LAB PROJECT Nº1

PERFORMANCE OF WIRELESS LOCAL AREA NETWORKS

1 INTRODUCTION

The performance of wireless networks is highly dependent of physical phenomena, such as path loss, shadowing, absorption, scattering, diffraction and fading. The shared nature of the wireless medium makes communication even more difficult, since wireless stations must share or compete for the available bandwidth. Transmission errors caused by collisions and/or Signal-to-Noise-plus-Interference-Ratio (SNIR) degradation due to interfering stations are very common in contention Wireless Local Area Networks (WLANs), such as IEEE 802.11. This lab project aims to study some of the factors that affect WLAN performance by means of simple tests implemented in the OMNET++ simulator.

Throughout this document, the students shall be asked to build simulation script files. Although the main guidelines for scenario construction are provided, additional procedures may be necessary, which require initiative from the students. The students shall also be asked to answer questions regarding the simulated network environment and performance results. The answers shall be included in the report, which is due in the week that immediately follows the end of the project. Together with the report, the students must deliver all the scripts, statistics files and spread sheets that resulted from their work.

2 WORK DESCRIPTION

The fixed wireless access performance analysis shall be done in several steps, in which relevant independent variables shall be analyzed.

2.1 Impact of Distance in Ad Hoc Point-to-Point Communications

Within the OMNET++ IDE, copy the *hosttohost_LAB0* example into hosttohost_ptp. Modify the example, removing the access point (*ap*), so that the nodes *srvhost* and *clihost* communicate directly in IEEE 802.11 ad-hoc mode (see Figure 1). In order to accomplish this, you will have to change the *wlan*'s *mgmt.typename* attribute to *"Ieee80211MgmtAdhoc"* (instead of *"Ieee80211MgmtSTASimplified"*) in both stations. Since the objective of this test is to measure the impact of distance on the communications performance, we need to use a static mobility model allowing us to set the precise locations of the nodes in the scenario map. The required mobility model is the *StationaryMobility*, instead of *CircleMobility*. **In order to be able to freely assign node positions from the *omnetpp.ini* file, the *initFromDisplayString* attribute of the mobility modules, should be set to false.** Node positions are set using the *initialX* and *initialY* attributes of the mobility modules.

Figure 1: Modified *hosttohost* exemple.

Assume the following scenario parameters:

```
**.wlan*.bitrate = 1Mbps

**.radio.ccenterFrequency = 2.4GHz

**.mac.dcf.rtsPolicy.rtsThreshold = 3000B

**.wlan.mac.dcf.recoveryProcedure.shortRetryLimit = 7

**.wlan.mac.dcf.recoveryProcedure.longRetryLimit = 4

**.wlan.mac.dcf.channelAccess.cwMin = 31

**.radio.transmitter.power = 18.0mW

**.backgroundNoise.power = -110dBm

**.radio.receiver.sensitivity = -85dBm
```

```
**.radio.receiver.energyDetection = -85dBm

**.radio.receiver.snirThreshold = 0dB
```

The default path loss model is the Friis free-space model. Change the path loss model to *TwoRayGroundReflection*. This is done by setting the *radioMedium.pathloss.typename* parameter. In order for this model to work, the network's *.ned* definition must include a *physicalEnvironment* model, besides the *radioMedium*:

```
physicalEnvironment: PhysicalEnvironment {
    @display("p=49.704147,135.50298");
  }
```

In the *.ini* file, specify the *physicalEnvironment.ground.typename* parameter to *"FlatGround"*. As you may know, the *TwoRayGroundReflection* model is highly dependent on the height of the transmitter and receiver antennas. In the INET framework, by default, the height of the antenna corresponds to the *Z* coordinate of the node. **Set the antenna height to 1 meter.**

Using the *hosttohost_ptp* example that was just built, answer the following questions in your report:

Q2.1.1) Calculate the maximum communication range and experimentally check that it matches the maximum range of the simulator. Note: you may have a look at the `computePathLoss` function of the respective path loss model. You can find the path loss models in *inet/src/inet/physicallayer/pathloss/*.

Q.2.1.2) **Configure the sending application to generate a data rate that saturates the network capacity[1].** Set the application start time to 0s and the application stop time to 20s. Measure the impact of distance and packet size on the application throughput (in kbits/s), application message latency (s) and percentage of packets received with errors. Explain the obtained results. The considered distances between the *clihost* and *servhost* should be 5 equally spaced points between 1m, and the maximum transmission range plus 1m. Three packet sizes should be considered: 25, 600 and 1500 octets.

The results should be presented in three charts (throughput, latency and average number of MAC retries), each with three curves, one curve for each packet size. Each point in the chart should be simulated at least 3 times, using different random seeds, setting the simulation running time of the application to 20 seconds[2], starting from 0s. The confidence intervals for the points in the chart should be represented as error bars or presented in a table. Please read Section 10.4 of the OMNET++ Simulation Manual in order to get more info on how to repeat experiments with different random seeds. Explain the results.

[1] The network is considered saturated when, despite increasing the packet generation rate, the throughput (correctly received data per unit time) does not increase.
[2] Not to be confused with the overall simulation time limit.

The chart represented in Figure 2 is an example of what the intended graphics might look like (note: the curves are nonsense, were invented for illustration purposes and do not conform to actual results).

