Emergence in a real scale free network

Federico Malizia

02/10/2019

UNIVERSITÀ DEGLI STUDI DI TORINO

1 Introduction

The sociological literature is full of works whose nature is to explain the typical characteristics of a social network, through the fundamental characteristics of network analysis it is possible to take conclusions and find interesting patterns that otherwise could not be observed. None of the four fundamental models, circular network, random, small-world, and preferential attachment, are able to faithfully reproduce the typical features of a social network. In this work it was build a toy model using ABM, to observe a hypothetical evolution of a Social Network formed by real users interacting with each other, getting to know each other and establishing virtual friendships.

"Agent-based modelling is the set of techniques [in which] relations and descriptions of global variables are replaced by an explicit representation of the microscopic features of the system, typically in the form of microscopic entities (agents) that interact with each other and their environment according to (often very simple) rules in a discrete space-time." (Gross Strand, 2000, p. 27)

According to Bonabeau (2002), an agent is a computational system that is situated in a dynamic environment and is capable of exhibiting autonomous and intelligent behaviour. Agents could interact, communicate and exchange information with each other. Agent-based modelling is a relatively new paradigm of system modelling. In recent years, there has been more and more research combining social network analysis and agent-based modelling. Manzo and Tubaro (2016) mention several fields that are using both social network analysis and agent-based modelling. These include strategic networks (Buskins et al, 2014) and specifying network topologies (Axtell, 2001). Also, epidemiology has used both social network analysis and agent based modelling to understand the contagion effect and the roles of reciprocity and feedback loops in the spread of disease (El-Sayed et al, 2012).

The aim of this study is to reproduce a dynamical study of a social networks involving multiple users, and how they interract between each other from the first stages of life of the system. It has been implemented a toy model using an agent-based modeling approach (ABM) using several python libraries. The ABM model contains the social network structure, individual behavior and parameters that are obtained from empirical data. In the next sections it will be illustrated the model structure and the agents behavior.

2 The Model

Agents are generated randomly and they have the possibility to approach each other, and consequently also the ability to "respond" to the approach, using the assumption of reciprocity between the two agents. Names are randomly associated to each agent, coming from a list of typical Italian names. From the early stages of their creation they possess unique characteristics that are crucial for the evolution of the network. These unique features have been initialized within the agent in a multidimensional vector form, where each dimension represents an attribute that is assigned to the agent. Those attributes are: district of origin, age, political party and personality. In this study were used Italian provinces and the results of the 2019 european election in Italy. To each one of this attribute, has been associated a real number representing the coordinate that will be assigned to the user's multidimensional features vector. The agents also, once created, have two singular attributes that can be entered manually at the start of the simulation, a coefficient of shyness and a tolerance coefficient. The first one will be crucial in order for the agent to take, or not, the decision to approach one of the users near the time, the second one instead represents a threshold trought a measure of sympathy, which represents what the agents are able to tolerate with respect to the geographical differences and characteristics of other network agents while approaching each other. Once the simulation has been completed, the agents will be divided into 5 categories, so-called "states" of the agents, depending on the degree they got within the network: *Segregated, asocial, sociable, cool and influencers.*

2.1 Agents

Agents assigned to nodes are instances of a class called User

```
class User():
   def __init__(self,name,age,district,party,unique_id,shyness):
      self.name = name
      self.age = age
      self.district = district
      self.party = party
      self.unique_id = unique_id
      self.shyness = shyness
      self.friends_count = 0
      self.friends_dict = {}
      self.sympaties_dict = {}
      self.state = "seg" #segregated
```

The agents are randomly generated, each of them is associated with the initial status of "segregated", since at the time of their creation they are deprived of friends. The "District" and "Party" attributes will be assigned using probability values depending on the population distribution in the Italian provinces and the election results of the European elections of 2019. In this way, the

model will be more realistic as the probability of having a greater number of users from more inhabited districs is higher.

