## Question 1.

### 

The calculation of the throughput is similar to the exercise in the Tutorial lecture, where *n* packets were sent every time. In our case, n=2. The equations are still fully developed here:

Given that we send 2 frames together:

Thus:

Calculating the throughput:

### 

As we can see, the throughput is dependent on 2 parameters – and . By taking the throughput from the original protocol, which is:

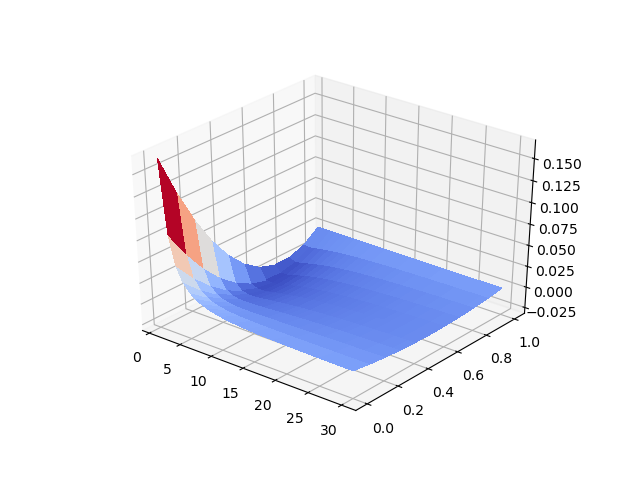
We are required to find the for which the equations holds:

For different values, the which results in the inequality is different. We use the graph to visualize. Using python, the throughput values were calculated for various values. 2 graphs are shown:

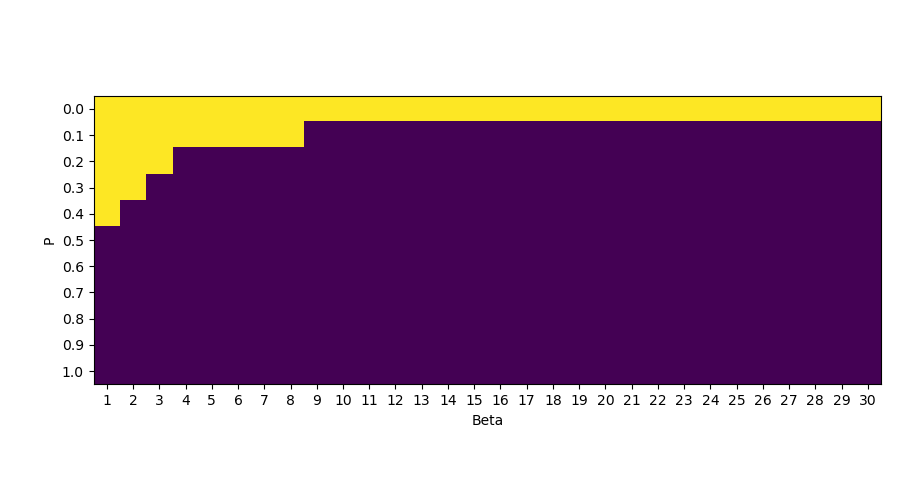
The first graph shows the result of the difference of the throughput values:

Where this values is positive, meaning using the original protocol is beneficial. Ranges used:

Result:



As we can see, where the probability of packet not being delivered is low, and Beta is low (propagation time is small), the original algorithm is more beneficial. To be sure, another binary chart shows yellow bricks where : (and purple if other)



We can observe that:

* With high error probability, the new protocol is beneficial
* If the error value is low, for certain Beta values we prefer the original algorithm
* If the error probability is 0, original protocol is better

All of those conclusions are also intuitive.

### 

Now, in the equation for the throughput calculation, , what changes is the value. Recalculating:

To validate our result, we calculate the average different between the average message in the original protocol, and in the protocol in this case:

We indeed see that on average the length of the successful transmission is less by :

* If error probability is 0, we always save time!
* As error probability increases, we save less time.

Which are intuitive conclusions.

## Question 2.

### 

First, calculating :

The optimal :

This is the optimal time. If after this time the ‘ack’ is not received, it is the sign that the message (or ack) didn’t reach the destination.

### 

The throughput is given by:

Where is the total time for 1 message for being processed

is a random variable showing the number of times the message is sent. Distributed geometrically with the probability of error . Thus, :

### 

The optimal window size would be the one, which can send maximal amount of messages before the for the first message has to arrive. This way, if there was no ack received, the first message will be resent. Calculating :

We use the ceil() function to round it up:

Thus, the window size is given by:

### 

Given:

The throughput is given again by:

As in tutorial, we define as number of times the message is resent. Then, for a given , the time until a message is transmitted:

The average time for 1 message to finish:

As in tutorial,

Thus,

Calculating the :

As we can see:

## Question 3.

### 

The scenario is similar to the one explained in the lecture, where it is promised that second package will reach the receiver with 100% certainty. To calculate the optimal window size, we take into account , which is not equal to .

Any increase in this window size will not benefit the throughput.

### 

The receiver optimal size is then:

### 

Calculating the :

is the probability of error on the first packet.

Summing up the probabilities of getting each transmission time for the packet to get the average transmission time for 1 packet.

### 

Finding the throughput through the equation: