EEL 6935: Distributed Computing Spring 2018 Instructor: J. Fortes

Homework 1 Assigned: 1/11/2018 Due: 1/23/2018 @ 5 Pm (use UF E-learning - Canvas)

Topology Management in Peer-to-Peer Systems

Preamble:

In this homework, you are required to write a C/C++/Java/Matlab program to implement the method for "Topology Management of Overlay Networks" described in Section 2.2.2 of Andrew S. Tanenbaum's Distributed Systems book and in attached reading supplement 1 (The paper "T-Man: Fast Gossip-based Constructions of Large-Scale Overlay Topologies" by Mark Jelasity and Ozalp Babaoglu). This algorithm is also known as Jelasity and Babaoglu's algorithm. The gist of this algorithm is discussed in the next paragraph (use the above references for the detailed descriptions).

In this algorithm, every node in the network maintains a list of neighbors. During the network-initialization phase, each node randomly selects k neighbors and places them into its neighbor list. During the network-evolution phase, in each cycle of the iterative algorithm, every node randomly selects one of its neighbors, and then sends a list consisting of the identifiers of its neighbors and of itself to that neighbor. The selected neighbor also sends its neighbors list back to the node which initiated the action. Upon receiving the new neighbor list, the nodes select the nearest k nodes from both the new and old lists as their neighbors and discards all the others.

You will write a sequential program that implements Jelasity and Babaoglu's algorithm so that in each cycle the nodes in the network initiate communication with each of their neighbors one by one. Your program MUST accept N, the total number of nodes in the network, and k, the number of neighbors each node maintains as the input parameters.

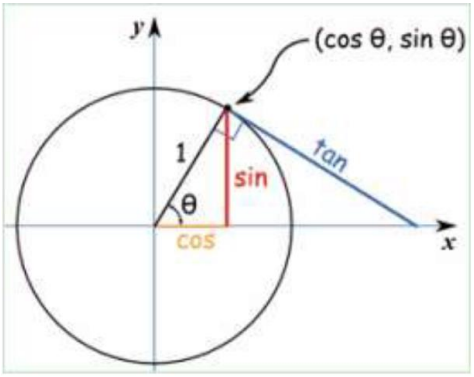
Tasks:

There are 2 tasks as part of this homework which are elaborated below:

1. (**"b" topology**): Assume there are *N* nodes in total. The first *N-1* nodes are aligned in the half circle of the "b" character, i.e. 1st Node (N_1) lies at the lowest point of "b" where N-1th Node (N_{N-1}) lies at the highest point on the half circle. Nth (N_N) is at the top of "b". Consider the case where the distance between two nodes, N_a and N_b, where $1 \le a \text{ and } b \le N$, is defined as follows. In a plane with N_a at $(\boldsymbol{x1}, \boldsymbol{y1})$ and N_b at $(\boldsymbol{x2}, \boldsymbol{y2})$, the distance is defined as a function of the node:

```
distance (Na, Nb) {
        if Nb == NN
                if Na == N1 || Na == NN-1
                        return 1;
                else
                        return Inf;
        else
                return |Na – Nb|;
}
```

Running the algorithm using this definition of distance will result in a network of shape similar to "**b**".
You can choose how to place the N nodes on the plane relative to the origin. For example, a node location can be given by $(cos\theta, sin\theta)$ where θ is the length of the curve or angle relative to the positive x-axis in radians.

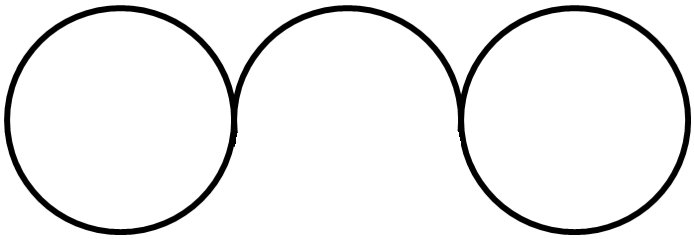


In that case, all the N-1 nodes are placed on the right half of a unit circle. That is,

$$x^2 + y^2 = 1\left(-\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2}\right)$$

The angle between adjacent nodes is then given by $\frac{\pi}{N-2}$. Hence, the nodes are located at ***(x, y)*** = ***(cos θ, sin θ)***, where $\theta = \left(\frac{\pi}{2} - (i-1) * \frac{\pi}{N-2}\right)$ and ***i=1, 2, … N-1.***

2. **(Spectacles Topology):** Change your code and the definition of distance so that a network shaped as a "spectacles" results from running the code instead of a "b" and show the corresponding node graphs.



