ITS323 Introduction to Data Communications Assignment

Students: Marut Puechpaibool, Jakchai Tripipatkul, Ukrit Wattanavaekin

Group: G22

1 Experiment Setup

Host CPU	2.3 GHz Intel Core i5
Host RAM	4 GB 1333 MHz DDR3
Host OS	Mac OS X 10.9.2
Virtual Machine	VirtualBox 4.3.18
Virtual Network	virtnet r41

Table 1: Experiment Setup

2 Parameter Values

2.1 Fixed Parameters

Parameter	Value	Unit
Downlink data rate	100	Mb/s
Downlink delay	0	ms

Table 2: Fixed Parameters

2.2 Variable Parameters

Parameter	Range	Unit	Count
Uplink data rate	1 .. 10	Mb/s	10
Data size	10000 11000 15000 20000 30000 50000 60000 75000 90000 100000	Bytes	10
Uplink delay	10 15 20 30 50	ms	10

Parameter	**Range**	**Unit**	**Count**
	100 200 300 400 500		
Number of packets	100 200 300 500 1000 2000	-	6
Max window size	1 3 7 15 31 63 127 255 1024	frames	9
Frame error rate	0, 0.01 0.02 0.03 0.04 0.05 0.1 0.15 0.2 0.3	-	10
Timeout	0.2 0.3 0.5 1 2 5 10	s	7

Table 3: Variable Parameters

3 Performance Metrics

The performance metric(s) are:

1. Throughput, in bits per second, as measured at the server

For the experiments using the Stop-and-Wait error control, we used the value of the throughput when the number of errors corresponds to the error rate. For example, if we sent 100 packets and the error rate was 0.02, then we use the throughput when there are 100*0.02=2 errors.

4 Mathematical Model

The variables use in the mathematical model used to calculate expected efficiency are:

Uplink data rate, dr_u Mb/s	Frame size, f Bytes
Uplink delay, d_u ms	Header size, h = 4700 Bytes
Downlink data rate, dr_d Mb/s	ACK frame size, ack Bytes
Downlink delay, d_d ms	Uplink transmission delay, d_{trans} s
Total uplink time, t_u s	Maximum window size, W
Total downlink time, t_d s	Timeout, $t_{timeout}$ s
Round trip time, RTT s	Error-rate, $error_r$

Total uplink time, t_u s, is calculated as:

$$t_u = \frac{(f+h)*8}{dr_u} + \frac{d_u}{1000}$$

Total downlink time, t_d s, is calculated as:

$$t_d = \frac{ack*8}{dr_d} + \frac{d_d}{1000}$$

Note: In this experiment, t_d is assumed to be 0 seconds because it is significantly small compared to the total uplink time t_u.

Round trip time, RTT s, is calculated as:

$$RTT = t_u + t_d$$

Uplink transmission delay, d_{trans} s, is calculated as:

$$d_{trans} = \frac{(f + h) * 8}{dr_u}$$

1. **Stop-and-wait flow control (no error)**

   - Expected throughput, ρ Mb/s, is calculated as:

$$\rho = \frac{f * 8}{RTT}$$

2. **Sliding-window flow control (no error)**

If (d_{trans}*W >= RTT)

Expected throughput, ρ Mb/s, is calculated as:

$$\rho = \frac{f}{f + h} * dr_u$$

else

Expected throughput, ρ Mb/s, is calculated as:

$$\rho = \frac{f * 8 * W}{RTT}$$

3. **Stop-and-wait error control**

   - Average retransmission time for 1 error, $t_{retransmit}$ s, is calculated as:

$$t_{retransmit} = (d_{trans} + t_{timeout}) * \ error_r$$

   - Expected throughput, ρ Mb/s, is calculated as:

