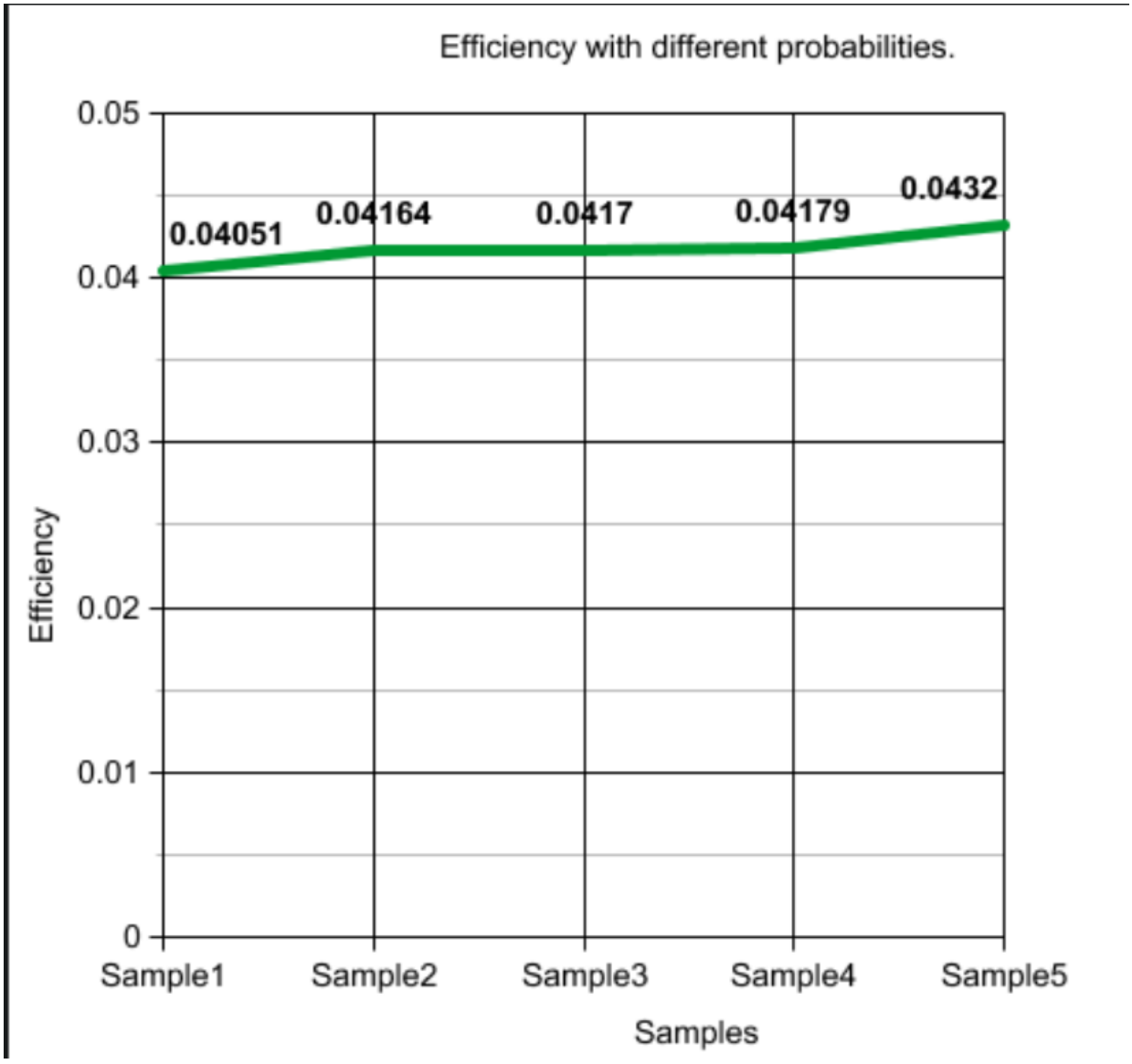
As we can neglect Tt for the ack packets and we also neglect procession delay and queuing delay.

Efficiency = (Tt for one packet)*(no of packets)/(Total time).

Efficiency analysis of the code written with different samples:

Case 1:

Keeping RTTs same and changing the probability of the crashing and corrupting and simulating it.

