ECE 18-755: Networks in the Real World

HW3 Assigned 11/07/2017 DUE 11/21/2017

**Problems 1 and 2 are mandatory for full credit. Problem 3 is optional, you may choose to solve it for extra credit.*

Problem 1 (40 pts)

We have discussed in class about fault-tolerant communication for Networks-on-Chip based on stochastic communication. More precisely, we have learned that stochastic communication can be modeled as a branching and annihilating random walk as opposed to a classical (simple) random walk. This implies that at any point in time, and at any node reached by a disseminated packet, there is a possibility to have multiple or no starting random walks (for instance, based on forwarding or router/link failure probabilities). The goal of this problem is to study the differences between single/multiple random walks and stochastic communication.

Algorithm *k-Random Walks*: We assume k = 5 random walks reside at a *source* node in an $N \times N$ mesh network. At each iteration (called communication round), each random walk advances with a probability p_{f_kRW} = 0.9 to a neighboring location chosen at random.

```
Step 1. Initialize the adjacency[1] matrix $A$ of the mesh.
Step 2. For $i$ = 1 to Simulation_length
    For $k$ = 1 to 5
        if ($rand <= p_{f\_kRW}$)
            Randomly select a neighboring location of the random walker $k$ and move the
            walker to this location.
        end
    end
end
```

Algorithm *Stochastic Communication:* The stochastic communication protocol starts from a specific source node (i.e., a *Spreader* node) and it proceeds as follows: For each node from the *List_of_spreaders*, if the selected random number (between 0 and 1) is smaller or equal than the annihilation probability p_a = 0.002, then this node becomes an ignorant node. Otherwise, for each of its neighbors, select a random number between 0 and 1 and if this number is smaller than or equal to the forwarding probability p_{f_SC} = 0.3, then the packet is sent to that particular neighbor[2].

[1] The adjacency matrix A of a finite undirected graph of N vertices and E edges is a matrix of size $N \times N$, where each row corresponds to a distinct node; the non-zero entries in this row denote the presence of connecting edges between this node and any other node in the network. For instance, if $A(i,j)$=1, then it means that there exist an undirected edge between node i and node j. Note that we do not allow self-loops in these graphs so the diagonal elements of $A(i,i)$ = 0. Moreover, the sum of all the elements in the upper triangular part of the adjacency matrix A should equal the total number of edges E in the graph.

[2] Note that for these experiments, we neglect the annihilation possibility at the source node during the firstround.

```
Step 1. Initialize the adjacency matrix A of the mesh and List_of_spreaders = {source_ID}.
Step 2. For i = 1 to Simulation_length
            if (length(List_of_spreaders > 0))
                For k = 1 to length (List_of_spreaders)
                    if (rand1 <= $p_a$ && i > 1)
                        Add List_of_spreaders [k] to List_of_ignorants and remove List_of_spreaders [k]
                    else
                        For each neighbor of List_of_spreaders [k]
                            if (rand2 <= $p_{f\_SC}$)
                                Add this neighbor to List_of_spreaders
                            end
                        end
                    end
                end
            else
                break and hitting_time = $10^7$
            end
        end
```

a) Implement in Matlab (or a programming language of your choice) the above pseudo-code and record the number of visited nodes as a function of the number of communication rounds, the number of visited nodes (*i.e.*, nodes that are visited by any random walk (*k-RW*) or the nodes that receive at least one copy of the disseminated packet for stochastic communication) for the following two cases:

1) Without faults: N = 20, *source_ID* = (1,1) (assuming a (0,0) origin), p_{f_kRW} = 0.9, p_{f_SC} = 0.3 and p_a = 0.

2) With faults: N = 20, *source_ID* = (1,1), p_{f_kRW} = 0.9, p_{f_SC} = 0.3 and p_a = 0.002.

What differences do you observe? Which scheme offers the best coverage?

You should visualize the spatial coverage of each communication protocol and mesh configuration integrated over the entire simulation time. This visualization should illustrate the "hotspots" where packets tend to be sent; this way, one can see a stark difference between the two protocols. Also, indicate on your figures where exactly the faults occur. (**Hint**: you can use Matlab's *pcolor* plot to indicate the frequency of visits to a given node.)