Explain your definition of distance for this case.
Hint: Spectacles topology can be obtained by constructing two circles connected by another semi-circle. The semi-circle touches the two circles externally.

Additional Instructions:

- Using your implementation of the algorithm, you must report the sum of distances of neighboring nodes during the initialization phase and after each running cycle. The sum of distances between neighboring nodes is defined as

$$\sum_{i=1}^{N} \sum_{node_j \in neighbors(node_i)}^{N} distance(node_i, node_j)$$

  where neighbors $(node_i)$ indicates all the nodes stored in the neighbor list of $(node_i)$.

- In the homework report, you should show the results of testing your program using N=1000 nodes and k = 30 neighbors and running it for 40 cycles.
  - You can choose the radius for the semi-circle in 'b' topology and the "spectacles" topology to report in the homework.
- You are also required to draw a two-dimensional plot showing the sum of distances between neighboring nodes after each running cycle of the initialization phase. In the plot, the vertical axis represents the sum of distances and the horizontal axis represents the number of cycles. You can use any tool of your choice to generate the plot.

Questions:

a. What methods do you use to ensure that there are no separated nodes in the "b" and "spectacles" topology?

b. Can a node's neighbor list show the same node in multiple entries?

Submission (1/23/2018@5 pm): **No deadline extension requests will be considered**

1. Your source code containing the main method must be named TMAN.<extension of your program>. The names of the other supporting classes, if any, is up-to your choice.

2. Comment every module/class/function in your code and use descriptive variable names.

3. Your program MUST run using the command line arguments as shown below. It must accept the input arguments ***N, k*** and *topology*, where the topology can be one of **B or S** which represents "b-topology" and "spectacles topology" respectively. The total number of cycles is fixed at 40 for this homework.

Below is an example for the different languages.

- ***C: gcc TMAN N k topology***
- ***C++: g++ TMAN N k topology***
- ***Java: java TMAN N k topology***
- ***Matlab: matlab TMAN N k topology***

4. The output of the program is the sum of distance for each cycle. For 1, 5, 10 and 15 cycles the program produces the node graph files (png, jpg, gif, …) and a file that contains the neighbor list for each node. Any output file name should have a prefix - <topology>_N<number of nodes>_k<number of neighbors>.
   E.g., B_N100_k3

   Then, the sum of distances file is named <prefix>.txt
   E.g., B_N100_k3.txt

   The node graph file is name <prefix>_<current cycle>.<file extension>
   E.g., B_N100_k3_10.png

   The neighbor list file is named <prefix>_<current cycle>.<file extension>
   E.g., B_N100_k3_10.txt

5. There are two files that you will be uploading as part of the homework submission. A tar file and a PDF file. **Do not tar the PDF file.**
   a. Upload the following files within a SINGLE tar file named "yourLastName_hw1.tar":
      - The source code of your program
      - A file named "makefile" to compile the program. The TA will type only "make" to compile the program. Other methods are not allowed.
      - A text file named "readme.txt" that precisely specifies the running environment including the operating system, language written, compiler version and any software needed to r un your program. It also describes the program structure such as files, classes and significant methods. If your source code is in multiple files, describe briefly the content of each file.
      - All these files should be tarred into a file, i.e., the following command should work to create the tar file:

        *$> tar cvf <yourLastName>_hw1.tar **

   b. A PDF document (<yourLastName>_hw1.pdf) describing the definition of distance for spectacles topology and all results. Include the plots and the answers to the questions. Also, include your source code in the PDF. The source code MUST be pasted towards the end of the PDF. Include all your classes/modules/functions in the PDF file.

NOTE: the PDF file MUST NOT be tarred. You will have two files to submit. A tar that contains your source code, readme file and a make file, and a PDF file hat contains plots, answers to questions, definition of distance for the spectacles and pasted text of your source code.

6. The tar file and the PDF must be uploaded to Canvas.

Submission policy:

- Do NOT include binary files. Use the file names as specified above. Incorrect submission formats will lead to a grade reduction.
- All submissions are expected by the deadline specified in the homework assignment. Grade is automatically reduced by **25% for every late day**.
- Make sure to test your submitted code using the tar file. If untar, make, compile or any other command needed to execute your program do NOT work, your homework grade will be **zero**.