CS550: Massive Data Mining and Learning
Homework 3

Due 11:59pm Saturday, April 18, 2020
Only one late period is allowed for this homework
Submitted by:
Name: Prakruti Joshi
NetID: phj15

Submission Instructions

Assignment Submission Include a signed agreement to the Honor Code with this assignment. Assignments are due at 11:59pm. All students must submit their homework via Sakai. Students can typeset or scan their homework. Students also need to include their code in the final submission zip file. Put all the code for a single question into a single file.

Late Day Policy Each student will have a total of *two* free late days, and for each homework only one late day can be used. If a late day is used, the due date is 11:59pm on the next day.

Honor Code Students may discuss and work on homework problems in groups. This is encouraged. However, each student must write down their solutions independently to show they understand the solution well enough in order to reconstruct it by themselves. Students should clearly mention the names of all the other students who were part of their discussion group. Using code or solutions obtained from the web is considered an honor code violation. We check all the submissions for plagiarism. We take the honor code seriously and expect students to do the same.

Discussion Group (People with whom you discussed ideas used in your answers):

On-line or hardcopy documents used as part of your answers:

I acknowledge and accept the Honor Code.

(Signed) Prakruti Joshi

If you are not printing this document out, please type your initials above.

Answer to Question 1(a)

Modularity of the original graph G:
Adjacency Matrix of Graph G:
A =

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Degree distribution:
k = [4, 3, 3, 3, 2, 2, 4, 1]
Number of nodes (m) = 11
S (community label vector) = [1, 1, 1, 1, -1, -1, -1, -1]
Now, modularity is defined as:

$$Q = \frac{1}{4m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) s_i s_j$$

Using the values of A, k, m and s, the modularity (Q) of the network comes out to be **0.39256**

$$\boxed{\therefore Q = 0.393}$$

Now, Partitioning the graph by removing edge (A-G), we calculate the modularity Q as follows.
Adjacency Matrix:
A =

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Degree distribution:
k = [3, 3, 3, 3, 2, 2, 3, 1]

Number of nodes (m) = 10
S (community label vector) = [1, 1, 1, 1, -1, -1, -1, -1]
Now, modularity is defined as:

$$Q = \frac{1}{4m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) s_i s_j$$

Using the values of A, k, m and s, the modularity (Q) of the network comes out to be **0.48**

$$\boxed{\therefore Q = 0.48}$$

Answer to Question 1(b)

The modularity (Q) of the original graph is **0.393**
Partitioning the original graph and retaining communities as per Q1-(a) and adding edge (E-H), we calculate the modularity (Q) as follows.
Adjacency Matrix:
A =

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Degree distribution: k = [4, 3, 3, 3, 3, 2, 4, 2]
Number of nodes (m) = 12
S (community label vector) = [1, 1, 1, 1, -1, -1, -1, -1]
Now, modularity is defined as:

$$Q = \frac{1}{4m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) s_i s_j$$

Using the values of A, k, m and s, the modularity (Q) of the community detection comes out to be **0.41319**

$$\boxed{\therefore Q = 0.413}$$

The modularity **went up** as compared to Q1-(a). This is because adding an edge to the original graph which is inside one of the community increases the intra-community connectivity and results into better overall community structure. Since the nodes E and H belong

to the same community, s_i, s_j value will be same and the product will be 1. Thus, it results to addition in the calculation of Q and thus, modularity increases.

Answer to Question 1(c)

The modularity (Q) of the original graph is **0.393**
Partitioning the original graph and retaining communities as per Q1-(a) and adding edge (A-F) and removing edge (E-H) from Q1-(b) part, we calculate the modularity Q as follows.
Adjacency Matrix:
A =

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Degree distribution: k = [5, 3, 3, 3, 2, 3, 4, 1]
Number of nodes (m) = 12
S (community label vector) = [1, 1, 1, 1, -1, -1, -1, -1]
Now, modularity is defined as:

$$Q = \frac{1}{4m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) s_i s_j$$

Using the values of A, k, m and s, the modularity (Q) of the community detection comes out to be **0.31944**

$$\boxed{\therefore Q = 0.319}$$

The modularity **went down** as compared to Q1-(a). This is because adding an edge to the original graph which crosses the two communities increases the inter-community connectivity. In partitioning the graph into two communities, we aim to minimize inter-cluster edges. Thus, increasing inter-community connectivity leads to decrease in the modularity of the network. Since the nodes A and F belong to the different community, s_i, s_j value will be different and the product will be -1. Thus, it results to decrease in the value of Q.

Answer to Question 2(a)

There are 8 nodes in graph G.

