CS550: Massive Data Mining and Learning
Homework 3

Due 11:59pm Monday, Apr 10, 2023

Submission Instructions

Assignment Submission Include a signed agreement to the Honor Code with this assignment. Assignments are due at 11:59pm. Students should submit their homework via Canvas. 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. Finally, put your PDF answer file and all the code files in a folder named as your Name and NetID (i.e., Firstname-Lastname-NetID.pdf), compress the folder as a zip file (e.g., Firstname-Lastname-NetID.zip), and submit the zip file via Canvas.

Late Policy: The homework is due on 4/10 (Monday) at 11:59pm. We will release the solutions of the homework on Canvas on 4/14 (Friday) 11:59pm. If your homework is submitted to Canvas before 4/10 11:59pm, there will no late penalty. If you submit to Canvas after 4/10 11:59pm and before 4/14 11:59pm, your score will be penalized by 0.9^k, where k is the number of days of late submission. For example, if you submitted on 4/13, and your original score is 80, then your final score will be $80 \times 0.9^3 = 58.32$ for $13 - 10 = 3$ days of late submission. If you submit to Canvas after 4/14 11:59pm, then you will earn no score for the homework.

Honor Code Students may discuss homework problems with peers. 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 with whom they discussed the homework. Directly using code or solutions obtained from others or 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): Abhishek Nayak

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

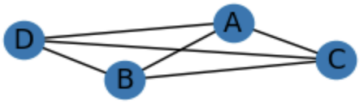
I acknowledge and accept the Honor Code.

*(Signed) Vikram Sahai Saxena*_____

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

Answer to Question 1(a)

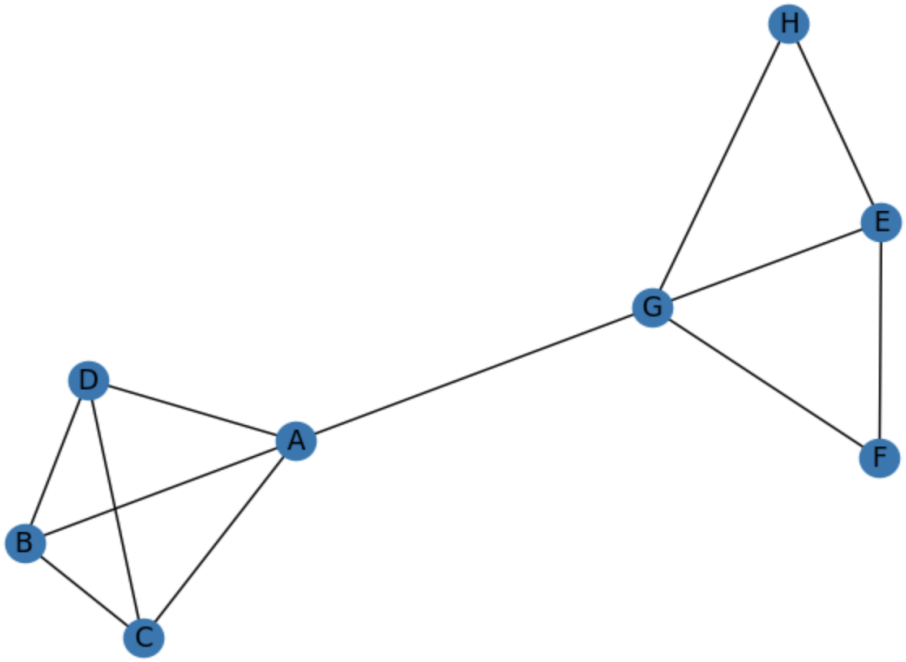
On removing edge (A, G), we partition the graph into the following two communities:



The modularity of the above partition = 0.48

Answer to Question 1(b)

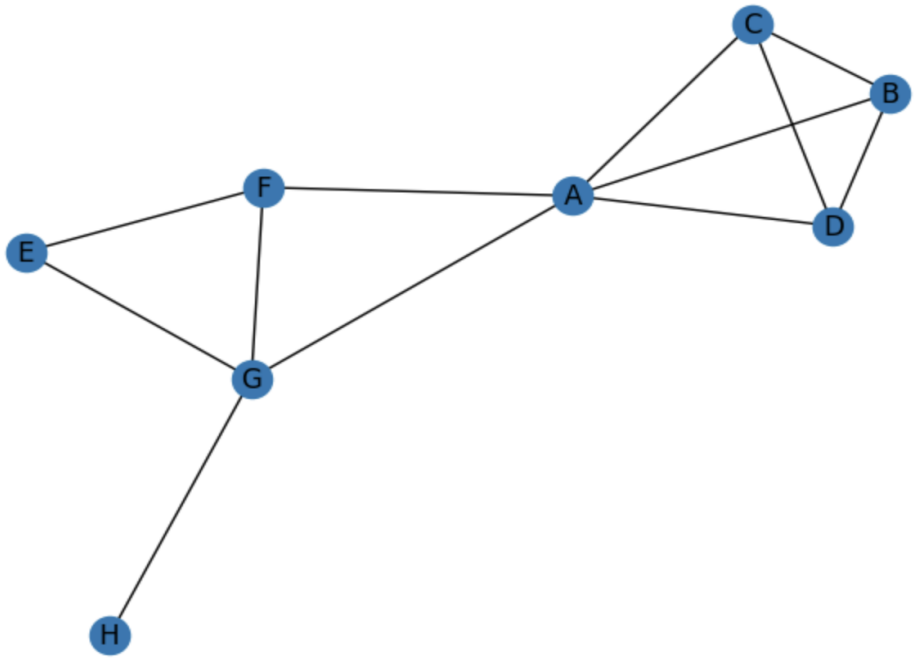
Modularity of original graph = 0.392562
On adding a link between nodes E and H, we get:



Modularity of above graph = 0.413194
After adding the link between nodes E and H, the modularity went up. This is because the nodes E and H belong to the same group partition which was identified in part (a) and adding an edge in the same partition makes it stronger.