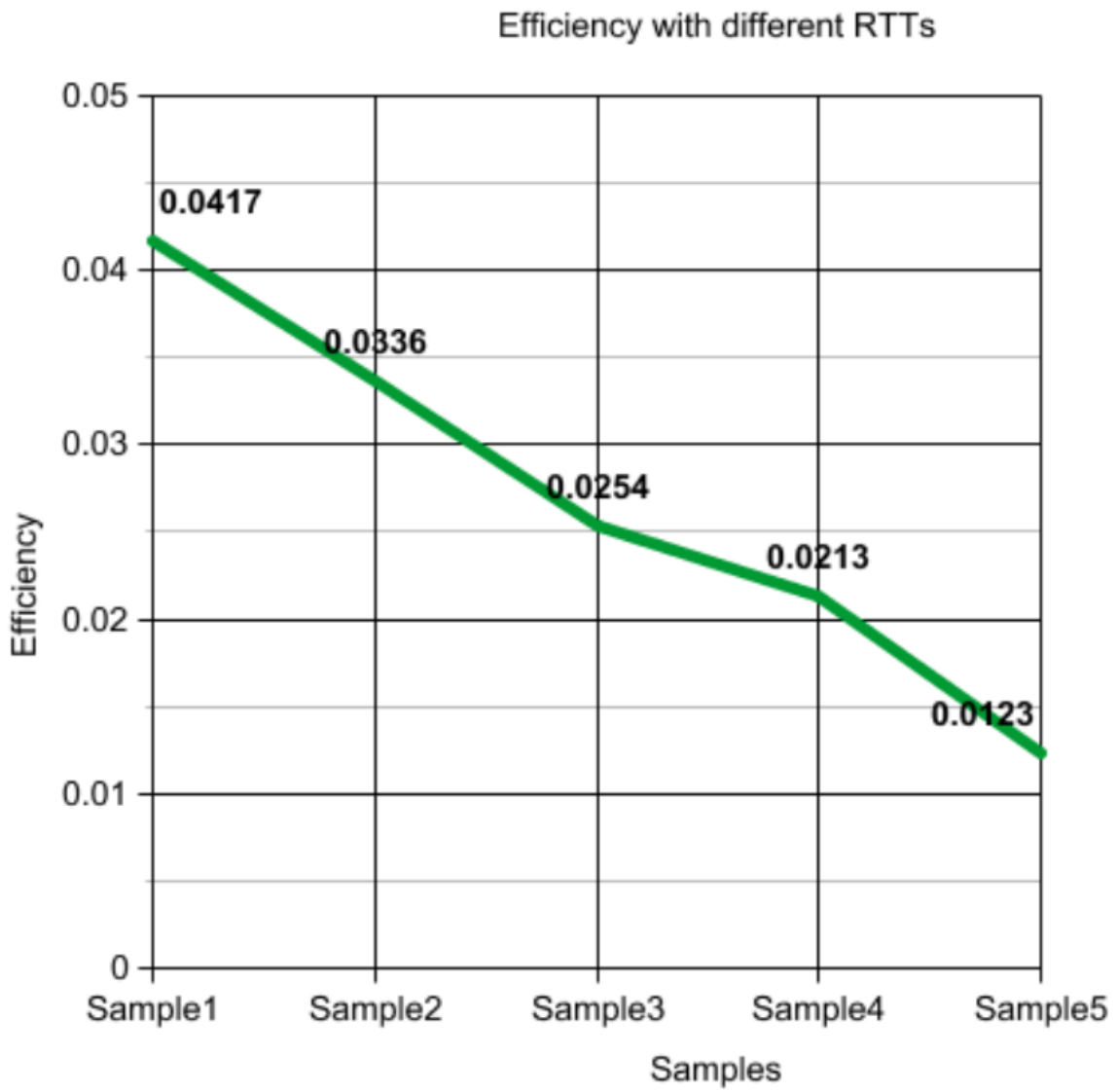


From Sample1 to Sample5, I am decreasing the probability of errors and simulating and from the formula If probability decreases time decrease hence efficiency increases as we can see this in the graph above.

Case 2:

Keeping Probabilities of crashing and corrupting same and changing the RTTs and simulating it.

Graph:



From Sample1 to Sample5 I am increasing the RTT Values of the packets so the time increase implies efficiency decrease.

Performance analyzing in high speed Data Networks:

Mainly till now we neglected the queuing effects in all cases but here comes the big problem when we are doing them in high speed data networks:
Queuing effects arise due to the presence of queues at various points in the network, such as routers, switches, or buffers, where packets temporarily wait before being processed or transmitted.

1) High-speed networks might experience queuing delays, especially during congestion or when there's a high volume of traffic. The stop-and-wait protocol, which waits for acknowledgments before sending the next packet, can lead to increased delays. The time spent waiting for acknowledgments to return from the receiver before transmitting the next packet can increase the overall transmission delay.

2) In high-speed networks, the rate at which data arrives at the receiver can exceed the rate at which it can be processed or acknowledged due to limitations in buffer sizes or processing capabilities. With stop-and-wait ARQ, if the receiver's buffer is not cleared quickly enough, it might lead to buffer overflow, causing packet loss and retransmissions, thus reducing efficiency.

So due to these problems queues in high-speed networks often implement different scheduling algorithms for managing packet transmission. Stop-and-wait ARQ might not inherently consider the fairness of transmission or prioritize certain types of traffic, potentially leading to inefficiencies when dealing with diverse types of data and also during periods of network congestion, queues can build up, leading to increased queuing delays. Stop-and-wait ARQ's inability to adjust its sending rate dynamically in response to congestion might increase the congestion problem, leading to suboptimal performance.

Disadvantages of Stop-and-wait:

As we discussed in the previous context some of them in high speed data networks. Let’s look into some other of them now.

As we can see that only one frame is sent at a time so that it decrease efficiency as it is utilizing the whole bandwidth for that frame itself so to increase the efficiency multiple frames should be in transition while waiting for the ACK.

So this above discussed mechanism is discussed in the improvements of SW that is Go-Back-N and Sliding window mechanism.