Adjacency Matrix: (8x8 symmetric matrix)

A =

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Degree Matrix: (8x8 diagonal matrix)

$$D = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Laplacian matrix:

$$L = D - A$$

$$L = \begin{bmatrix} 4 & -1 & -1 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Answer to Question 2(b)

For the Laplacian matrix L computed in 2(a), the eigenvalues and the corresponding eigenvectors are as follows (sorted in ascending order):

- 2.1403818880962127e-16, $\begin{bmatrix} -0.35355339 \\ -0.35355339 \\ -0.35355339 \\ -0.35355339 \\ -0.35355339 \\ -0.35355339 \\ -0.35355339 \\ -0.35355339 \end{bmatrix}$
- 0.3542486889354087, $\begin{bmatrix} -0.24701774 \\ -0.38252766 \\ -0.38252766 \\ -0.38252766 \\ 0.38252766 \\ 0.38252766 \\ 0.24701774 \\ 0.38252766 \end{bmatrix}$
- 1.0000000000000049, $\begin{bmatrix} 0.00000000e+00 \\ -3.18493382e-17 \\ 8.59836280e-17 \\ -5.79022479e-17 \\ -4.08248290e-01 \\ -4.08248290e-01 \\ 3.76795815e-18 \\ 8.16496581e-01 \end{bmatrix}$
- 3.0000000000000036, $\begin{bmatrix} 0.00000000e+00 \\ -2.70599246e-17 \\ -1.12776339e-16 \\ 1.50488323e-16 \\ 7.07106781e-01 \\ -7.07106781e-01 \\ -1.06520593e-17 \\ 1.12242936e-16 \end{bmatrix}$

- 3.999999999999996, $\begin{bmatrix} 0.60717154 \\ -0.27939608 \\ -0.1005666 \\ -0.22720886 \\ -0.20239051 \\ -0.20239051 \\ 0.60717154 \\ -0.20239051 \end{bmatrix}$
- 4.000000000000001, $\begin{bmatrix} 0.00000000e+00 \\ 5.62206567e-01 \\ 2.31676233e-01 \\ -7.93882800e-01 \\ 2.16840434e-16 \\ -2.82759927e-16 \\ -1.11022302e-16 \\ -1.71737624e-16 \end{bmatrix}$
- 4.000000000000002, $\begin{bmatrix} -0.07964119 \\ -0.56053094 \\ 0.80283611 \\ -0.16266398 \\ 0.02654706 \\ 0.02654706 \\ -0.07964119 \\ 0.02654706 \end{bmatrix}$
- 5.645751311064582, $\begin{bmatrix} 0.66255735 \\ -0.14261576 \\ -0.14261576 \\ -0.14261576 \\ 0.14261576 \\ 0.14261576 \\ -0.66255735 \\ 0.14261576 \end{bmatrix}$

Answer to Question 2(c)

1. Second smallest eigenvalue $\lambda_2 = 0.35424868893540984$
   Approximately, $\lambda_2 = 0.3542$
2. Eigen vector corresponding to $\lambda_2 =$
   $[-0.24701774 \quad -0.38252766 \quad -0.38252766 \quad -0.38252766 \quad 0.38252766 \quad 0.38252766$
   $0.24701774 \quad 0.38252766]$

3. Partioning the graph with 0 as the boundary:

   Community 1: Negative points

Node IDs	Node	Eigen vector value
1	A	-0.24701774
2	B	-0.38252766
3	C	-0.38252766
4	D	-0.38252766

   Community 2: Positive points

Node IDs	Node	Eigen vector value
5	E	0.38252766
6	F	0.38252766
7	G	0.24701774
8	H	0.38252766

Answer to Question 3(a)

Given: C_i is a set of nodes of G that are divisible by i.
To prove: For any integer i greater than 1, C_i is a clique
Solution:
Any two nodes in C_i will have atleast 1 factor common : i. This is because C_i consists of all the nodes that are divisible by i. Thus, any two nodes belonging to C_i will have an edge between them. Thus, for any integer $i > 1$, C_i is a clique.

Answer to Question 3(b)

Solution: C_i is a maximal clique for every prime integer $i < 1000000$.

Explanation:
As stated in the question, a clique C is maximal when every node not in C is missing an edge to atleast one member of C. Now, consider two cases:

1. Case 1: i is not a prime number
   If i is not a prime number, consider an integer j which is a factor of i such that $1 < j < i$. Since j is not divisible by i, it will not be in C_i. But node j will have an edge with every member of C_i because j is a common factor. Thus, none of the node is missing an edge. Hence, C_i will not be maximal.

2. Case 2: i is a prime number
   Since i is prime, there are no nodes which are not in C_i. All the nodes in C_i has an edge with i itself. For example, let j be such a node. j can not be a factor of i because i is a prime number. But since there is an edge between i and j, j must be a multiple of i. Therefore, node j should already be in C_i. Hence C_i is maximal.

Also, if $i > 1000000$, C_i is an empty clique. Thus, C_i is a maximal clique for every prime integer $i < 1000000$.

Answer to Question 3(c)

As proved in Q3-(b), C_i can be maximal only when i is a prime integer. From all the prime numbers, $i = 2$ has the maximum multiples (even numbers constitute half of the numbers) and thus C_2 has maximum elements in the set as compared to other maximal cliques. Therefore, C_2 is the largest maximal clique possible and is thus a unique maximal clique.