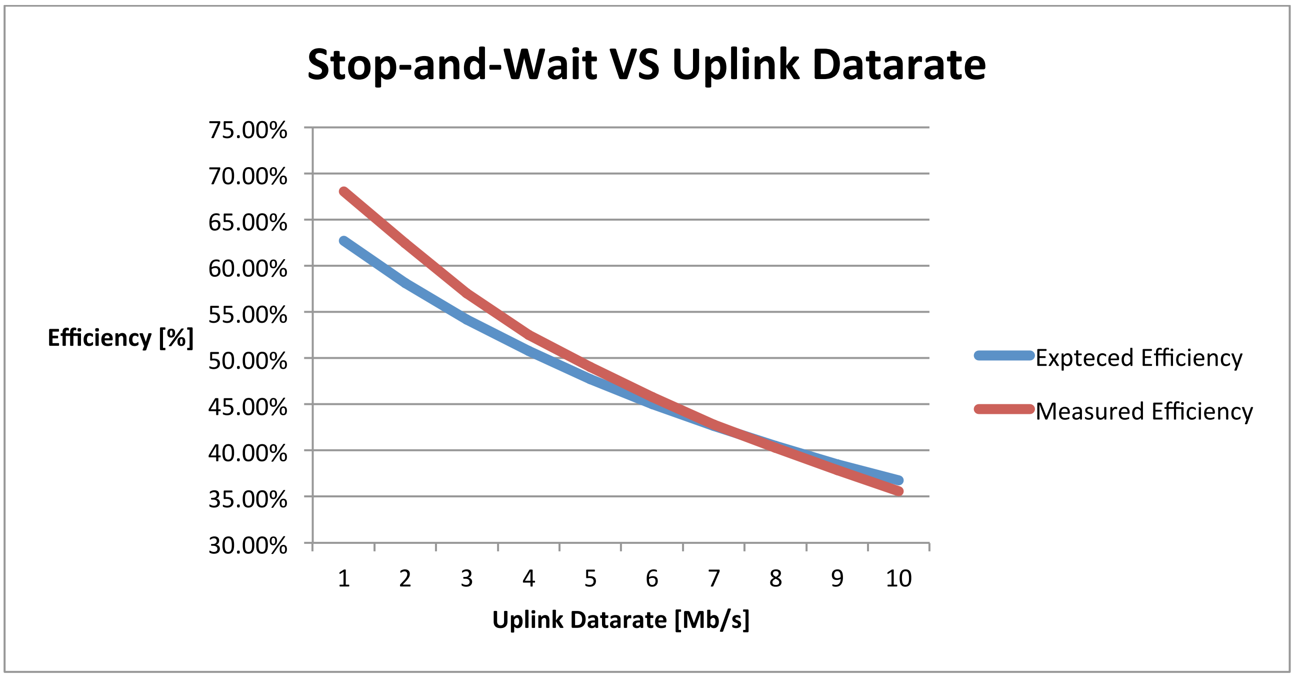
$$\rho = \frac{f * 8}{RTT + \ t_{retransmit}}$$

Expected efficiency for all protocols, η, is calculated as:

