Concurrent and Distributed Programming Project

Mauro Conti, PhD

November 1, 2011

The students have to develop an JAVA application to simulate protocols for Wireless Sensor Networks (WSNs). In particular, the students have to simulate the working of the RED and the LSM clone detection protocols.

1 Specification

For the simulations, we assume that we have a given number of nodes `N`. These nodes are place on an unit square area. The location of each node n_i on this unit square area is specified by (x_i, y_i). In addition, the position of each node n_i is determined at random and this position does not change as time passes by. We also assume that each node n_i can communicate with all the nodes within its communication range `r`. Within this framework the students have to simulate the RED and LSM clone detection protocols.

The simulator has to be developed in such a way that it is easy to add and run flooding algorithms other then those required in this project. Other requirements for the simulation algorithm are that it has to be able to perform `NSIM` number of simulations; for each single iteration the simulator has to locally compute statistics. For all the statistics collected from the different iterations, the application has to compute the min / max / avg / variance of the data.

More specifically, considering the network graph where the vertices are the nodes of the network and arcs indicate the existence of the possibility of communication between two nodes. The output should contain the following statistical information:

- whether the network graph is connected
- the number of connected components in the network graph
- the time units required for the termination of the clone detection
- the energy consumption of the nodes

In addition to the requirements above the application has to be developed in such a way that it implements the following 4 steps:

1. make Socket connection to receive configuration file
2. parse the received configuration file and initialize the simulator

3. execute the simulation `NSIM` times and compute the statistics for each iteration
4. store the collected statistical data on the data server using RMI

In the first step, the simulation application has to connect to http://www.math.unipd.it/∼conti making use of a TCP Socket connection. After this connection is established, then the simulation application has to send a HTTP GET request to retrieve the configuration file `config.txt` which contains all the information (e.g. NSIM, N, (x_i, y_i) for each node n_i) needed to initialize the simulation application.

In the second step, the simulation application uses JAVA File I/O and standard JAVA parsing functions (e.g. StreamTokenizer) to respectively read and parse the configuration file. To make the reading and parsing of the configuration file much easier it is formatted in the following way. Each line of the configuration file is either formatted as `[VARIABLE_NAME] = [VALUE]`, or it is a comment line (starting with the string "//"). The comment lines should be ignored by the simulation application.

In the third step, the simulation application uses the configuration information collected from the configuration file to initialize the simulator. More specifically, this functionality has to initialize the unit square containing all N nodes at positions (x_i, y_i) as specified in the configuration file. Then simulations are executed NSIM times. The simulation application has to compute and store the required statistical information for each iteration locally.

Finally, in the last step all the statistical information that was stored locally is stored on a remote data server making use of RMI.

2 Non-Specified Aspects

We leave it to the students to make insightful decisions on all the matters concerning design choices that are not specified in this document. In addition, we strongly suggest the students to discuss all unclear matters in the mailing list of the course.