CS 6380: Distributed Computing
Section 001
Project 1

Instructor: Neeraj Mittal

Assigned on: Wednesday, February 1, 2023
Due date: Wednesday, February 22, 2023 at midnight

You can work on this programming project either individually or in a group. A group can contain up to two students. *Code sharing among non-team members is strictly prohibited and will result in disciplinary action being taken.*

You can do this project in C, C++ or Java. Each student is expected to demonstrate the operation of this project to the instructor or the TA. Since the project involves socket programming, you can only use machines dcXX.utdallas.edu, where XX $\in$ {01, 02, .., 45}, for running the program. Although you may develop the project on any platform, the demonstration has to be on dcXX machines; otherwise, you will be assessed a penalty of 20%.

1 Project Description

This project consists of three parts: (a) build a message-passing synchronous distributed system in which nodes are arranged in a certain topology (given in a configuration file), (b) implement Peleg's leader election algorithm to select a distinguished node, and (c) build a breadth first spanning (BFS) tree rooted at the distinguished node.

You can assume that all links are bidirectional. You will need to use a synchronizer to simulate a synchronous system. Details of a simple synchronizer will be discussed in the class.

Output: Each node should print the following information to the screen when appropriate: (i) UIDs of its parent and children nodes in the BFS tree, and (ii) its degree in the BFS tree.

2 Submission Information

All the submissions will be through eLearning. Submit all the source files necessary to compile the program and run it. Also, submit a README file that contains instructions to compile and run your program as well as the names of *all your team members*.

Only one group member should submit the files.

3 Configuration Format

Your program should run using a configuration file in the following format:

The configuration file will be a plain-text formatted file no more than 100kB in size. Only lines which begin with an unsigned integer are considered to be valid. Lines which are not valid should be ignored. The first valid line of the configuration file contains **one** token, denoting the number of nodes n in the system. After the first valid line, the next n lines, say L_A, consist of three tokens. The first token is the UID of the node. The second token is the host-name of the machine on which the node runs. The third token is the port on which the node listens for incoming connections. After the first $n+1$ valid lines, the next n lines, say L_B, consist of a space delimited list of at most $n-1$ tokens. The k^{th} valid line in L_B is a space delimited list of node UIDs which are the neighbor of the k^{th} node in L_A. Your parser should be written so as to be robust concerning leading and trailing white space or extra lines at the beginning or end of file, as well as interleaved with valid lines. The # character will denote a comment. On any valid line, any characters after a # character should be ignored.

You are responsible for ensuring that your program runs correctly when given a valid configuration file. Make no additional assumptions concerning the configuration format. If you have any questions about the configuration format, please ask the TA.

Listing 1: Example configuration file

```
# number of nodes in the system
5

# nodeUID hostName listeningPort
123   dc02  2234
5     dc03  3233
23    dc04  5217
1047  dc05  2432
89    dc06  6221

# space delimited list of neighbors for each node
5     23
123   1047
123   1047     89
5     23       89
23    1047
```

Any changes to the configuration file format has to be approved by the instructor or the TA.