# AIS: Final Project Report

Jozef Hruska

February 1, 2023

## 1 Introduction

The technologies available to humanity in these modern times enabled people to have larger social networks than ever before. Large social networks mean a piece of information can potentially spread faster. This work aims to create a tool to simulate and analyze information diffusion between people.

The assignment of this final project is included in the first section 2. The following section 3 describes the social network model used in this project, along with all parameters that can be used to configure the simulation run(s). Section 4 discusses the complexity of the simulation algorithm. Section 5 proposes two different methods of identifying important nodes in the social network to help or block opinion diffusion. Lastly, section 6 shows four experiments on networks with different dimensions.

## 2 Assignment

**Project 2: Opinion Diffusion in Networks**

The purpose is to implement a systems to analyze opinion diffusion in a complex network.

In the model the nodes represent individuals that, are characterized by an opinion state, among a fixed number of choices. The opinions can be exchanged/transmitted to neighbor nodes.

At each time step the opinion diffusion is calculated with respect to system configuration parameters.

All nodes are initially in given opinion state, and can change. At each time step, for each node with opinion A, there is a probability `punchangeA` of not changing opinion, and a probability `1 - punchangeA` of adopting the opinion of one of its neighbors. If the individual is going to change opinion it will take with uniform probability the opinion of one of its neighbors (it holds a random tournament among opinions depending on the number of neighbors and their opinion).

Once an individual has adopt an opinion, it stays with that opinion without changing it for at least `tSTAY` time steps.

The nodes transitions are synchronous, i.e. they all take place at the end of the time step. The systems should allow to configure:

- the set of possible opinions
- `punchangeA` probability/resistance to change for each single opinion A
- `tSTAY` time steps that an individual persist in an newly adopted opinion

The system should allow:

- to load any graph in in csv format

- to configure the parameters, in particular the set of opinions and for each opinion the conservation parameters resistance to change
- to start and run one or more simulations with same initial node states
- to display statistics and metrics
- to display a graphical representation of the graph before, during or at the end of the simulation by coloring the nodes basing on their states, and to provide opinion distribution over a number of simulation on the same parameters and initial states.

**The candidate should develop a technique to find the most important nodes to block or help opinion diffusion.**

## 3 Social network model

A social network graph in this work is represented by a pair of `.csv` files, where:

- `graph.csv`: An edge-list graph representation, where each line is a pair of two node identifiers signifying an edge between these two nodes. Each node in a graph symbolizes a person in the social network graph.
- `state.csv`: A key-value list where each line is a part of a node identifier and an initial state of this node (initial opinion).

Opinions are represented as string values (i.e., `"A"`). Each individual node of the social network holds a state value representing its current opinion.

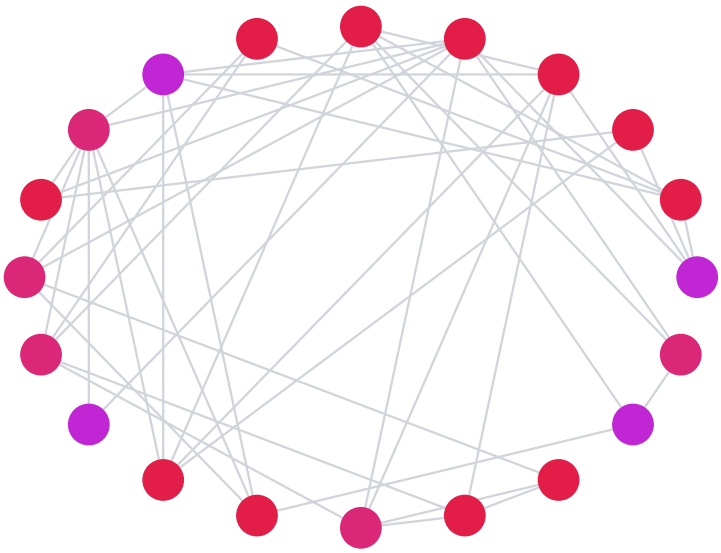

Figure 1: An example of a social network graph.