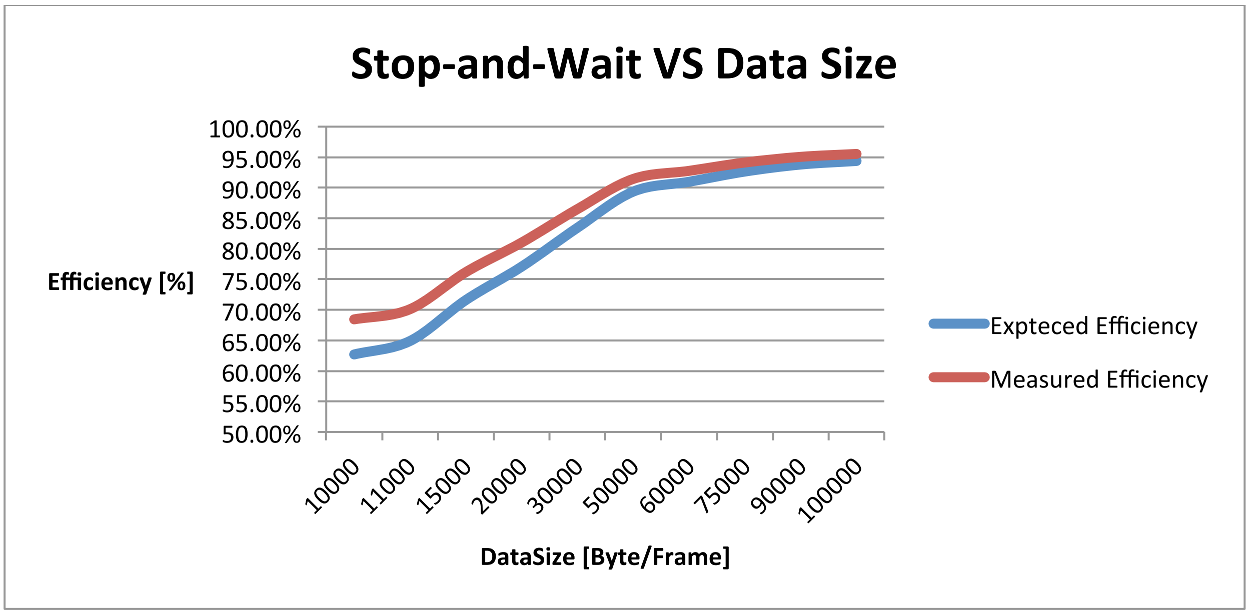
$$\eta = \ \rho \,/\, dr_u$$

5 Results and Conclusions

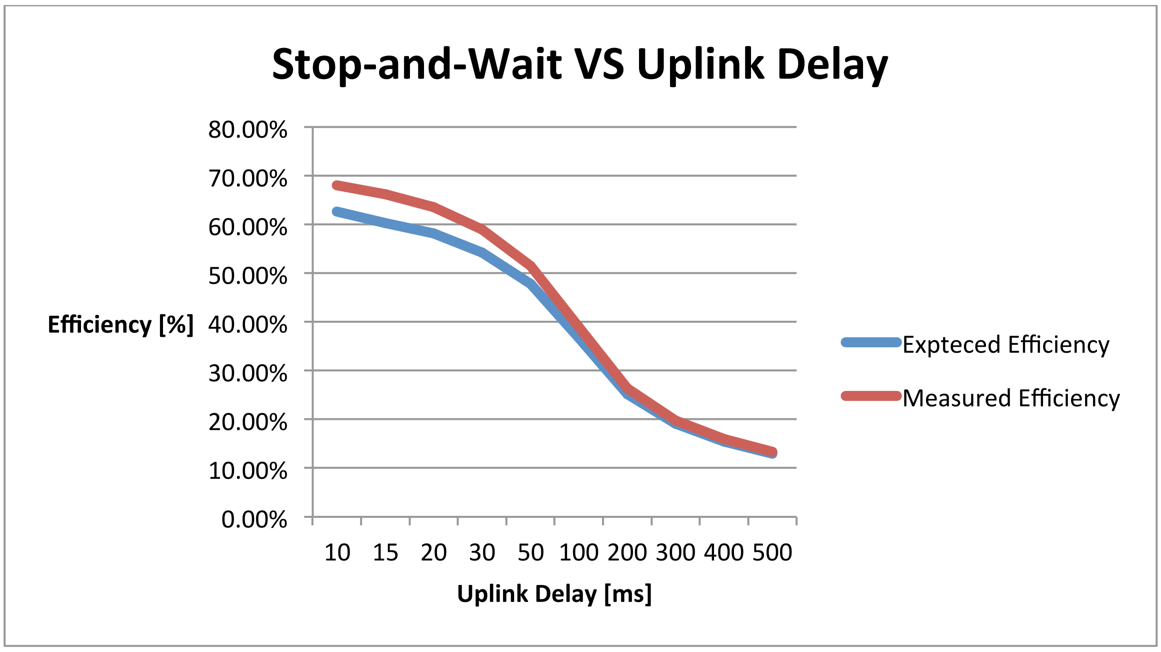
5.1 Stop-and-wait flow control



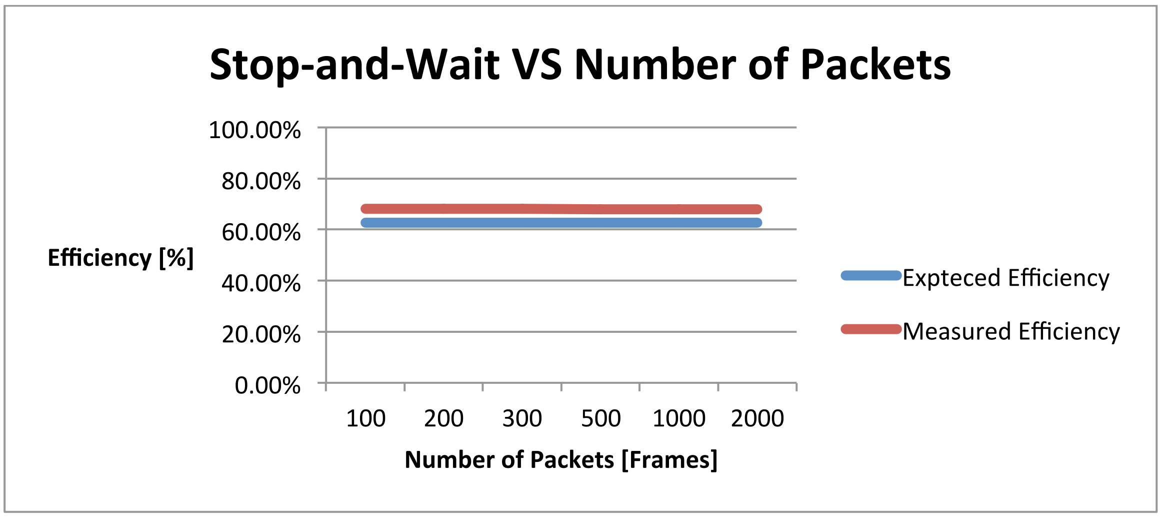
We observed that with increasing the uplink datarate, the efficiency of stop-and-wait flow control decreased. Although the throughput has increased, the efficiency level dropped because the value of transmission delay relative to the delay is more significant since we have to divide the throughput by a higher uplink data rate. The measured efficiency follows the trend of the expected efficiency.



We observed that with increasing the data size, the efficiency of stop-and-wait flow control increased. The increase was much more significant when the data size was within the range of 11000 to 50000 Bytes. This is because as we increased the data size, the header size becomes less significant. The measured efficiency was a little higher than the expected efficiency but follow the same trend.

