LSINF2345 Project:

The Peer Sampling service

8 December 2020

Course — LSINF2345 – Languages and Algorithms
Authors — Thibaut Piquard - Stanley Goffinet
Professor — Van Roy Peter
Academic year — 2020–2021
Study Program — Master in Computer Science

UCLouvain

1 Criteria 1 : Deployment of bootstrap network.

Our first mission was to deploy a bootstrap network. Our implementation takes the form of a circular double-linked list. Adding node to the network is made in two steps. First we add them in the list and then one all of them are in place we initialize them. This allows us to return their neighbours on initialization. We have created multiple tests to verify the accuracy of the list. A call to *test_network* allows to test the add operation.

2 Criteria 2 : PS service implementation.

To test our implementation, we need to launch *launch(H,S,C,Peers,PushPull,Time,N)*. The arguments are :

1. **H** : self-healing parameter
2. **S** : swapper parameter
3. **C** : the maximal size for the node view
4. **Pees** : the two peer selection policies (can be *rand* or *tail*)
5. **PushPull** : the two view propagation policies (can be *true* of *false* if we want launch the pushPull policy or not)
6. **Time** : the Time in milliseconds between two cycles
7. **N** : the number of Nodes

3 Criteria 3 : Deployment of experimental scenario (s) and conclusions.

Our last mission was to deploy our implementation in a different way. Thanks to the lauch function which is fully parameterized, mentioned in point two, it is easy for us to change its parameters to study and analyze all useful cases.
To launch the different scenarios, we built a python script. First, it compiles all the erlang files (node.erl, network.erl and node_functions.erl). Then it executes the lauch function in 4 ways.

1. **healer_rand()** : Selectpeer with random selection and H = 4 and s = 3
2. **healer_tail()** : Selectpeer with random selection and H = 3 and s = 4
3. **swap_rand()** : Selectpeer with tail selection and H = 4 and s = 4
4. **swap_tail()** : Selectpeer with tail selection and H = 3 and s = 4

As for the other parameters, they are equal for the 4 scenarios. The time between each cycle is 3000 milliseconds, the maximum size of the view is 7 elements, the propagation rule is pushpull, which is an obligation because of the push rule which is useless according to the literature, and finally N equals 128 nodes.

The creation of the log.txt file in the script makes it possible to record and save all the outputs of the executed erlang command. Each line of the file includes as information the cycle, the id of the nodes as well as its view made up of the nodes with which it has already communicated.

Once the log file has been filled in, the script begins a phase of reading and processing the log file to build the 4 files.data These files, once launched in the console with the gnuplot command, will create the graphs that will allow a better analysis of the implementation of the ps service.

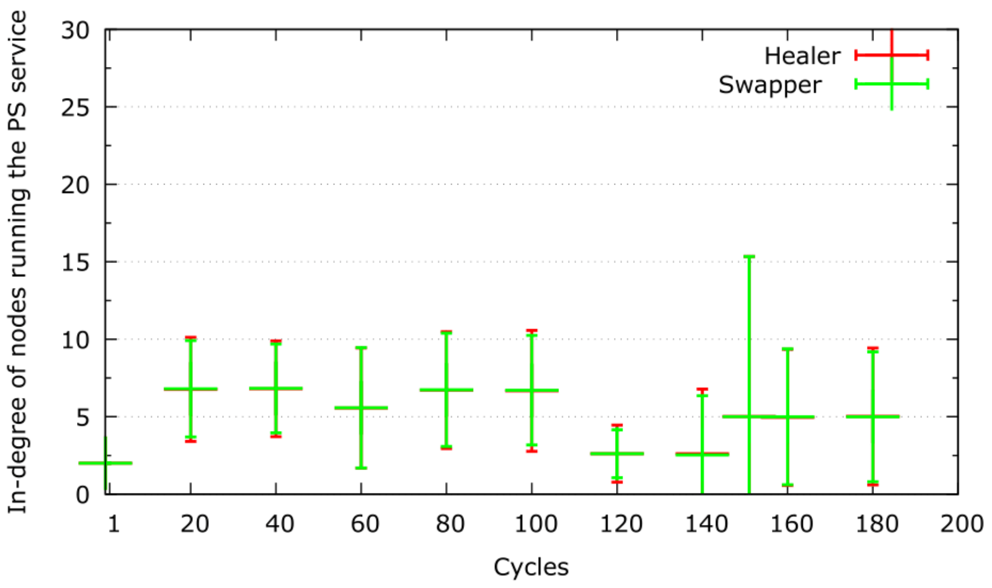

Figure 1: Random selection

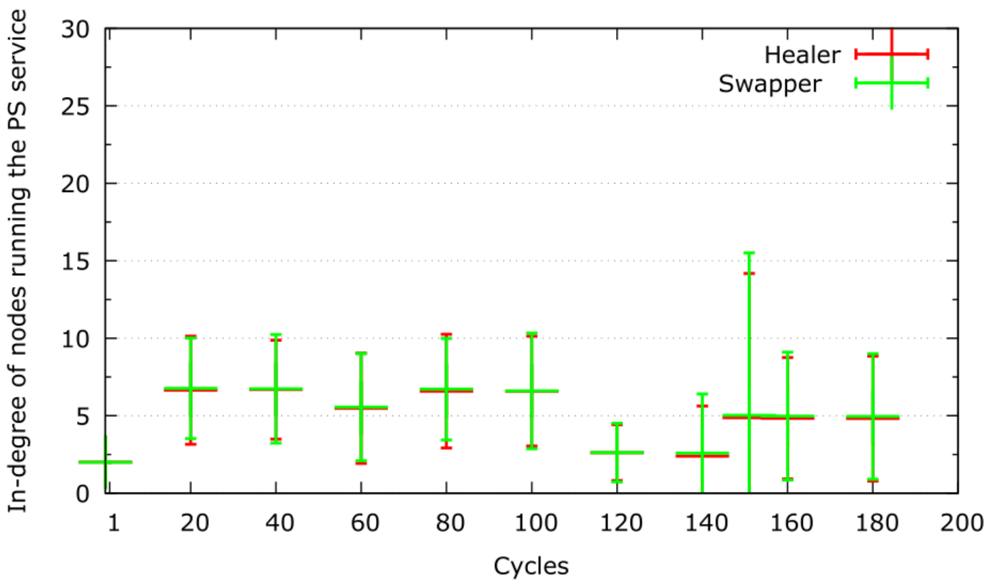

Figure 2: Tail selection

As we can see on the graphs, the Selectpeer parameter has very little influence on the scenario. On the other hand, as expected, the crash phase strongly decreases the standard deviation and the global indegree average. Thereafter the recovery phase in cycle 151 increases the standard deviation enormously because the new view of the majority of the nodes is composed of only one node while the old ones have 6 or 7 nodes. The analysis ends on a phase of restabilization which tends towards 5.