**Virus on Network(Assign2)**

**REPO:** [**https://github.com/ddlxdd/ABM-assignment/tree/main**](https://github.com/ddlxdd/ABM-assignment/tree/main)

I chose to replicate the “Virus on Network” model, which I think is a foundational example within agent-base modeling field. This model simulates the spread of virus through the random selected population. The core purpose of this model is to explore the dynamics of infectious disease like the pandemic we just experienced two years ago within a networked community, demonstrating how individual behaviors and network structures. The model allows for a more complex interactions and the emergent patterns of virus transmission, recovery, and resistance.

**Design concepts**

I think the model adheres to principles from epidemiology and network theory, aiming to reflect the nature of spreading diseases and potentially exploring intervention methods to mitigate outbreaks. The original model categorizes agents into three states: susceptible, infected, and resistant, encapsulating the progression of an epidemic. Interactions among agents are dictated by the underlying network structure, with infection probabilities influenced by these interactions. The stochastic nature of disease spread and individual variance in susceptibility and recovery are central to the model's design.

**Enhancements**

I made two changes to the model:

1. **Watts-Strogatz Small-World Network:**

The original network is relatively simple and random, so I decide to transfer the old network to a Watts-Strogatz small-world network model. This network is characterized by high clustering and short average path lengths. I think in real world, high clustering is a normal phenomenon, which lead individuals tend to form tight groups characterized by a higher-than-average likelihood of connectivity among members. This clustering is typical in social circles where friends are more likely to know each other. Despite the high clustering, most individuals can be reached from any other node by a small number of steps. This phenomenon enhances the capability of a disease to spread rapidly across a diverse population.

1. **Sequential Updates:**

I also modified the update mechanism from a random order of agent updates to sequential updates. This change ensures that all agents are updated in a deterministic order, which can provide insights into the temporal dynamics of disease spread without the variability introduced by random update sequences.

**Details**

The enhanced model initializes with a Watts-Strogatz small-world network of agents, with a subset initially infected to simulate virus introduction. The modifications include not only the network structure but also how the agents are updated during the simulation. The model operates with minimal external data, relying on parameterized simulations to explore scenarios. Parameters include the number of agents, rewiring probability, initial outbreak size, virus spread chance, recovery chance, and the network degree influencing each agent's interaction.

1. **Watts-Strogatz Small-World Network Implementation:**

This change was implemented by modifying the network initialization code in the *VirusOnNetwork* class constructor. Here, I replaced the original network creation line with: s*elf.G = nx.watts\_strogatz\_graph(n=self.num\_nodes, k=nearest\_neighbors, p=rewiring\_prob).* The Watts-Strogatz model introduces a higher clustering coefficient and shorter path lengths, making it a potential better representation of human social networks. The parameters nearest\_neighbors (average degree of each node) and rewiring\_prob (probability of rewiring each edge) were set to optimize the small-world properties. This structural change aims to examine how such network characteristics influence viral spread patterns compared to purely random networks.

1. **Sequential Updating Scheme:**

To address the randomness in the agent updating sequence and make it to a sequential order. I implemented a sequential scheduler. This scheduler updates agents one at a time in the order they were added to the schedule, replacing the original random update mechanism. The new scheduler was incorporated by defining a new class *SequentialScheduler,* extending *mesa.time.BaseScheduler:*

*class SequentialScheduler(BaseScheduler):*

*def step(self):*

*for agent in self.agents:*

*agent.step()*

*self.steps += 1*

*self.time += 1*

I then set this scheduler in the model’s constructor: *self.schedule = SequentialScheduler(self).* This change ensures that every simulation step proceeds through agents in a fixed sequence, which eliminates variations due to the order of updates and allows for clearer observation of the impacts of interventions or changes in agent behavior over time.

**Conclusion**

My version of model introduces elements such as the small-world network, highlight the importance of network topology in the dynamics of epidemic spread. The sequential updates provide a clearer view of the progression and impact of interventions over time. By implementing these changes, the model offers better look into how the structure of the network and the method of processing updates can significantly influence the trajectory of disease spread, providing deeper information for understanding and managing real-world epidemics.