

Social Network Analysis Project

Luigi Maiese – 0622900080 – l.maiese1@studenti.unisa.it

Mario Cerino – 0622701191 – m.cerino8@studenti.unisa.it

Alberto Vitiello – 0622701186 – a.vitiello62@studenti.unisa.it

Arcangelo Palma – 0622701208 – a.palma27@studenti.unisa.it

Contents

[Midterm Project 3](#_Toc76911901)

[Exercise 1 3](#_Toc76911902)

[Task 3](#_Toc76911903)

[Solution 3](#_Toc76911904)

[Exercise 2 3](#_Toc76911905)

[Task 3](#_Toc76911906)

[Solution 4](#_Toc76911907)

[Performance Analysis 4](#_Toc76911908)

[Exercise 3 5](#_Toc76911909)

[Task 5](#_Toc76911910)

[Solution 5](#_Toc76911911)

[Exercise 4 5](#_Toc76911912)

[Task 5](#_Toc76911913)

[Solution 6](#_Toc76911914)

[Final Project 7](#_Toc76911915)

[Exercise 1 7](#_Toc76911916)

[Task 7](#_Toc76911917)

[Solution 7](#_Toc76911918)

[Exercise 2 8](#_Toc76911919)

[Task 8](#_Toc76911920)

[Solution 8](#_Toc76911921)

[Exercise 3 8](#_Toc76911922)

[Task 8](#_Toc76911923)

[Solution 10](#_Toc76911924)

# Midterm Project

## Exercise 1

### Task

Consider the “Facebook Large Page-Page Network” dataset available at the Stanford Large Network Dataset Collection (SNAP) (https://snap.stanford.edu/data/facebook-large-page-page-network.html). Note that network edges are provided in the file musae\_facebook\_edges.csv contained in the zip file linked in this page. Analysing this file, you can see that nodes of the network can be partitioned in four categories: politicians, governmental organizations, television, shows and companies. We will refer to this partition as the real clustering. You are required to cluster nodes of the network in at least 4 clusters using each of the partition algorithms seen in class: hierarchical, k-means, Girman-Newmann (betweenness-based clustering), spectral. Note that the network is very large and the naive implementations of these algorithms may be very expensive. Hence, you are required to optimize these algorithms (by sampling, parallelism, and adhoc optimizations) to make their running times feasible. Compare the clustering obtained through each of your algorithms with respect to the real clustering given in the file musae\_facebook\_target.csv. Discuss the trade-off between precision and running time of each of your proposed implementations.

### Solution

For exercise 1 the network has been clustered using the following methods:

1. Hierarchical:
   1. Naïve
   2. Custom ad-hoc optimization
2. K-Means
   1. Naïve
   2. Custom ad-hoc optimization
3. Betweenness:
   1. Naïve
   2. Parallel
4. Spectral:
   1. Naïve
   2. Parallel

### Performance Analysis

Performance Analysis

## Exercise 2

### Task

Return the top 500 nodes (that are approximatively the top 2%) of the Facebook Large Page-Page Network according to each of the following centrality measures: degree, closeness, betweenness, PageRank, HITS. For the first three measures you can use the algorithms presented in class. For PageRank and HITS algorithms you have to provide both a naive and a parallel implementation. As in the previous exercise, you have to optimize your algorithms to make their running times feasible on a very large network. For each of the required centrality measures describe which is its best implementation, by taking into account both the running time and the precision. In particular, for the measures involving the choice of some parameters, such as PageRank and HITS, discuss of the best choice of parameters. Compare the results of the different algorithms and discuss about the similarities and the differences among the returned outcomes.

### Solution

For exercise 2 the network has been analysed using the following centrality measures:

1. Degree:
   1. Naïve
2. Closeness
   1. Naïve
   2. Parallel
3. Betweenness:
   1. Naïve
   2. Parallel
4. PageRank:
   1. Naïve
   2. Vectorized
   3. Networkx
5. HITS:
   1. Naïve
   2. Parallel

### Performance Analysis

The heuristic used for the evaluation of each centrality measure follows these rules:

1. The ground truth of each centrality measure is represented by the result of the naïve version.
2. Each optimization is evaluated with respect to:
   1. Running time.
   2. Precision compared to the ground truth.