To simulate the opinion diffusion, each simulation holds an internal discrete "clock". This clock is incremented by one after each *round*. A round is over when all nodes decide whether to adopt an opinion of one of their neighbors or retain their current opinion. Each simulation run can be configured with the following parameters:

- `opinions`: A key-value record, where the key is an identifier of an opinion, and the value is the *probability this opinion will be retained* (unchanged) in the next round.

- `colors`: A list of HEX color codes used in graph visualization. Each opinion will use one of these colors in the exported graphs. The length of this list has to be equal to or greater than the number of available opinions.
- `runs`: An integer value specifying how many simulations run the program should run. Each simulation run will start with the same graph representation and the same initial node states.
- `freeze_length`: An integer value specifying the minimum number of rounds a node will keep a newly adopted state.
- `finish_time`: A number of rounds in one simulation run.

## 4 Algorithm and complexity

The algorithm controlling the flow of the simulation is a linear loop of `finish_time` rounds. In each round, every social network node randomly selects a neighboring node. Then, based on the retention probability of the neighboring node's opinion, it chooses whether to adopt the new opinion or not.

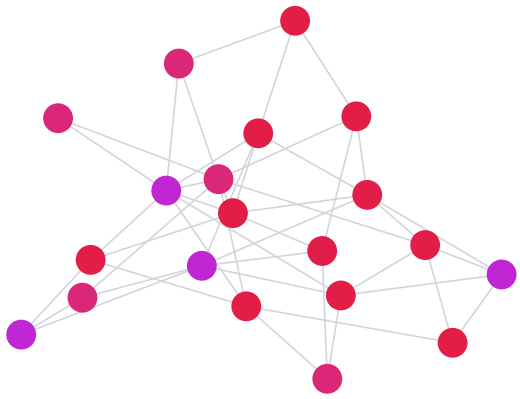

Figure 2: A social network with two opinions.

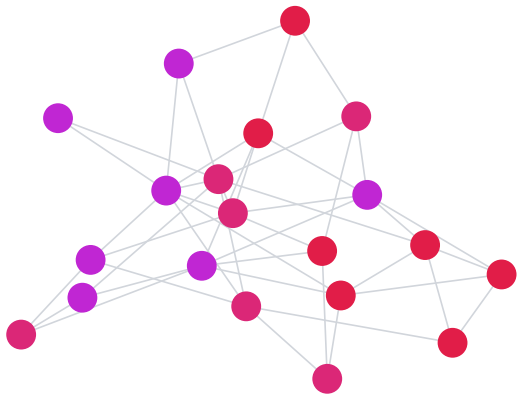

Figure 3: The same social network after 5 rounds.

All opinion changes are only committed (written) at the end of the round to simulate these events simultaneously. Because of this, the algorithm could be potentially run in parallel to improve performance.

## 5 Finding important nodes

As the assignment dictates, the neighbors for opinion adoption are selected by a random tournament. Because of this, each individual node of the network has exactly the same importance as other nodes.

One option is to find clusters of nodes with the same opinion. In these clusters, the outer nodes have higher importance than the inner nodes, as only the outer nodes can possibly change their opinion in the next round. Increasing or reducing the opinion adoption of these nodes could have an effect on opinion diffusion.

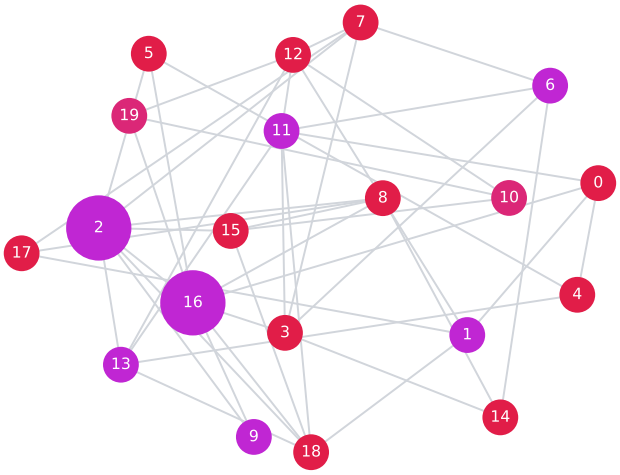


Figure 4: An example of a social network graph.