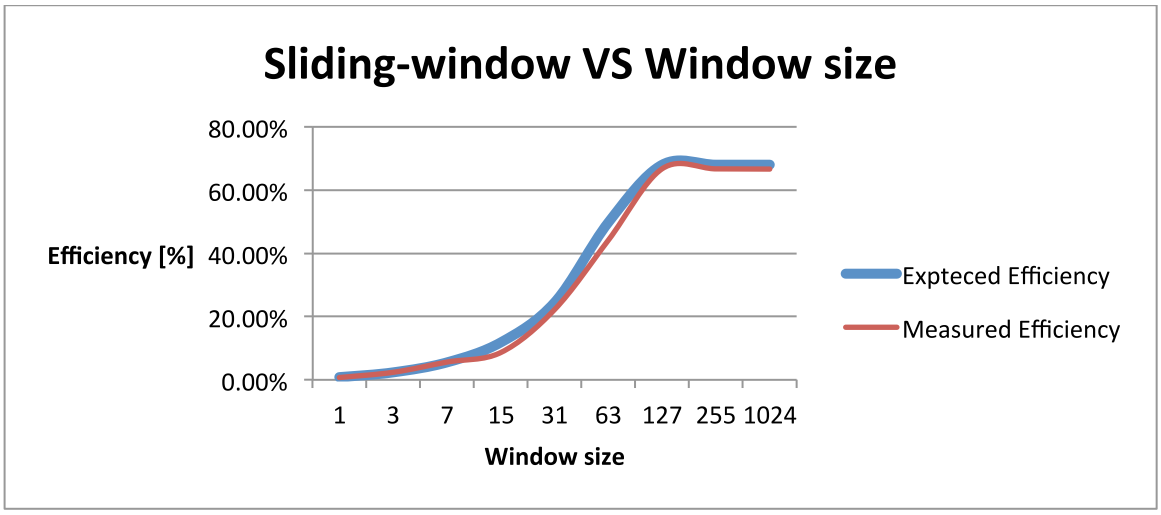


We observed that with increasing the uplink delay, the efficiency of stop-and-wait flow control fell. This is because the uplink delay is larger compared with the transmission delay, which in turn increases the RTT.

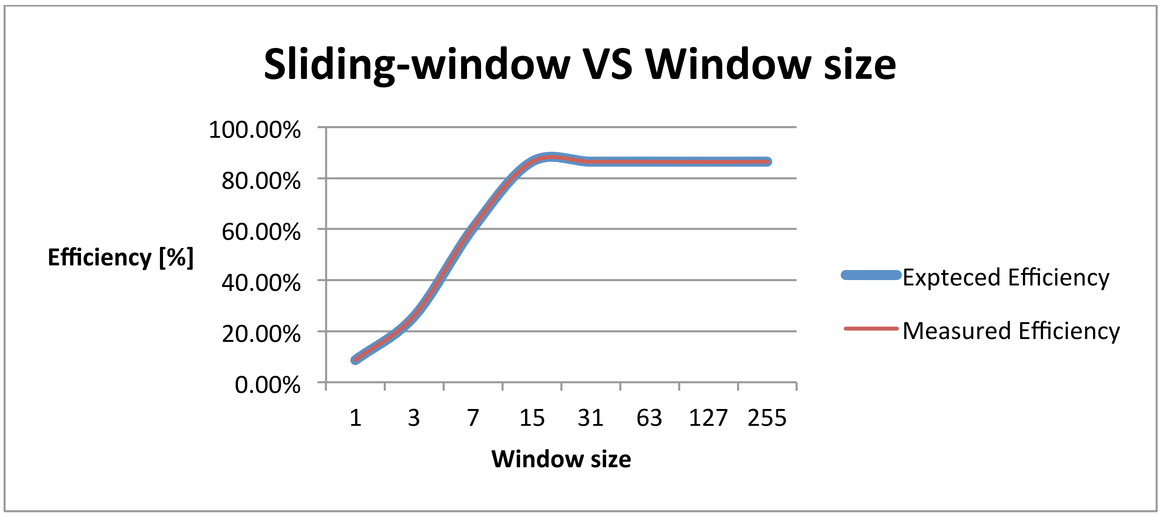


We observed that with increasing number of packets, there is no change in efficiency for stop-and-wait flow control. This is because when we calculate the throughput for stop-and-wait flow control, we only calculate based on one packet.