Answer to Question 1(c)

Modularity of original graph = 0.392562
On adding a link between nodes F and A, we get:



Modularity of above graph = 0.319444
After adding the link between nodes F and A, the modularity went down. This is because the nodes F and A belong to different group partitions and adding an edge between different partitions makes it weaker.

Answer to Question 2(a)

A, D and L:

```
Adjacency matrix A:
 [[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]]

Degree matrix D:
 [('A', 4), ('B', 3), ('C', 3), ('D', 3), ('E', 2), ('F', 2), ('G', 4), ('H', 1)]

Laplacian matrix L:
 [[ 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]]
```

Answer to Question 2(b)

Eigen values and vectors of L:

```
Eigen values of the Laplacian matrix:
 [ 5.64575131e+00  4.00000000e+00 -1.43234382e-16  3.54248689e-01
  1.00000000e+00  3.00000000e+00  4.00000000e+00  4.00000000e+00]

Eigen vectors of the Laplacian matrix:
 [[ 6.62557346e-01 -6.12372436e-01  3.53553391e-01  2.47017739e-01
  -5.35259428e-17  6.06092159e-18 -1.38130079e-01  4.10538607e-02]
 [-1.42615758e-01  2.04124145e-01  3.53553391e-01  3.82527662e-01
  -1.64148370e-16  3.33214157e-17  5.11283241e-01 -7.58666091e-01]
 [-1.42615758e-01  2.04124145e-01  3.53553391e-01  3.82527662e-01
  -5.44146987e-17  2.56779553e-16 -7.45346657e-01  7.33157817e-02]
 [-1.42615758e-01  2.04124145e-01  3.53553391e-01  3.82527662e-01
  -1.51522333e-17 -2.89842000e-16  3.72193496e-01  6.44296448e-01]
 [ 1.42615758e-01  2.04124145e-01  3.53553391e-01 -3.82527662e-01
  -4.08248290e-01 -7.07106781e-01  4.60433598e-02 -1.36846202e-02]
 [ 1.42615758e-01  2.04124145e-01  3.53553391e-01 -3.82527662e-01
  -4.08248290e-01  7.07106781e-01  4.60433598e-02 -1.36846202e-02]
 [-6.62557346e-01 -6.12372436e-01  3.53553391e-01 -2.47017739e-01
   1.75084671e-16  3.04138855e-17 -1.38130079e-01  4.10538607e-02]
 [ 1.42615758e-01  2.04124145e-01  3.53553391e-01 -3.82527662e-01
   8.16496581e-01 -1.92579648e-16  4.60433598e-02 -1.36846202e-02]]
```

Answer to Question 2(c)

Eigen vector corresponding to the second smallest eigen values:

```
Eigen vector corresponding to the second smallest eigen values:
 [ 0.24701774  0.38252766  0.38252766  0.38252766 -0.38252766 -0.38252766
 -0.24701774 -0.38252766]
```

If we use 0 as the boundary to partition the graph into two communities, the first community contains the nodes that have negative eigenvector value and the second community contains the nodes that have a positive eigenvector value. So, we get the following partitioning result:

Partition 1: A, B, C, D
Partition 2: E, F, G, H

Answer to Question 3(a)

If i is any integer greater than 1, then the set C_i of nodes of G that are divisible by i is a clique, because node i will have an edge with every node in the set Ci as all the nodes in the set are divisible by i. Also, all the other nodes in the set C_i will have an edge with other nodes in Ci because they all have a common factor i. Hence proved.

Answer to Question 3(b)

For C_i to be a maximal clique, i must be a prime number ≤ 1000000.

Proof:

- Case 1: If $i <= 1000000$ and is not prime. Let k be a factor of i and $1 < k < i$. Node k is not in C_i, however it has an edge to every member of C_i, because it has k as a common factor. So, Ci is not maximal.
- Case 2: If $i >= 1000000$. C_i is an empty clique and adding one node will make it a 1-clique, which is not maximal.
- Case 3: If $i <= 1000000$ and is prime. There is no node outside C_i that has an edge to the node i itself. Suppose we have an outside node k, then i and k have a common factor other than 1, which can only be i, since i is prime. Now, k becomes a multiple of i since k has i as a factor and so k is already in C_i. Thus, making it maximal.

Hence proved.

Answer to Question 3(c)

From part 3(b), we showed that C_i can be maximal only when i is a prime number. We know that out of all the prime numbers, $i = 2$ has the maximum multiples in the set because it is a factor for all the even numbers greater than 2. So, C2 will have maximum elements. Hence, C2 is larger than any other clique and is thus a unique maximal clique.