Distributed Systems Group (DS)
Department Software Technology
Faculty EEMCS
DELFT UNIVERSITY OF TECHNOLOGY

DISTRIBUTED ALGORITHMS (IN4150)
LAB EXERCISES

Exercise 3b

Implementation of Minimum-Weight Spanning Trees

D.H.J. Epema

December 3, 2019

1 Goal

The algorithm for creating the Minimum-Weight Spanning Tree of asynchronous weighted networks of Gallager, Humblet, and Spira is a milestone in the area of distributed algorithms. In this exercise this algorithm has to be implemented.

2 Assignment

Implement the the algorithm for creating the Minimum-Weight Spanning Tree of asynchronous weighted networks of Gallager, Humblet, and Spira (Algorithm 4.30 of the lecture notes). The implementation can be done in Java/RMI or Python. In designing, implementing, and testing your algorithm, take into account the following issues:

1. Your program should be truly distributed in that processes in the distributed algorithm run on multiple machines (so don't use a single JVM on a single machine that simulates all processes; it is of course allowed to have a single JVM in one machine simulate multiple processes).
2. Build into your program artificial random delays before processes perform their actions.
3. First test the correctness of your program for small networks.
4. Also test the correctness of your program for networks of any size that your program allows for which the solution is obvious.
5. Try to drive the execution of your program to a number of processes that is as large as possible.

3 Report

Write a short report (about 4 pages) in which you list for each test case that you run:

- the numbers of messages of the different types sent/received
- the number of merges and absorbs that have occurred
- the level and the core of all intermediate and the final MST

Run your program multiple times on the same network and check that always the core and the level of the final result is the same. Don't include the algorithm itself or its explanation in the report.