5.2 Sliding-window flow control

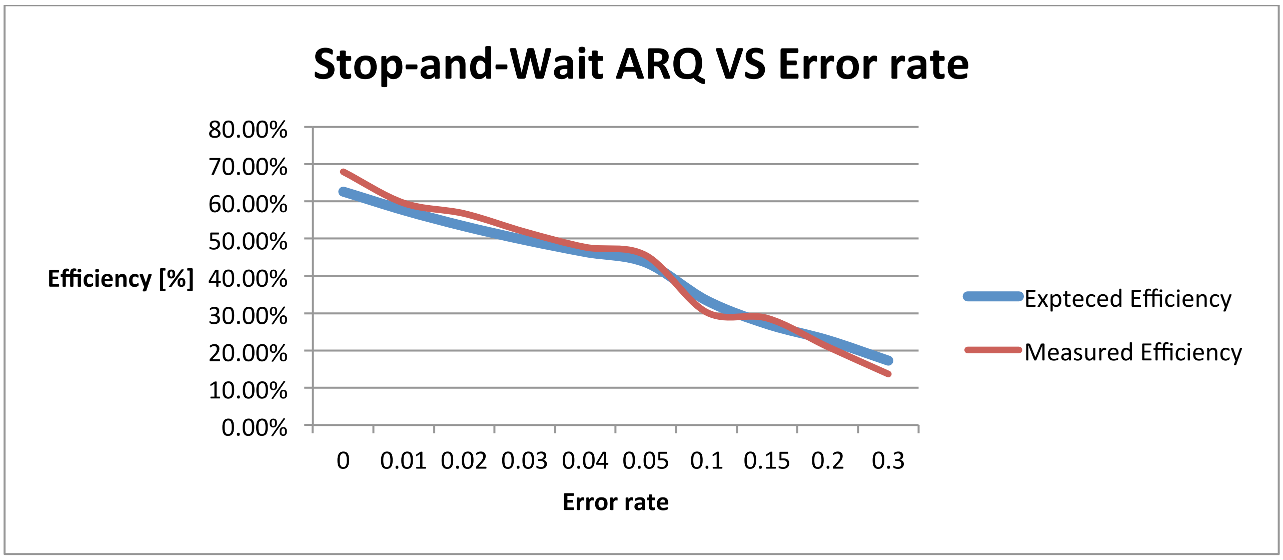


In this particular example, we set the data rate to 10 Mb/s and the delay to be 1000ms. By our calculation for data size of 10000 Bytes, the maximum efficiency can be achieved by setting the window size to be at least 87 frames. We observed that the efficiency of sliding-window protocol increased when the window size is the range of 1 to 127. Then it levels off at the efficiency around 68%. This is what we expected since the maximum efficiency can be achieved at 87 frames.

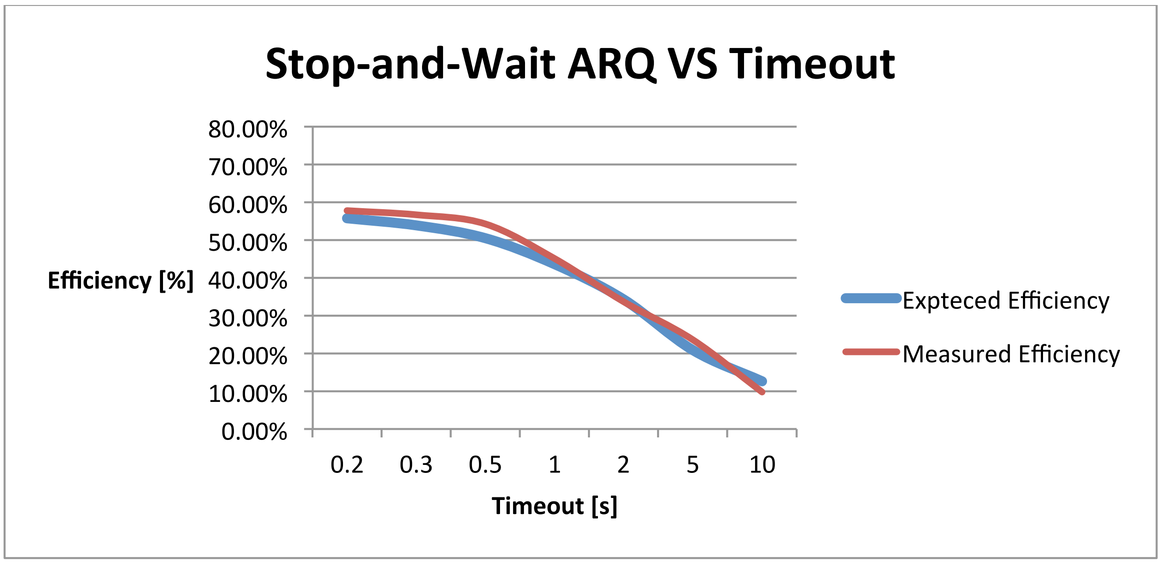


In this example, we set the data rate to 5 Mb/s and the delay to be 500ms. By our calculation for data size of 30000 Bytes, the maximum efficiency can be achieved by setting the window size to be at least 11 frames. We observed that the efficiency of sliding-window protocol increased when the window size is the range of 1 to 15. Then it levels off at the efficiency around 86%. This is what we expected since the maximum efficiency can be achieved at 11 frames.