x	Province	p	Popolazione
95	Agrigento	0,0072046599	434870
3	Alessandria	0,0069795754	421284
58	Ancona	0,007807017	471228
1	Aosta	0,0020819573	125666
49	Arezzo	0,0056768817	342654
59	Ascoli Piceno	0,0034324148	207179
4	Asti	0,003555991	214638
83	Avellino	0,0069302377	418306
76	Bari	0,02074227	1251994
77	Barletta - Andria - Trani	0,0064614634	390011
29	Belluno	0,0033623513	202950
84	Benevento	0,0045894646	277018
15	Bergamo	0,0184658447	1114590
5	Biella	0,0029089848	175585
27	Bolzano	0,0088002319	531178

Figure 1: A small portion of districts dataset file

The attributes will be converted into a multidimensional vector thanks to python dictionaries; to each province or political party, a numerical value is associated (in ascending order by geographical proximity regarding the provinces and ordered by "political closeness" regarding the parties). A random number will represents the personality attribute. Once the agents has been created, an undirected graph will be initialized using the NetworkX library, then to each node an agent is assigned. The graph represents the dynamic of a social network, where the edges are precisely the friendships that have been established over time, and the nodes represent the agents. At the initial stage of the network, the nodes will be withouth any link; within the simulation, one agent at a time will move with small steps, looking around for another agent to interact with, once found, the process is divided into two phases: there will be the approaching agent and the agent being approached. During the approach phase, a random number will be generated that if it is greater than or equal to the shyness degree of the approaching agent, then the approach will take place, otherwise the agent will withdraw. When the approaching agent decides to approach, the two agents will begin to interact with each other. Using a sympathy function, the agents will become friends if only the Euclidean distance between the agents features vectors will be less than a "tolerance" value.

```
def look_around(self,N):
   if self.friends_count <= 2:
      randomsteps = [i for i in range(int(-N/10),int(N/10))]
      self.step = np.random.choice(randomsteps)
   if self.friends_count > 2 and self.friends_count <=5:
      randomsteps = [i for i in range(int(-N/5),int(N/5))]
      self.step = np.random.choice(randomsteps)
   if self.friends_count > 5:
      randomsteps = [i for i in range(-N,N)]
      self.step = np.random.choice(randomsteps)
   return(self.step)

def make_your_decision(self):

   if random.random() >= self.shyness:
      decision = True
   else:
      decision = False
   return(decision)

def sympathy(self,friend):
   v_1 = np.array(self.features, dtype = "int")
   v_2 = np.array(friend.features, dtype = "int")
   sympathy_degree = np.linalg.norm(v_1 - v_2)
   self.sympaties_dict[friend.get_id()] = sympathy_degree
   return(sympathy_degree)

def approach(self,friend,threshold,decision):
   success = False
   if friend.get_id() not in self.friends_dict.keys() and
       friend.get_id() !=  self.get_id():
      if decision == True:
         if self.sympathy(friend) <= (threshold*100):
            success = True
            self.add_friend(friend,success)

   return (success)

def got_approached(self,friend,success):
   return(self.add_friend(friend,success))

def add_friend(self,friend, success):
   if success == True :
      self.friends_count += 1
      self.friends_dict[friend.get_id()] = friend
   else:
      pass
   return(success)
```

2.2 States

The agents will all be generated with the status of Segregates, since in the first moments of their generation they will not be connected with any friend. During the simulation, their status will be updated according to the number of friends they will have within the network compared to the maximum degree that can be observed inside it. Here is the list of all the five states that the nodes can have during the simulation:

- The nodes that will not be able to connect with anyone, will take the status of Segregated
- the nodes with a degree less than or equal to 25% of the maximum degree of the network will take the status asocial.
- The nodes with a degree between 25% and 50% of the maximum value reached in the network, will take the name of Socievoli
- The nodes with a degree between 50% and 75% will take the status of Cool users.
- The nodes with a degree greater than or equal to 75% will be called "Influencers", or rather the hubs of the network.

2.3 User interface

```
What is the network size (number of nodes)? 100
Do you want to set a global agent shyness rate? (type -1 for random values or choose a value between 0 and 1)-1
What is the tolerance degree between agents?? (-1 for random values or choose a value between 0 and 1) 0
How many interaction between agents do you want?0
```

At the beginning of the simulation there will be four parameters that can be chosen by the user, which will determine the evolution of the network.