For this purpose, we chose a rule that enabled us to make this decision, in particular we analysed the results following two parameters:

* **Similarity Rate**: this parameter represents the percentage of the nodes in the top 500 of the ground truth that are also in the top 500 of the optimized version
* **Equality Rate**: this parameter represents the percentage of the nodes in the top 500 of the ground truth that are in the same position respect to the top 500 of the optimized version

## Exercise 3

### Task

Consider the following scenario. A restaurant is evaluated by a reviewer with respect to three features: Food, Service, and Value. For each of these features the reviewer can assign from 0 to 5 points. Observe that not all the restaurants can be evaluated with respect to all the features. Indeed, even if all restaurants must be always evaluated on food. a restaurant that only offers take-away service cannot be evaluated about service. Similarly, a restaurant that only runs as company canteen cannot be evaluated about value. The Michelin guide must assign a score to each restaurant. Three scores are possible: one star, two stars, three stars. This year, the Michelin guide’s principal has decided to not use his expensive team of experts to evaluate the restaurants, but to run an algorithm that, taken in input the scoring assigned by the experts in the past, and some new reviews, assigns the scores. However, the algorithm must avoid that a restaurant “with service” will receive a higher score than when it declares to be “only take-away”. Similarly, it must avoid that a restaurant “a la carte” will receives a higher score when it declares to be a “company canteen”. Provide a classifier for the Michelin guide’s principal that satisfies all the required features. You must convince the principal that your approach satisfies the required features, either by providing a formal proof, or by running massive experiments showing that the required features are (almost) always metre.

### Solution

## Exercise 4

### Task

Consider the same setting as in the previous exercise but suppose that the principal can accept to trade off precision and efficiency of the classifier with its robustness. Thus, it is accepted that misreporting occurs, but we would like that the cases in which a restaurant has an incentive to misreport its features are as few as possible, but he requires that the classifier be as precise and fat as possible. Provide a classifier to the principal that satisfies these requirements. Motivate your choice by comparing your choice with different alternatives and showing how your choice experimentally outperforms the other alternatives in terms of incentive-compatibility, precision, or performance.

### Solution

# Final Project

## 

## Exercise 1

Discussion about exercise 1

### Task

Implement the following game-theoretic centrality measures:

1. shapley\_degree:

this is the Shapley value for the characteristic function value(C) = |C| + |N(C)|, where N(C) is the set of nodes outside C with at least one neighbour in C;

1. shapley\_threshold(k):

this is the Shapley value for the characteristic function value(C) = |C| + |N(C, k)|, where N(C, k) is the set of nodes outside C with at least k neighbours in C;

1. shapley\_closeness:

this is the Shapley value for the characteristic function value(C) = Σu 1/dist(u, C), where dist(u, C) is the minimum distance between u and a node of C.

Recall that the naive implementation of Shapley value requires a running time that is exponential in the number of nodes of the network. You are instead required to provide a polynomial time algorithm for the above measures. On the e-learning platform you fill find material that will help you in designing and implementing these algorithms.

Implement also the Friedkin-Johnsen (FJ) dynamics, that works as follows:

* each node u has a private belief bu in [0, 1] and a stubbornness value su in [0,1];
* at each time step t each node publicizes an opinion xu(t) in [0,1] where:
  + xu(0) = bu, i.e., the initial opinion is exactly its belief;
  + xu(t) = su bu + (1-su) sumv in N(u) 1/N(u) xv(t-1), i.e., the opinion at time t is a weighted average of the private belief and of the opinion publicized by its neighbours at the previous step.