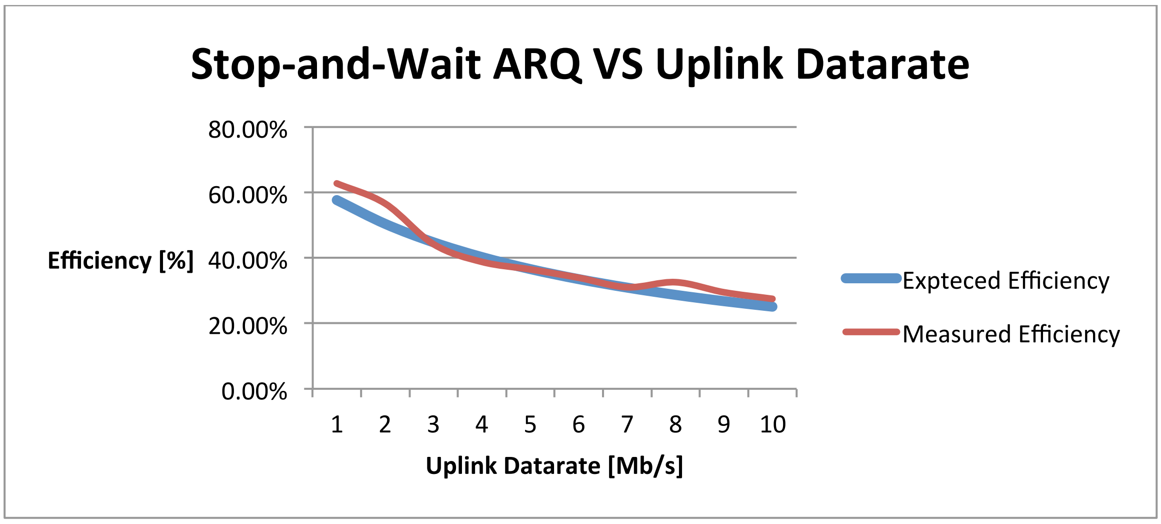
5.3 Stop-and-wait error control



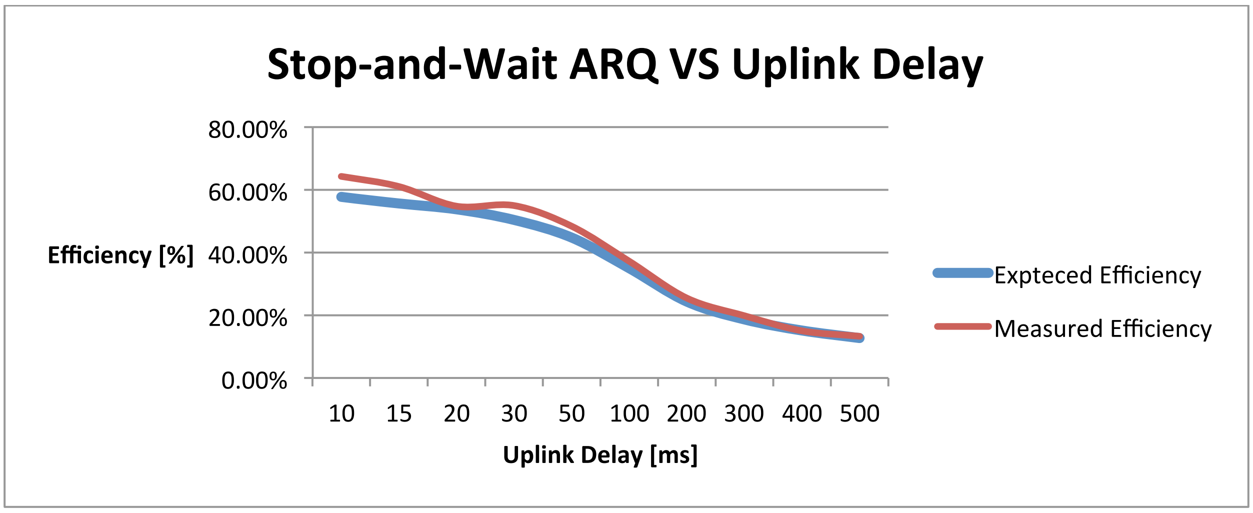
We observed that with increasing the error rate, the efficiency of stop-and-wait ARQ decreased. This is because as there are more chance of error, the protocol will have to take additional time to retransmit the packets. And this increase the overall duration of the packets transfer.