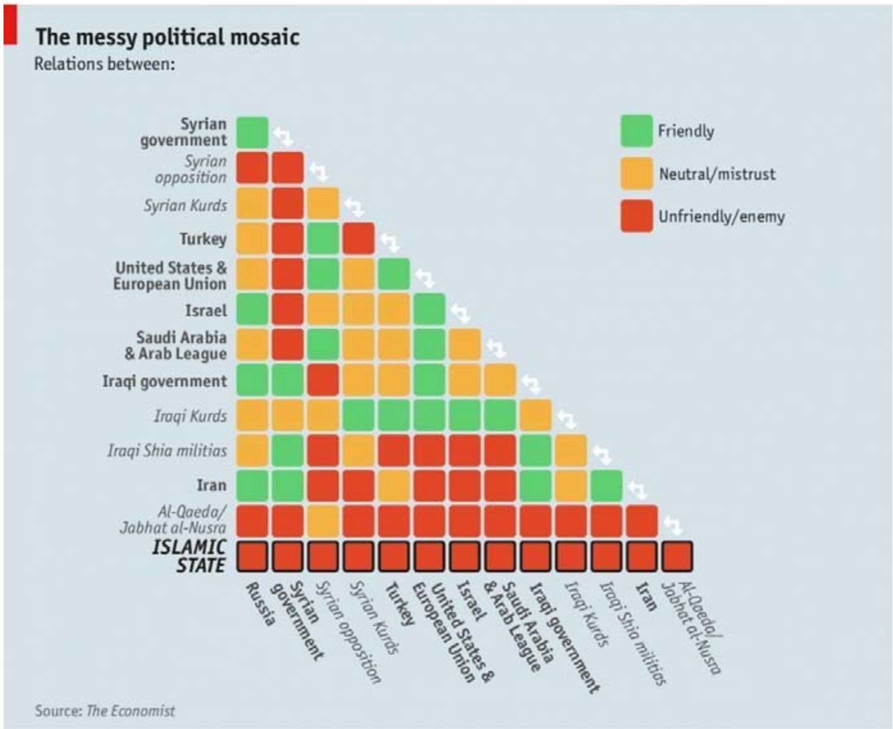
b) Based on the *k-RW* and stochastic communication approaches presented above, implement in Matlab (or a programming language of your choice) two scripts for computing the average hitting time (*i.e.*, time it takes for a destination node to be reached by at least one packet or random walker from a source node) obtained from 100 runs for the following two cases:

1) Report the average hitting time without annihilation as a function of network size and the following parameters: N = {4, 5, 6, 7, 8, 9, 10, 11}, *source_ID* = (1,1), *destination_ID* = (N,N), p_{f_kRW} = 0.9, p_{f_SC} = 0.3, and p_a = 0.

2) Now consider anihilation with p_a = 0.002. Compute and contrast the average hitting time with and without annihilation (i.e., p_a = 0.002 and p_a = 0), for the following parameters: N = 9, *source_ID* = (1,1), *destination_ID* = (N,N), p_{f_kRW} = 0.9, p_{f_SC} = 0.3, and p_a = 0.002. What differences do you observe? Which scheme offers the smallest hitting time? Why?

Problem 2 (35 pts)

Have you ever considered what it would take to resolve conflicts among political entities? If not, now's your chance! Imagine you are a diplomatic agent working for the United Nations and your main objective is to reduce the degree of conflict among the primary actors in the Middle Eastern political theater. You are given the following information about each political entity's relationship with one another.



	Russia	Syrian government	Syrian opposition	Syrian Kurds	Turkey	United States & European Union	Israel	Saudi Arabia & Arab League	Iraqi government	Iraqi Kurds	Iraqi Shia militias	Iran	Al-Qaeda/ Jabhat al-Nusra
Syrian government	🟩												
Syrian opposition	🟥	🟥											
Syrian Kurds	🟧	🟥	🟧										
Turkey	🟧	🟥	🟩	🟥									
United States & European Union	🟧	🟥	🟩	🟧	🟩								
Israel	🟩	🟥	🟧	🟧	🟧	🟩							
Saudi Arabia & Arab League	🟧	🟥	🟩	🟧	🟧	🟩	🟧						
Iraqi government	🟩	🟩	🟥	🟧	🟧	🟩	🟧	🟧					
Iraqi Kurds	🟧	🟧	🟧	🟩	🟩	🟩	🟩	🟩	🟧				
Iraqi Shia militias	🟧	🟩	🟥	🟧	🟥	🟥	🟥	🟥	🟩	🟧			
Iran	🟩	🟩	🟥	🟥	🟧	🟥	🟥	🟥	🟩	🟧	🟩		
Al-Qaeda/ Jabhat al-Nusra	🟥	🟥	🟧	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	
ISLAMIC STATE	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥

🟩 Friendly
🟧 Neutral/mistrust
🟥 Unfriendly/enemy

Fig.1: Political mosaic involving different countries

In this problem, you will be exploring the political network shown above within the scope of Structural Balance theory. The adjacency matrix depicted above is provided within *political_EdgeList.csv*.