Another option is to identify a node (or a set of nodes) that interconnects clusters of nodes in the network. It is possible to use the *betweenness centrality* measure to identify these nodes as the value is higher for the nodes in question. The figure 4 shows a social network graph where the two nodes with the highest betweenness centrality (2, 16) are highlighted.

## 6 Experiments

Each experiment has an assigned number that corresponds to the dataset used in this experiment. For example, the data used for Experiment 01 can be found in the folder `data/01`. All datasets have been generated using the `gaussian_random_partition_graph` method. Parameters of the generation can be found in the `README.md` file included in each dataset.

### Experiment 01: 20 nodes, 3 opinions

The experiment features 20 nodes and 3 opinions ("A", "B", "C") with the following probabilities of opinion retention `(A=0.4, B=0.2, C=0.2)`.

The opinion freeze length (`freeze_length`) was set to 4 rounds. The number of simulation runs (`runs`) was set to 4, with each run finishing (`finish_time`) after 50 rounds.

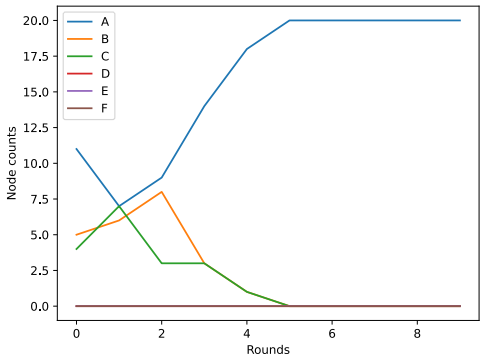


Figure 5: Node distribution of the first run.

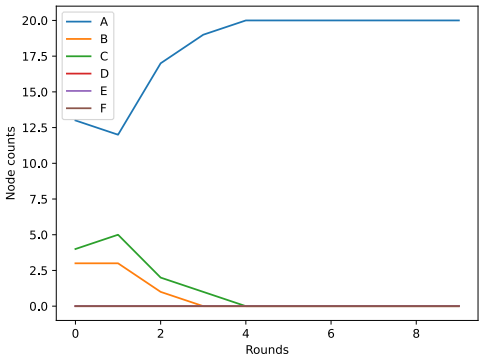


Figure 6: Node distribution of the second run.

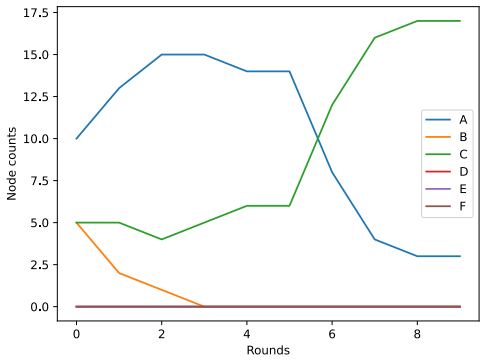


Figure 7: Node distribution of the third run.

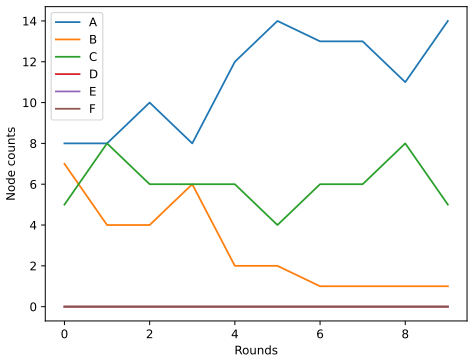


Figure 8: Node distribution of the fourth run.