- *Number of nodes*: it represents the number of nodes within the network and consequently the number of agents that will be randomly generated. A greater number of nodes will produce greater randomization and network amalgamation, as there will be more users and therefore greater behavioral diversity among individuals, but will require more computing power. It is not advisable enter a number of nodes larger than 300-400 units.
- *Shyness rate*: Represents the shyness degree of each individual node. It is possible to enter a value between 0 and 1 equal for every node of the network or by entering the value -1 to generate a different random value for each node of the network. The greater the value, the less the agents' tendency to approach with other network agents.
- *Tolerance rate*: Represents the degree of tolerance in the interaction between each pair of agents. Also in this case it will be possible to enter a

value between 0 and 1 or initialize a different random value for each agent. Obviously, the lower the value, the less likely the agents to make friends with evident geographic and behavioral inequalities

- *Number of interactions*: It represents the number of interactions between nodes, successful or failing attempts to approaches, and is therefore the quantity that regulates the time of the simulation. It is advisable to enter a value equal to or greater than the maximum number of edges in an indirect network: $\frac{N(N-1)}{2}$ to observe the evolution of the network up to an equilibrium state between the nodes.

The agents state will be updated in real time according to the distribution of the nodes degree within the graph. In the simulation the nodes will have different colors depending on their state: black for *segregated*, blue for *asocials*, green for *sociable* users, orange for *cool* users, red for *influencers* representing hubs, fraction of nodes that own the higher number of edges within the network.

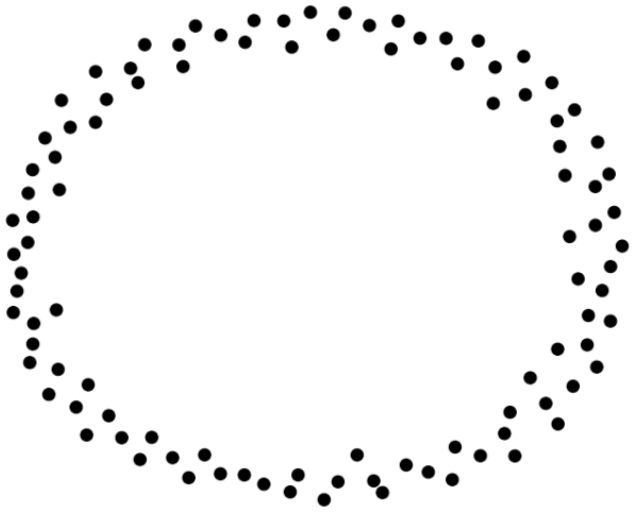
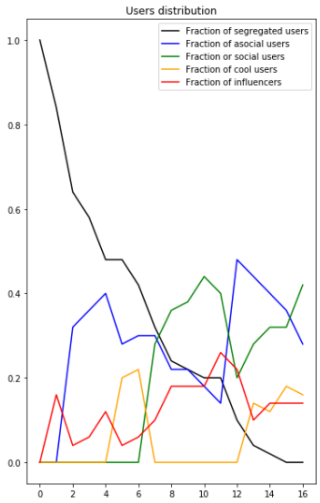
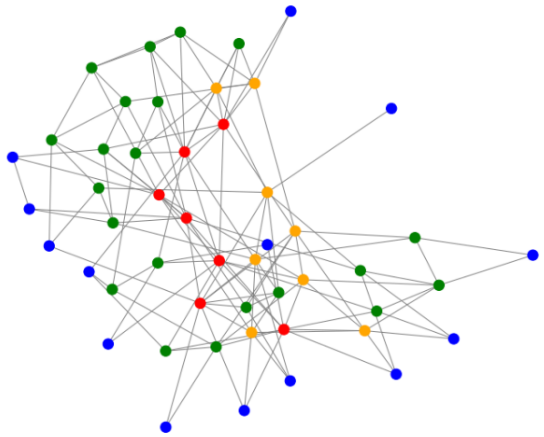


Figure 2: An example of simulation with 100 agents at early stages before they begin to approach. The black color nodes represents the segregate state.