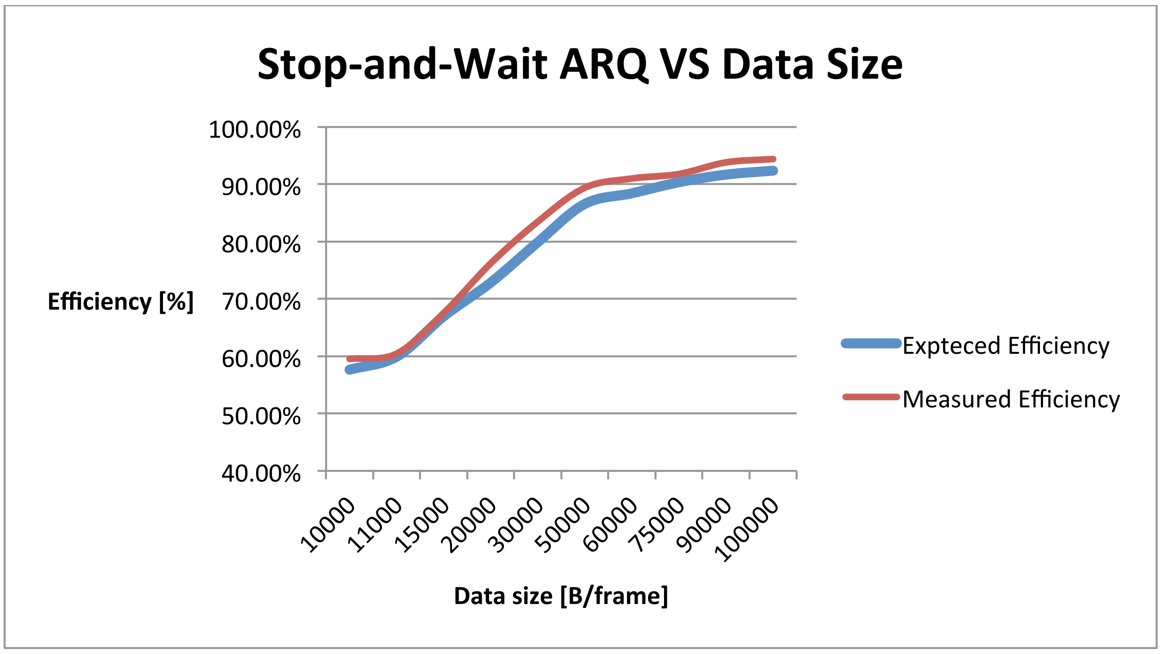
We observed that with increasing the timeout the efficiency of stop-and-wait ARQ declined. The drop in efficiency was much more significant when the timeout was within the range of 1 to 10 seconds. This is because of the additional time required to retransmit dropped packages.



In this example, we can see that the efficiency of stop-and-wait arq decreased as the uplink data rate increases. It shows the same trend as the efficiency of stop-and-wait flow control against the uplink data rate. The error rate and the timeout only shifts the line downwards but the trend stays the same.



We observed that with the increase in the uplink delay, the efficiency of stop-and-wait ARQ declined. This is because the uplink delay is larger compared with the transmission delay that in turn increases the RTT. Again, the error rate and the timeout only shift the line downwards.



We observed that with the increase in the packet size, the efficiency of stop-and-wait ARQ increases. This is because as we increased the data size, the header size becomes less significant. This graph shows the same trend as with the stop-and-wait flow control against the data size but gets shifted down due to the error rate and timeout values.