2. Übung, SS 2013
Verteilte Systeme/ Distributed Systems
Ausgegeben am 15.5.2013
Vorstellung 23.5.2013

Katinka Wolter
Benjamin Güldenring
Inst. für Informatik
Freie Universität Berlin

## Assignment 1. Peersim and gnuplot

If not already done download Peersim from `http://peersim.sourceforge.net` and download and install gnuplot from `http://www.gnuplot.info`.

### 1.1 Gossip Averaging

1. Run the cycle based gossip averaging example (`example/config-example1.txt`) with the `LinearDistribution` initializer control with min=0 and max=100 for the network sizes 500, 5000 and 50000. Collect the results from each run and plot the variances of each run together in one single window (see hints below). What does the result tell us about the gossip averaging protocol? Solution: the averaging protocol scales very well, i.e. linear.

2. Run the event based gossip averaging example (`example/config-edexample.txt`) with the `LinearDistibution` initializer control with min=0 and max=100 for the network size 1000, using

   - at first MINDELAY, MAXDELAY, DROP = 0
   - and then MINDELAY=50, MAXDELAY=100, DROP=0.5

   and compare the results. Solution: the first one works, in the second a wrong result is calculated.

3. **Bonus Question** If something is different, why? Solution: Consider three nodes A(3), B(1) and C(2), the gobal average is 6/3=2. A and C get at the same time the value from B, average the value with theirs and send the results back. One of them is then overwriting the result of the other. Assuming C overwrites the result of A this leads to A(2), B(1.5), C(1.5). The global average is now 2.5. Since the global average can never be wrong (inductive argument) this must lead to an incorrect result.

### 1.2 Gossip Minimum

Using the source code from the averaging example, implement a protocol which calculates the minimum instead of the average:

1. as a cycle based protocol and

2. as an event based protocol.

3. Do the same evaluation with these protocols as in 1.1.2 and compare the results of gossip averaging with the gossip minimum calculation. Solution: The minimum calculation gives a correct result while the average calculation didn't

4. **Bonus Question** If something is different, why? Solution: No formal proof here but an informal argument: if a node has a minimal value it can never lose this value. Therefore the global minimum never changes. When we assume that at least every now and then a comparison is successful (i.e. the result is written back and a node is updated with the minimal value) we can conclude that the algorithm finally converges.

## Hints

Set your java classpath to include the Peersim directory and the folder where you plan to solve the assignments. This way you don't always have to provide the parameters `-cp <peersim files>` to java.

If you are using a UNIX like system you can put the following line

```
alias peersim="java peersim.Simulator"
```

in your `~/.bash_profile` file to be able to execute the Peersim simulator with the shorter command `peersim`.

### Gossip Averaging and Peersim

You might want to have a look at the tutorials

- `http://peersim.sourceforge.net/tutorial1/tutorial1.pdf`, and
- `http://peersim.sourceforge.net/tutorialed/tutorialed.pdf`.

for a better understanding of the example source code and the averaging algorithm.

### Gnuplot

The output of peersim is in general ready to be plottet using gnuplot without further processing. If you e.g. evaluate a protocol with 500, 5000 and 50000 nodes and write the results from Peersim's `stdout` in the files `n500`, `n5000` and `50000`, you can plot all of them in one window (and showing only the x interval $[0, 5]$) using the following gnuplot command:

```
gnuplot> plot [0:5] "n500" using 2:7 title "n=500", \ >"n5000" using
2:7 title "n=5000", \ >"n50000" using 2:7 title "n=50000"
```

Using the AverageObserver control the output of the Peersim simulator contains in column 2 the number of the current iteration step and in column 7 the variance.