Does these dynamics converge to a stable state (i.e., a state in which no agent updates her opinion – you may assume a finite precision for opinion of at most 5 decimal digits)? Provide either a formal proof or experimental evidence for your answer.

### Solution

Proposed solution.

## Exercise 2

Discussion about exercise 2

### Task

Consider the network N represented in the file net\_x, that has been generated with one of the network models seen during the course.

You have to analyse the network N and guess which model has been used for creating it. Your guess has to be supported by an appropriate set of experiments to confirm that networks generated with the proposed model have characteristics similar to N (note that you have to guess also the parameters of the model).

During the discussion of the project, you will be asked to motivate your guess. Motivations may be related to both theoretical properties of the models seen during the course (e.g., “I analysed the provided network and I observed that its node degree distribution follows a power law. Hence, I conclude that it is not possible that the graph has been generated with a model random(n,p).”), and to experimental evidence (e.g., “I generated a lot of random graphs with p = 1/3, and none of them had similar properties as the provided network. Hence I conclude that it is improbable that the graph is random(n, 1/3)”).

A bonus point will be assigned to all the components of the groups that correctly guessed the model (and parameters) used to generate N.

### Solution

Proposed solution.

## Exercise 3

Discussion about exercise 3

### Task

Suppose there is an election and the voters are connected through a social network. G = (V, E). Suppose that there are n voters, represented by the nodes of the graph G, and m candidates. Each candidate i has a position pi in [0,1] that represents her political tendency (for example, a candidate whose position is close to 0 or 1 is, respectively, an extreme-left or an extreme right candidate, while a candidate with position close to 1/2 is moderate).

Each voter u has single-peaked preferences with peak in bu, i.e., she ranks candidates according to the distance of their positions from bu, by breaking possible ties in favour of the candidate on the left of bu (thus, the most preferred candidate is the one whose position is closest to bu, the second most preferred candidate is the one with the second closest position and so on).

The election occurs according to a plurality voting rule (see lesson about voting for a definition). We call an election truthful if each voter u votes for the candidate closest to her peak bu.

On the other hand, a voter can be influenced by opinion campaigns run over the social network and she could be induced to vote a candidate different from her favourite one. Specifically, we consider a manipulator that wants to improve the outcome of a given candidate c. To this aim the manipulator can select at most B voters (in the following called seeds), alter their peaks and use their influence to induce a change in the votes expressed by other voters.

We assume that the voting opinions diffuse over the network according to a FJ dynamic.

Specifically, if S is the set of at most B seeds chosen by the manipulator, then:

1. Set xu(0) = bu, and su=1/2 for every u not in S
2. For u in S, let bu = b’u, where b’u is defined by the manipulator, and set xu(0) = bu, and su=1
3. Run the FJ dynamics with this configuration
4. Once the dynamics reaches the equilibrium at time step t, update the preferences of voters by setting the peak pu = xu(t)
5. Re-run the election with voter’s peaks in pu. We call this election manipulated.

You have to design an algorithm that, given a network G, a set of m candidates with their positions (p1, …, pm), a special candidate c, a budget B, and the initial peaks of all the voters (b1, …, bn), returns a set S of at most B seeds and a peak value b’u for each seed u in S, such that the difference between the number of votes obtained by the candidate c in the manipulated election and the truthful one is maximized.

All the proposed manipulation algorithms will be tested on a common input. The group providing the larger increment in the number of votes for the candidate c will receive a bonus point.

INSTRUCTION FOR THE SUBMISSION:

Your code must include a function manipulation(G, p, c, B, b), where G is an undirected, unweighted graph, p is a Python list with each element in [0,1], c is in {0, …, len(p)-1}, B is a positive integer, and b is a Python list such that len(b) = len(G.nodes()) with each element in [0,1].

The function must print only one string that contains the following three elements separated by a comma:

* the group number;
* the number of votes for candidate c before the manipulation occurs;
* the number of votes for candidate c after the manipulation

### Solution

Proposed solution.