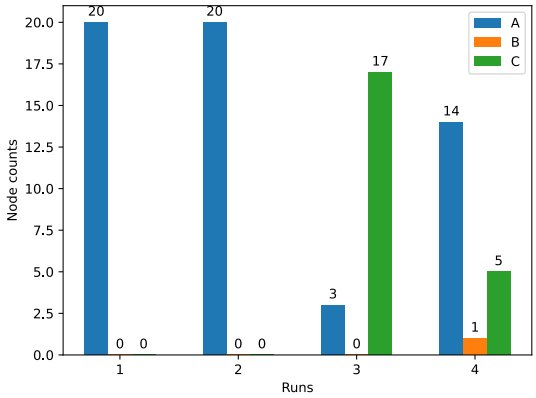


Figure 9: Node distribution in the final step of each run.

### Experiment 02: 50 nodes, 4 opinions

The experiment features 50 nodes and 4 opinions ("A", "B", "C", "D") with the following probabilities of opinion retention `(A=0.4, B=0.2, C=0.2, D=0.4)`.

The opinion freeze length (`freeze_length`) was set to 4 rounds. The number of simulation runs (`runs`) was set to 4, with each run finishing (`finish_time`) after 80 rounds.

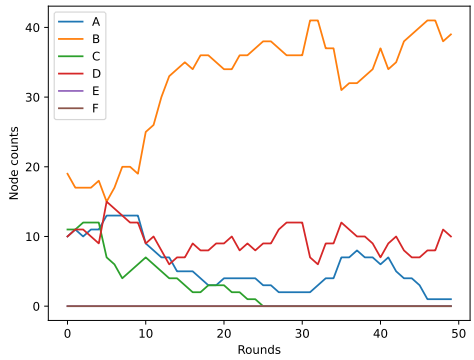


Figure 10: Node distribution of the first run.

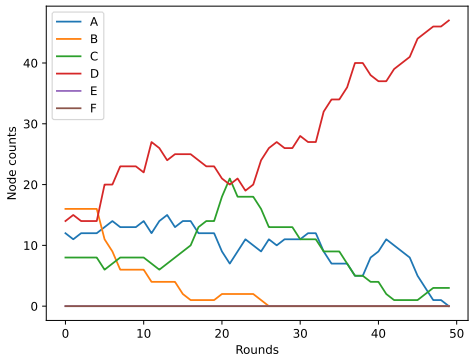


Figure 11: Node distribution of the second run.

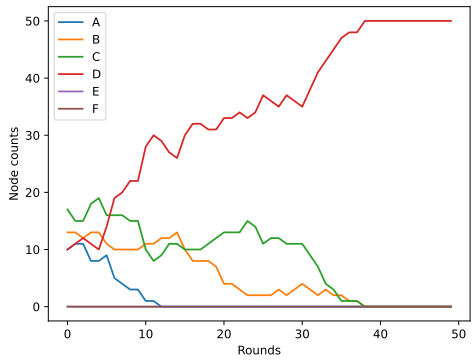


Figure 12: Node distribution of the third run.

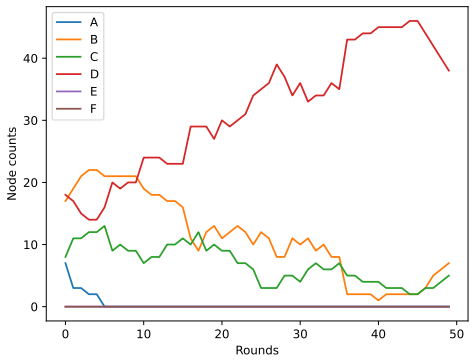


Figure 13: Node distribution of the fourth run.

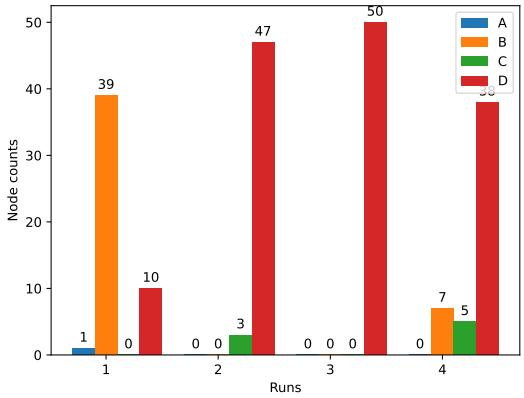


Figure 14: Node distribution in the final step of each run.