3 Results

In literature there are many works where it is being studied, and tried to replicate, the dynamic structure of a social network. There are many machine learning models which try to predict the evolution of the network from a pre-established pattern. The research is still open to a real formulation of a model that replicates the intrinsic structure of a true social network. There are many studies applied to rumor spreading within social networks and all these have highlighted the difficulty of studying the intrinsic mechanisms of social media functioning, since there are various local mechanisms, for example degree of influence and degree of response of individual users to repeated exposure to all information and long-term and short-term interactions plus the existence of various sub-layers of the network. The following study will not be formalized on rumor spreading, as the nodes will not propagate information to any kind, but simply a toy model using ABM simulation for the interaction of users and how their approach with different users tastes and geographical positions and how they can interract forming a complex network. The following section will report some cases obtained by entering different parameters in the user interface. All cases will be observed with a number of interactions approximately equal to the maximum number of links for an indirect network: $E_{max} = \frac{N(N-1)}{2}$ where N is the number of nodes. The network conformation is totally dependent on the randomization of the nodes, different simulations with the same number of parameters will not be equal since the parameters of the agents are randomly generated.





The above result represents a network consisting of 50 nodes, in which random tolerance and shyness parameters that have been used are different for each node. From the graph representation we can immediately notice how the network apparently takes the form of a real network, the red nodes are the most

connected and takes central position in the graph, while the peripheral nodes (blue color) are the least connected. Despite the low number of randomly generated nodes and a random distribution of the parameters, the following graph confirms a distribution of the near-scale free degree.

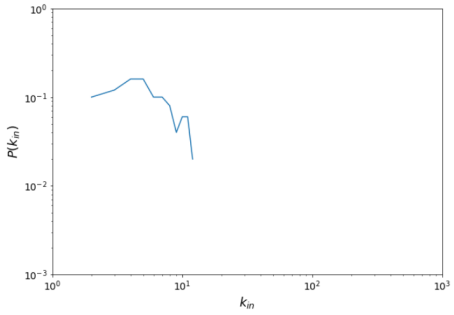


Figure 3: Degree distribution of the network with random values as parameters, the x-axes represents the degree and the y-axes is the fraction of node with that specific degree

In the next representation we could see how by setting a very low tolerance value, many nodes in the network will struggle to establish friendships with other nodes, this will involve a very high fraction of segregated nodes, with small marginal communities formed with a bigger populated component.

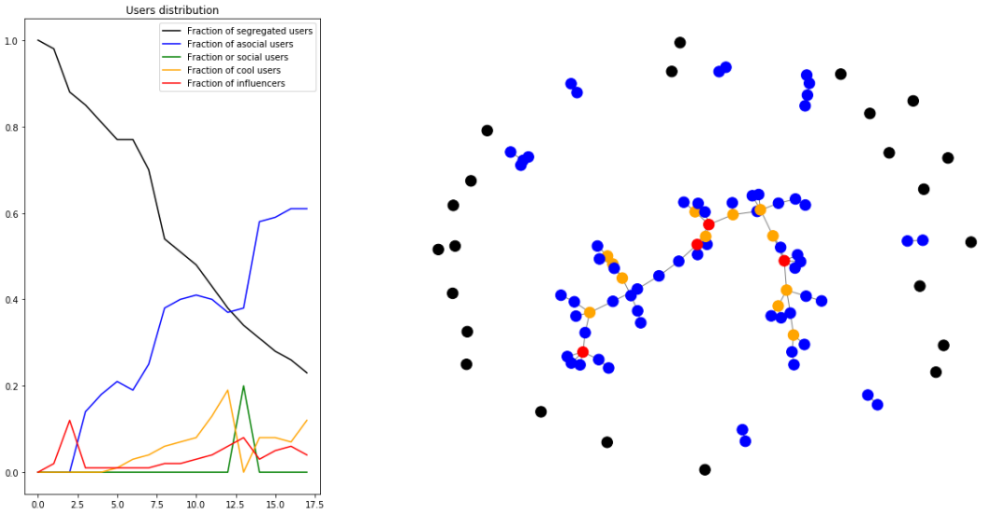


Figure 4: Graphical representation generating 50 intolerant nodes.

In the next representation we will use a higher number of nodes than the first two. Two hundred nodes will be generated with random parameters, but in this case we will see how in addition to how the nodes will be distributed according to their degree, we will see the network conformation according to the users' political preference.

