EE2T21 bonus assignment 1

Christian van den Berg, Marijn Adriaanse & Guus Dohmen

### The assignment:

The goal of the assignment is to simulate the S&W protocol and to do several experiments with this simulation. In order to do this we have chosen to use python as our programming language, due to the familiarity and ease of use.

### How to run the program:

The program is configured to run all 3 experiments, as described in the assignment, at once. Therefor there is no configuration needed on the user side. However, as it was required in the assignment description, the possibility to manually define P1, P2 and the input list of integers has been added.

One can use *python3 SnW.py*  in order to run all experiments at once.

To run a custom experiment, one can use the following command: *python[3] SnW.py*  *<infile> <p1> <p2>*, in which *<infile>* refers to the file path of the input data file, and *<p1>* and *<p2>* refer to the probabilities p1 and p2, respectively. Note that Python 3 is usually available under either *python3* or *python*.

The input data file is a text file, in which each new row is treated as a new integer. An example of this can be found in the code repository.

### Experiment 1

In experiment one both P1 and P2 were zero. This means that we had an errorless data transmission. Due to the protocol only sending one frame at a time, the number of iterations needed to send all data is expected to be linearly proportional to the message length. Moreover, because in this experiment the communication is errorless, the number of iterations should be equal to the message length (in this implementation, the number of iterations is equal to the number of frames sent). The results reflect these expectations, as can be seen in figure 1.

### Experiment 2

In experiment 2, the chance of an error happening in the acknowledgement is left at 0 and the number of symbols is constant at 200. The goal of this experiment is to see the influence of the error probability in the message on the number of iterations the protocol has to do. Therefore, P1 is varied from 0 to 1 (not including 1) with a step size of 0.05. The results of this experiment can be seen in the graph below. As the probability of success is (1-P1), the expected number of transmissions needed is I=S/(1-P1). This means the number of iterations needed is inversely proportional to the probability of success. For P1=0, this means that I=S (as shown in experiment 1), whereas for values of P1 close to 1, the number of iterations needed ramps up to infinity. This is a logical result, since every message is guaranteed to get corrupted. The results can be seen in figure 2, and match the expectations described above. Each of the measurement points are the average of 500 independent runs.

### Experiment 3

In experiment 3, P1=P2=P. Both P1 and P2 are varied (in the same manner as in experiment 2), whilst the number of symbols is left constant at 100. This means that both the message and the acknowledgement signals have the same chance to be corrupted. Due to this, a similar relation between P and the number of iterations as in experiment 2 is expected. However, the chance of a successful frame transmission (successful transmission of a frame and its corresponding ACK) is now equal to (1-P)(1-P)=1-2P+P^2; the probability of success is squared. Therefore, the curve is expected to be vastly steeper than the function in experiment 2. The results of this experiment can be seen in figure 3. Here it is clear that the function is far steeper; for P=0.95, I is approximately 40000, which while in experiment 2 an error probability of 0.95 only required 2000 iterations (for 100 frames).

Chart, line chart

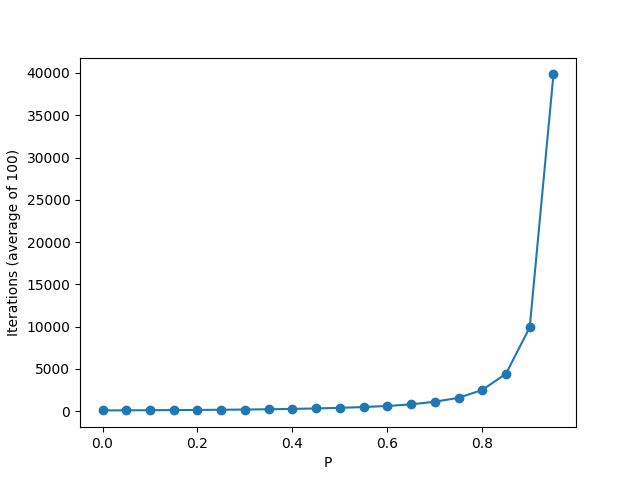
Description automatically generated

*Figure 1: Results of experiment 1.*

Chart

Description automatically generated

*Figure 2: Results of experiment 2.*



*Figure 3: Results of experiment 3.*