The theory of Structural Balance is a classic social theory from Gestalt psychology. The theory stipulates that friends of friends cannot be enemies and enemies of enemies cannot be enemies. The following figure (Cartwright & Harary, 1956) demonstrates these relationships with signed graphs where dashed lines are negative ties.

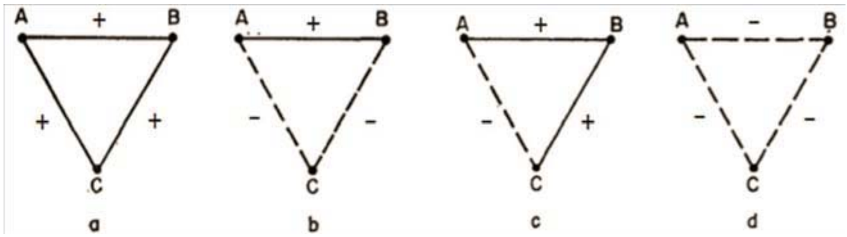


Fig.2: Four s-graphs of three points and three lines each. Structures *a* and *b* are balanced, but structures *c* and *d* are not.

We can see that if the product of signed edges in a triad is positive the triad is balanced; otherwise it is unbalanced. For example, in the figure structure b is balanced since $AB \cdot BC \cdot CA > 0$.

1) Preliminary Exploration

a. Identify the top five key players based on unsigned degree centrality.

b. How many “friendly” political communities exist? (Hint: change the edges to denote degree of friendship (-1-> 0, 0 ->1, 1 - >2) and run community detection using “resolution = 1.0” in Gephi).

c. Visualize the political network such that **red ties** denote negative relationships and **blue ties** denote positive relationships. Color the nodes in terms of the community to which they belong. Be sure to include the labels of the political entities in your visualization.

d. To characterize the tension within this political network, we define the Degree of Conflict (DoC) as:

$$DoC = \frac{(\#\ of\ negative\ ties)}{(\#\ of\ negative\ ties) + (\#\ of\ positive\ ties)} \times 100\%$$

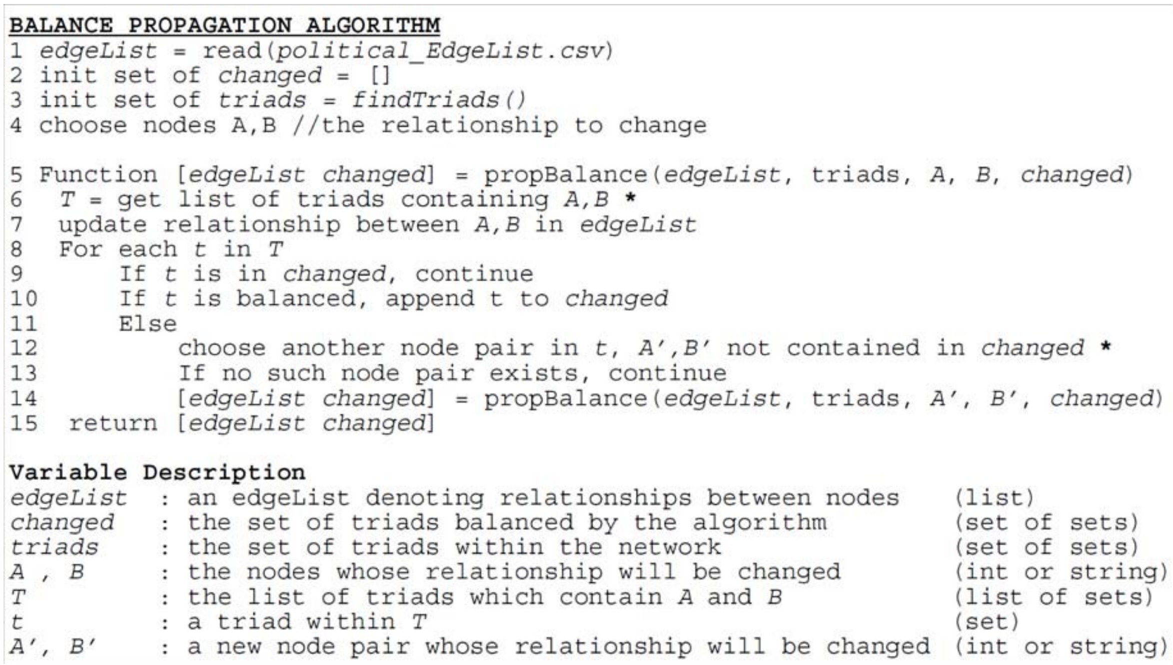
What is the DoC within this network?