### Experiment 03: 200 nodes, 5 opinions

The experiment features 200 nodes and 5 opinions ("A", "B", "C", "D", "E") with the following probabilities of opinion retention `(A=0.4, B=0.2, C=0.2, D=0.4, E=0.6)`.

The opinion freeze length (`freeze_length`) was set to 6 rounds. The number of simulation runs (`runs`) was set to 4, with each run finishing (`finish_time`) after 200 rounds.

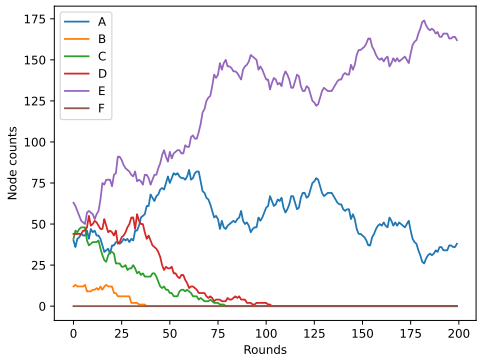


Figure 15: Node distribution of the first run.

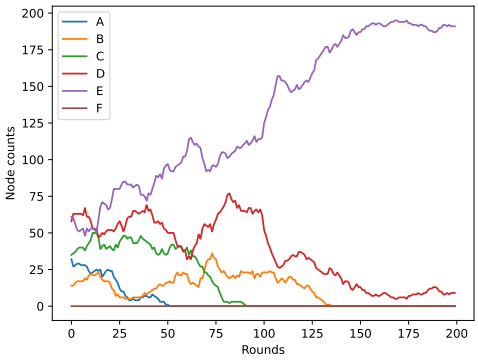


Figure 16: Node distribution of the second run.

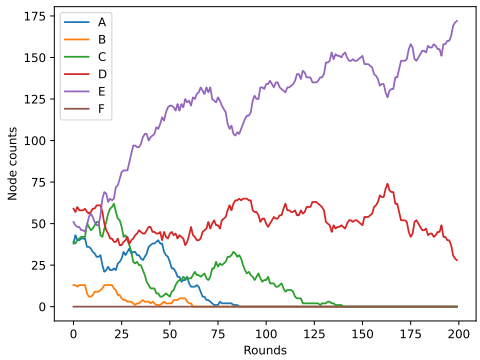


Figure 17: Node distribution of the third run.

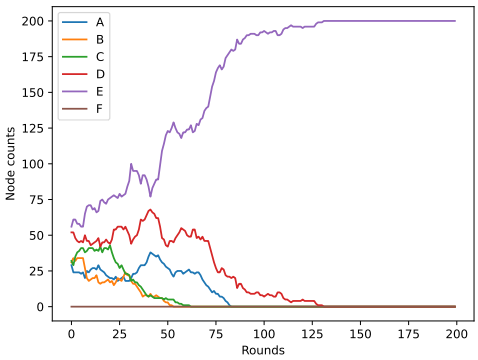


Figure 18: Node distribution of the fourth run.

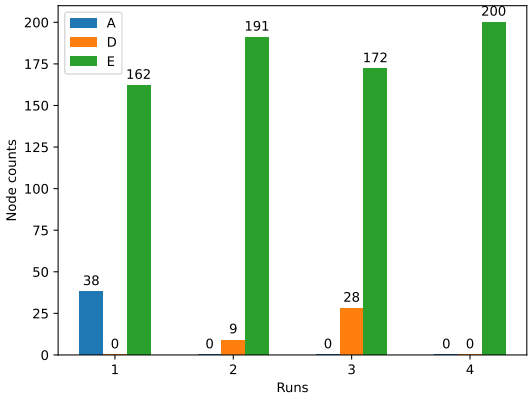


Figure 19: Node distribution in the final step of each run.