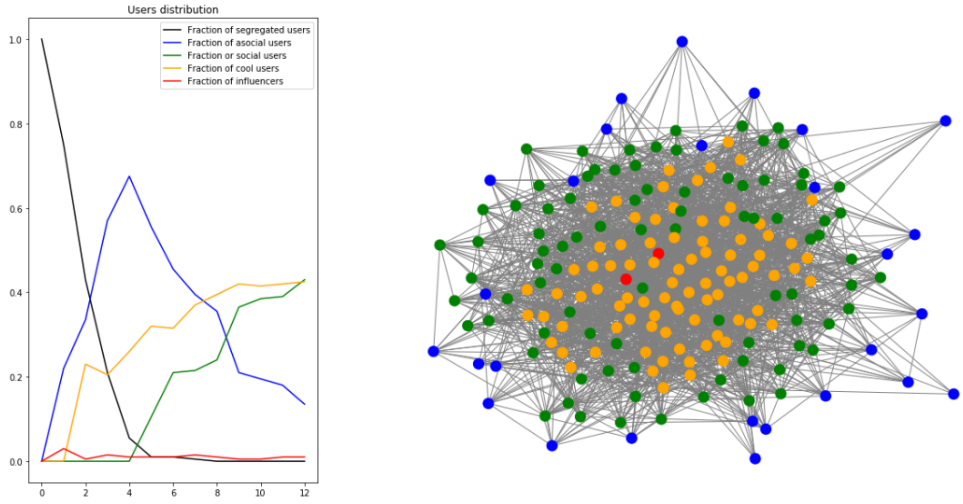


Figure 5: Graphical representation of 200 nodes generated with random parameters

The network is homogeneously connected, in this case the degree distribution will be more similar to a erdos renyi random graph, with a pair of nodes having a very high degree, and advancing towards the bounds, degree decreases almost linearly. The graphical representation according to political preferences sees in red the users who voted for center-left parties, in blue center-right parties, in yellow the ones who preference is for "Movimento 5 Stelle", in black the extreme right-wing parties and in orange for small independent parties.

From the representation we can see that although there is not a clear division of the network according to political preferences, users micro-communities are formed with many users that share the same political ideas and a lot of users that are most likely connected with other nodes having similar political preferences, with the few users belonging to the extreme right wing (who represent a very small portion of the electorate) isolated with each other.

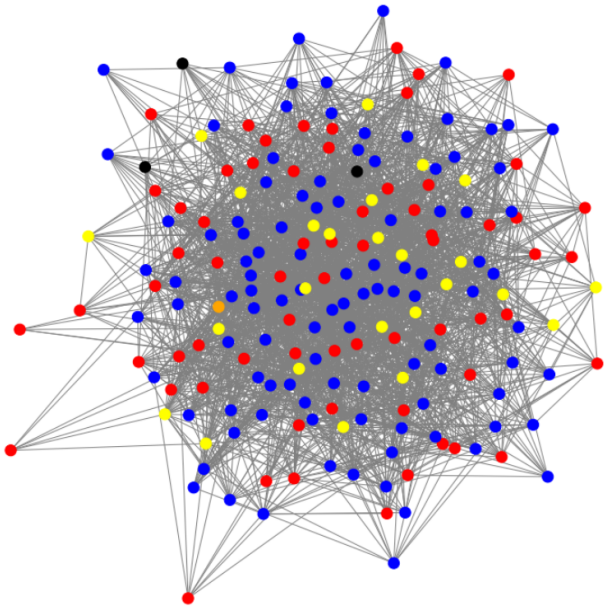


Figure 6: Graph representation by nodes political preference

4 Conclusions

The proposed model represents a small incipit to the potential of agent-based programming for the schematic representation of a social network. This model only with two variable parameters, shyness and tolerance, which in some way represent two of the many macro peculiarity of an average social network user, are not enough to represent the dynamical behavior of a social network. A model with multiple parameters and a larger number of data able to use greater features besides political affiliation and geographical proximity, as for example mutual pages or groups in Facebook, number of mutual friends, musical or culinary tastes. Anyhow agent programming is undoubtedly a very powerful tool that can be useful in this task.