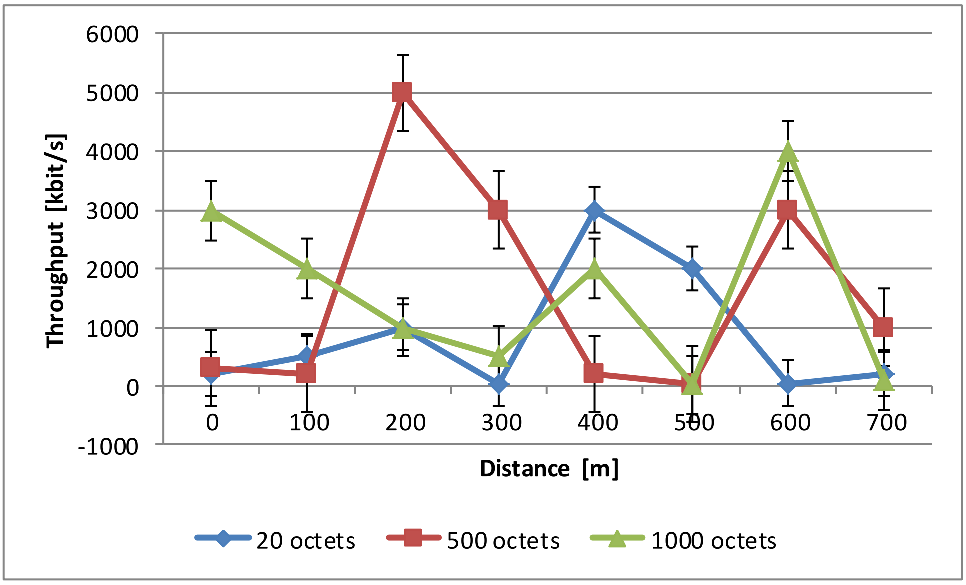

Figure 2: Example of chart exemplifying throughput as a function of distance between nodes.

Q2.1.3) Repeat Q2.1.3, but this time making `**.mac.dcf.rtsPolicy.rtsThreshold = 500B`. Explain the obtained results. Revert to the previous parameter configuration before proceeding.

Q2.1.4) Make `**.backgroundNoise.power=-84dBm` and check the new communications range running a simulation with packet size of 25 octets. Explain the results. Revert to the previous configuration before proceeding.

2.2 Performance of IEEE 802.11 Ad Hoc under Contention

Within the OMNET++ IDE, duplicate the *hosttohost_ptp* example, renaming it to *hosttohost_star*. Modify the example so that a variable number of client nodes are deployed in circle topology around a server node, with all nodes remaining within communications range of each other. The clients nodes should be located at equally sized angles around the server[3]. **Make sure that the sending applications are configures to generate data at a rate that saturates the network.** Other modifications may be required, which are not specified in this document. Using the *hosttohost_star* example that was just built, please answer the following questions in your report:

Q2.2.1) Measure the average total throughput (application layer), with the following numbers of client nodes: 1, 5, 25, 50 (average over 3 trials, application running time of 20 seconds). Consider packet sizes of 25, 600 and 1500 octets, with `**.mac.dcf.rtsPolicy.rtsThreshold=3000`. Explain the results.

Q2.2.2) Modify Q2.2.1, so that clients in opposite sides are outside of the interference range of each other. Repeat the simulations and compare the results.

Q2.2.3) Modify Q2.2.2, so that `**.mac.dcf.rtsPolicy.rtsThreshold=500`. Repeat the simulations and compare the results with Q2.2.1 and Q2.2.2.

Q2.2.4) Based on the simulations performed in Q.2.2.1, calculate the average total throughput measured at the MAC layer and compare with the application layer throughput. Explain the found differences.

[3] Since the angles depend on the total number of clients, you may wish to automate the procedure by using the following code:

```
SimpleWLAN.numCli = ${N = 1, 4, 20, 50}
**.cliHost[*].mobility.startAngle = parentIndex() * 360deg/${N}
```

2.3 Performance in Infrastructure Mode:

Return to the *hosttohost_LAB0* example, and duplicate it as *hosttohost_if*. Modify this example so that the parameterization is the same as in the previous sections. Modify the example so that a variable number of client nodes are deployed in circle topology, at equally sized angles, around the access point, all nodes within range of each other. The server can be located anywhere within range of the access point. Using this *hosttohost_if* example that was just built, please answer the following questions in your report:

Measure the average total throughput (application layer), with the following numbers of client nodes: 1, 5, 25, 50 (average over 3 trials, application running time of 20 seconds). Consider packet sizes of 25, 600 and 1500 octets, with `**.wlan.mac.rtsThresholdBytes=3000`. Compare the results with those obtained in Q.2.2.1, explaining the differences. **Note:** the sending applications should be configured to generate a data rate that saturates the network.

3 GRADING

Grading of the responses shall be performed according to Table 1.

Table 1: Grading of LAB1 responses.

Question	Value
Q2.1.1	2.0
Q2.1.2	2.0
Q2.1.3	2.0
Q2.1.4	1.0
Q2.2.1	3.0
Q2.2.2	3.0
Q2.2.3	2.0
Q2.2.4	2.0
Q2.3	3.0

The organization of the delivered report and scenario files contributes with 0.0-3.0 negative points to the final mark.