### Experiment 04: 1000 nodes, 6 opinions

The experiment features 1000 nodes and 6 opinions ("A", "B", "C", "D", "E", "F") with the following probabilities of opinion retention `(A=0.4, B=0.2, C=0.2, D=0.4, E=0.6, F=0.4)`.

The opinion freeze length (`freeze_length`) was set to 10 rounds. The number of simulation runs (`runs`) was set to 4, with each run finishing (`finish_time`) after 400 rounds.

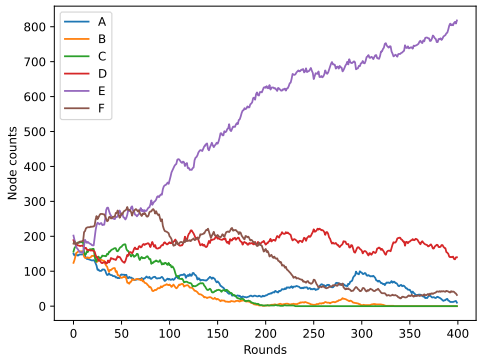


Figure 20: Node distribution of the first run.

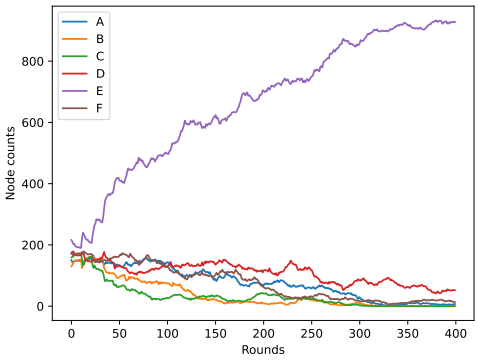


Figure 21: Node distribution of the second run.

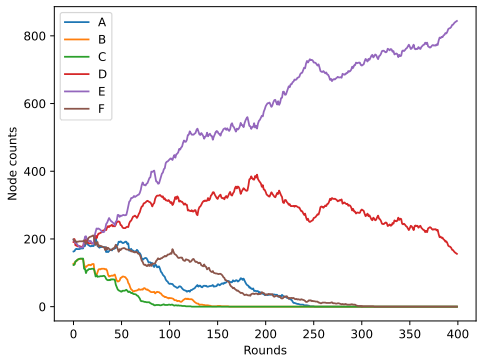


Figure 22: Node distribution of the third run.

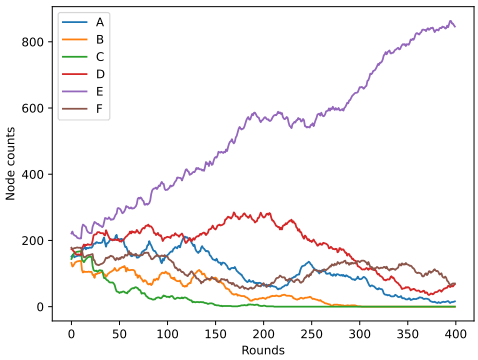


Figure 23: Node distribution of the fourth run.

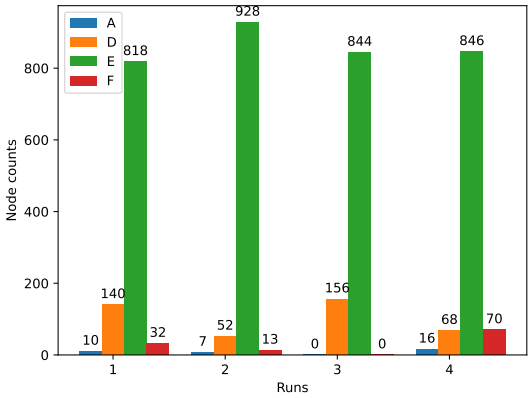


Figure 24: Node distribution in the final step of each run.