Peer-to-Peer Systems
Exercise 4
Published: 11.12.2014
Presentation date: 18.12.2014

Peer-to-Peer Systems – Exercise Winter Term 2014/2015

General Remarks

Welcome to the exercise for the lecture Peer-to-Peer Systems.

Please follow the general remarks regarding the organization of the exercise.

- The lecture's website is to be found here: `http://tsn.hhu.de/teaching/lectures/2014ws/p2p.html`
- For further inquiries, please contact the lecturer under the following email address: `graffi@cs.uni-duesseldorf.de`

Problem 4.1 - Visualization of Chord on PeerfactSim.KOM

In order to evaluate peer-to-peer protocols, simulations are typically used. They allow to investigate the behavior and interactions of a large set of nodes. e.g. 1000 nodes. Simulations allow to measure the expected quality of service of a protocol (e.g. Chord) under specific circumstances. In this exercise, we evaluate the performance of Chord with 51 nodes in a visualized environment. In order to answer the following questions, please download and install the p2p simulator PeerfactSim.KOM from `http://www.tsn.hhu.de/teaching/lectures/2014ws/p2p.html`. You also have to download and install gnuplot `http://www.gnuplot.info/` to visualize the results.

a) Visualizing Chord

Simulate the config-file /config/visualization/chord.xml. In the visualization you can switch between two different views for Chord. Describe what is used as a basis for the positions of the nodes in Chord in the schematic and the topological view. Also describe in this context the meaning of GNP - global network positioning. Please make following modifications. Change the churn to be active from minute 30m on instead of minute 90m. Please also let the nodes in Latin

America start lookups from minute 35 on and extend the simulation time to 120m.

b) Inspecting the simulation results

Having gained insights in the configuration and the visualization of the overlay, next we inspect the simulation results. With the previous simulation of Chord the simulation results have been written to the outputs folder. The folder with the current date as the suffix, the string "chord" and the seed as the suffix lists the metrics corresponding to the simulation. The gnuplot script files have been also generated automatically. Please note, that not all gnuplot-files are always applicable, thus errors might be thrown if you try to plot them all. Make a careful selection and plot the corresponding graphs to answer the following questions:

- What is the number of nodes in the simulation, how does it change over time?
- When have lookups been initiated and resolved. How many lookups does a node initiate in maximum?
- What is the average and median hop count per lookup?
- What is the overhead (in terms of forwarded messages, served messages and outgoing traffic) over all nodes, is there an imbalance?

Problem 4.2 - Comparison of Chord and Pastry

In this exercise, we compare Chord to Pastry with respect to the amount of messages both overlays send to maintain the network.

a) Chord and Pastry

For comparison, please simulate the overlays Chord and Pastry with 3 seeds each and compare the results. Here the configuration files from config/education/ are to be used. What are the differences in terms of sent and received messages, especially on the network layer?

b) Observations

Explain the outcome of your observations from the previous task a).

Problem 4.3 - Extension of the EduStar overlay

In the Peer-to-Peer lecture you have been introduced to a very simple overlay, the *EduStar*. In this exercise it is planned to extend and change the overlay slightly. The overlay can be found in the package *org.peerfact.impl.overlay.dht.edu.star* of the simulator.

a) Gnuplot

First, create a simple gnuplot script which allows you to visualize the number of received ping messages in the network. For this task you might consider the simple analyzer class *EduAnalyzer*.

b) Online Nodes

Next, extend the analyzer and your gnuplot script with the possibility to plot the number of active nodes in the network.

c) Churn

As a last step we want to add churn behavior to our simple overlay. Therefore you can extend the method *connectivityChanged(ConnectivityEvent ce)* in the class *StarNode*. Whenever the simulator executes a generated churn event so that the connectivity of one node is affected, this method is called. Now, add the functionality to stop the periodical sending of ping messages in case one node goes offline. If the same node changes it's connectivity to *online* again, it should start the periodical operation of sending ping messages to the center node again. Try to visualize the churn behavior by using the *EduAnalyzer*.