2) Structure imbalance

a. According to the definition of balance, how many triads within the political network are unbalanced? (Hint: you may use the Matlab script *findTriads.m* provided).

b. Now, consider your task as a United Nations diplomat. Your goal is to minimize the degree of conflict within this network. Assume that any changes made to the network propagate according to Structural Balance theory (the algorithm provided below). How would the network change if the diplomat changed the relationship between Israel and Islamic State? Visualize your results as in part 1c, report the DoC, and the number of unbalanced triads. By how much did the DoC change? By how much did the number of unbalanced triads change?

To answer this question, you must implement the following recursive algorithm to propagate any change made to the network:



```
BALANCE PROPAGATION ALGORITHM
1 edgeList = read(political_EdgeList.csv)
2 init set of changed = []
3 init set of triads = findTriads()
4 choose nodes A,B //the relationship to change

5 Function [edgeList changed] = propBalance(edgeList, triads, A, B, changed)
6   T = get list of triads containing A,B *
7   update relationship between A,B in edgeList
8   For each t in T
9       If t is in changed, continue
10      If t is balanced, append t to changed
11      Else
12          choose another node pair in t, A',B' not contained in changed *
13          If no such node pair exists, continue
14          [edgeList changed] = propBalance(edgeList, triads, A', B', changed)
15   return [edgeList changed]

Variable Description
edgeList   : an edgeList denoting relationships between nodes    (list)
changed    : the set of triads balanced by the algorithm         (set of sets)
triads     : the set of triads within the network                (set of sets)
A , B      : the nodes whose relationship will be changed        (int or string)
T          : the list of triads which contain A and B            (list of sets)
t          : a triad within T                                    (set)
A', B'     : a new node pair whose relationship will be changed  (int or string)
```

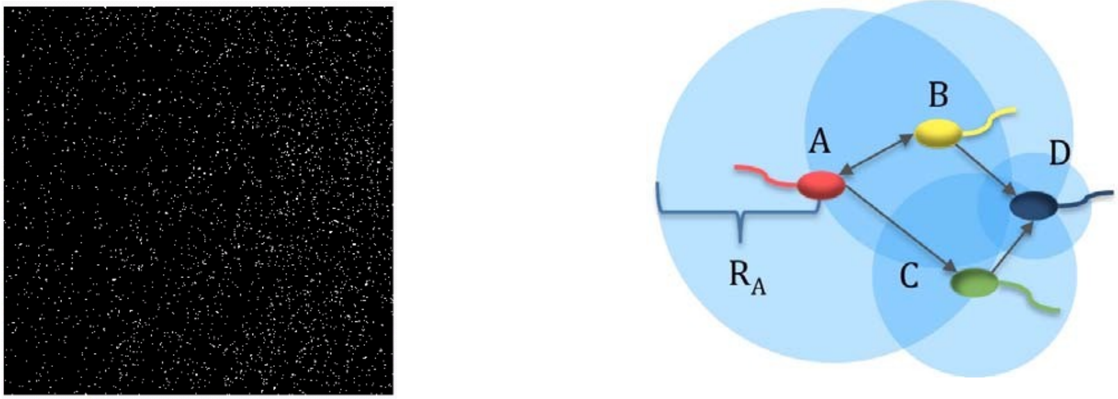
Note: Organize the list of triads in *T* (line 6) by increasing order with respect to the *nodeID* of *C*, the third node of each triad. When choosing another node pair *A*', *B*' (line 12), give highest preference to pairs with negative ties and secondary preference to pairs with the lowest sum of node IDs. The *nodeIDs* are the indices of each node in a list organized alphabetically.

c. Using your implementation of the balance propagation algorithm, try to reduce the DoC by 20% by finding a single relationship to change from the original network. Present your findings similarly to part 2*b* above. Submit your solution network in a csv file in your submission folder.

