Anna Spiro and Charlotte Gray
701 Written Project Report

In this project, we used a graph approach to create a model of a potential COVID-19 spread on a college campus. We created a user interface to allow administrators to see how changing variables related to an institution's COVID-19 response changes the modeled spread of the virus. Specifically, we allow administrators to manipulate how many people each student has masked or unmasked contact with and the testing strategy that the institution chooses to employ. We also ran three example spreads of COVID-19 in a graph approximating Middlebury's campus and found significant changes in the resulting spread based on the test condition.

At the heart of the code, we have a Person class and a Graph class. The Person class keeps track of an a Peron's contacts as a dictionary with Person IDs as keys and transmission probabilities as values, as well as a Person's COVID-19 state. The Person class includes methods to sample from a distribution to determine how many time steps after infection a Person will start to show COVID-19 symptoms, as well as a method for COVID-19 testing. The Graph class includes a method to probabilistically spread COVID-19 from an infected individual, as well as a method to be called at each time step to increment the overall spread of the virus. The Graph class also has a method to dynamically test a list of Person IDs, as well as a method to randomly assign mutual contacts in the Graph.

On the front-end, we used Flaskapp (which implements Jinja for templating) and Pyvis. The user inputs a set of restrictions and http routing takes them to a slideshow of the graphs for each day. When the user requests to see aggregate data for several simulations, that is available through a dictionary. This could be improved in the future with more time, as we had hoped to include a chart showing average cases over time. Because of initial indecision about the libraries we were going to use, a lot of the code bounces back and forth between Networkx, Pyvis, and our Graph class. We streamlined this, now the front end mainly uses Pyvis for displaying graphs and our Graph class for running the simulation. We use Networkx for layout features, but not a lot else. We use HTML and CSS as well as Javascript and Jinja to display the graphs and handle form submissions. This is a basic implementation, and with a few more weeks could have been much more polished, but due to the steep learning curve of software development in general we aimed more for practicality than appearance.

To test the outcomes of different COVID-19 policies in our model, we ran 1000 simulations with each of three different test conditions approximating variations of Middlebury's COVID-19 plan. In all conditions, we used 2275 students. In condition 1, each Person has 4 close contacts, 5 tangential contacts, and ⅓ of students are tested each week. In condition 2, each Person has 10 close contacts, 5 tangential contacts, and ⅓ of students are tested each week. In condition 4, each Person has 4 close contacts, 5 tangential contacts, and all students are tested each week.

We then analyzed the data from these three conditions in SPSS to look for statistically significant differences in outcome of asymptomatic and quarantined populations. A one-way ANOVA demonstrated a significant effect of condition on final number of asymptomatic cases, $F(2, 2997) = 4566.8$, $p < 0.05$, $R^2 = 0.75$. Tukey's post-hoc tests showed that number of cases in condition 3 ($M = 2.9$, $SD = 5.3$) was significantly lower than the number of cases in condition 1 ($M = 21.5$, $SD = 16.8$; $p < 0.001$) and the number of cases in condition 2 ($M = 320.8$, $SD = 143.6$;

$p < 0.001$). In addition, the number of cases in condition 1 was significantly lower than the number of cases in condition 2 ($p < 0.001$).

A one-way ANOVA demonstrated a significant effect of condition on final number of quarantined cases, $F(2, 2997) = 4031.1$, $p < 0.05$, $R^2 = 0.73$. Tukey's post-hoc tests showed that number of cases in condition 2 ($M = 483.2$, $SD = 229.4$) was significantly higher than both the number of cases in condition 3 ($M = 14.2$, $SD = 18.1$; $p < 0.001$) and condition 1 ($M = 26.0$, $SD = 16.7$).

These results are in line with typical expectations about Middlebury's COVID-19 response, namely that reducing the number of individuals with whom a person has unmasked contact and maintaining a frequent testing schedule are critical factors in controlling a potential COVID-19 spread. Although we can note that these differences are statistically significant, it is extremely important to consider the key assumptions that we make in this model.

First, we assume that an individual self-quarantines on the day they begin to show COVID-19 symptoms. Although this assumption seems reasonable to us in the context of Middlebury's campus, it is unfortunately not always the case. In future work, it would be relevant to consider the role of symptomatic spread. In addition, it is important to note that the constants we used to for transmission probability (17.4% without masks, 3.1% with masks) and the mean and standard deviations defining the number of days after infection a person will become symptomatic (4.98 and 4.83 respectively) are based on relatively small case studies. Furthermore, these numbers may be different for asymptomatic compared to symptomatic transmission. These transmission probabilities are also not specified to be transmission probabilities per day, but we chose to treat them as transmission probabilities per time step.

In addition, although the assignment of contacts in our graph does ensure that contacts are mutual, it assigns contacts randomly. From real social network data, and specifically from the social network data from NSCI 0401 that we analyzed, we know that this is not the case: social networks in real life exhibit significant small-world clustering. Unfortunately, the NSCI 0401 dataset was not complete enough for us to use it in our graph creation. Although we started this project with the intention to connect our graph so as to actually model Middlebury’s campus, we did not realistically have enough data to do so. In addition, we realized that the greatest risk of COVID-19 spread comes from those with whom a person has unmasked or prolonged masked contact, and these individuals are usually people socially connected to the person, rather than connected by academic schedules or building assignment. The benefit of our graph being connected randomly is that our model is more easily applicable to other college campuses or highly-connected groups.

Despite this upside, in future work, it would be relevant to incorporate information on typical social network structure into our model. In addition, if we had significantly more time and data, it would be helpful to test the predictions of this model with real-world case studies. Unfortunately, there have been many instances of real COVID-19 spread on college campuses. Comparing the results of these spreads with our models could help us see how accurate our model is and how we can improve it.

In general, we were challenged by a combination of a lack of available data to use and an excess of seemingly relevant variables to consider. In addition, the front end was a much more complicated undertaking than it initially seemed. Despite these challenges, we were able to use our background from Middlebury computer science courses to approach a socially-relevant issue and create a usable interface. We learned about graph modeling, network structures, probabilistic spread, and became much more familiar with external libraries and new languages like JavaScript, HTML, and CSS. Despite the challenges we faced, we are proud of our execution and of the potential impact of our final result.