Problem 3 (25 pts)

As discussed in class, some bacteria use quorum sensing (QS) to communicate with each other. In this problem, we are going to construct a bacteria communication network based on QS. Once the bacteria reach the QS activation threshold, they express proteins (they appear as white dots in Fig. 2(a)). The amount of protein expressed by bacteria is positively correlated with the concentration of *QS signaling molecules*.

These QS signaling molecules diffuse into the environment and spread from the source bacterium. Other bacteria can then sense the concentration of QS signaling molecules in its immediate surroundings and take action if necessary. By using these properties, we define the influence range of a particular bacterium as the longest distance at which other bacteria can still sense the QS signals originating from the bacterium. For example, as shown in Fig. 2(b), the influence range ($\mathbf{R_A}$) of bacterium ***A*** is the maximum distance at which any other bacterium (e.g., bacterium ***C***) can sense **A**'s QS molecules. Therefore, we create a directed edge $\boldsymbol{E_{AC}}$ (***A*** to ***C***). By iterating this procedure for every bacterium, we can create a bacteria communication (influence) network.



(a) **(b)**

Fig. 2. (a) White dots represent the proteins expressed by bacteria during QS. You are given this data as a matrix where each entry represents the protein intensity at that location. (b) Bacteria communication model.

Data: You are given a file containing the (spatial) distribution of protein intensities as a 512-by-512 matrix; an entry in this matrix (I_{ij}) is greater than zero, if there exist a bacterium at that particular location and the bacterium generates proteins during QS signaling. Of note, for each entry *(i,j)* in the *I* matrix (I_{ij}), *i* and *j* correspond to the (spatial) *x* and *y* coordinates, respectively.

Task 1: *Network Construction (15pts)*
(i) Compute the influence range
Given the matrix (**I**), we can compute the entries of the concentration of signaling molecules ($\mathbf{S_{ij}}$) as follows:

$$\mathbf{S_{ij}} = \mathbf{Q}\ (\mathbf{I_{ij}}/\max(\mathbf{I}))$$

where **Q** is a normalization constant and the max function returns the largest value within the intensity matrix (**I**).
The mapping from the concentration matrix (**S**) to the influence range (**R**) is given by[1]:

$$\mathbf{R_{ij} = \log(S_{ij})/D}$$

[1] We compute influence range only for S_{ij}> 0 values.

where **D** is the diffusion coefficient of the QS signaling molecules. In this step, we use **Q = 10** and **D = 0.2** to find the influence range.

(ii) Form the network edges
Consider the scenario in Fig. 2(b). Bacteria B and C are within the influence range of bacterium A; this results in directed edges from A to both B and C. Similarly, bacteria A and D are within the influence range of bacterium B; this results in directed edges from B to both A and D. In contrast, since the influence range of bacterium D is too short, D does not influence any other bacteria and so no edges are drawn from bacterium D.
In this step, using the influence ranges calculated in Task 1(i) above, you should create (directed) edges from each bacterium to all bacteria that lie within its influence range. For example, if $R_{ij} = 3$, you should create a directed edge from entry *(i, j)* to *(x, y)*, for all x and y that satisfy $|x - i| < 3$, $|y - j| < 3$, and $I_{xy} > 0$, i.e., bacteria exist at location (x,y).

(iii) Visualize the bacteria network
Consider a particular file type .vna in Gephi - see the link below:
https://gephi.org/users/supported-graph-formats/netdraw-vna-format/
and the sample file provided on Canvas that assigns each node with *x* and *y* coordinates. Output the .vna file and import in Gephi to visualize the network.

Task 2: *Network Analyses Using Gephi (15pts)*
(i) Plot the degree distribution of your bacteria network. Report the number of nodes, edges, average degree, and average path length.
(ii) Compute the number of communities using resolution = 1.0. How many unique communities do you get?
(iii) Color the network based on discovered communities and change the size of each node based on its degree. Show the snapshot of the bacteria network you obtain.

Note on submission:

***** For all problems in this homework, everything is handled electronically. Prepare your answers using either Word or Latex and ***create a single PDF file for submission*** *(i.e., this PDF file should contain the answers to all the problems in the HW).* Also, put the source code and all related files for each problem in a *separate folder*.
*****
Finally, compress *everything* and name it as "yourAndrewId_hw3.zip", and deposit it on Canvas under Assignments > HW3 Submission.

***** Work *individually* on all problems.

Good luck!